1 Przedmioty obowiązkowe

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Logika dla informatyków** **Logic for Computer Science**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOLI**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **pierwszy**
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin, repetytorium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu –
13.	Cele kształcenia dla przedmiotu Wykształcenie umiejętności ścisłego formułowania myśli i poprawnego rozumowania. Zaznajomienie studentów z podstawowymi pojęciami i metodami logiki i teorii mnogości, przydatnymi w pracy informatyka.
14.	Treści programowe 1. Zasada indukcji. 2. Składnia i semantyka rachunku zdań i rachunku predykatów. Pojęcie spełniania i prawdziwości formuł. Niesprzeczność zbioru formuł. 3. Podstawowe pojęcia teoriomnogościowe i operacje na zbiorach: suma, iloczyn, iloczyn kartezjański, zbiór potęgowy, relacje, funkcje, relacje równoważności, klasy abstrakcji, zbiór ilorazowy. 4. Moce zbiorów. Zbiory skończone i nieskończone. Zbiory przeliczalne i zbiory mocy continuum. Twierdzenia Cantora i Cantora-Bernsteina. 5. Porządki częściowe i liniowe. Dobre porządki. Indukcja noetherowska. 6. Unifikacja termów. Informacja o metodzie rezolucji. 7. Dowodzenie twierdzeń. Informacja o systemie naturalnej dedukcji.

15. Zakładane efekty uczenia się

Wiedza

zna wybrane, podstawowe pojęcia teoriomnogościowe (zbiory i operacje na zbiorach), pojęcie mocy zbioru, pojęcia porządków częściowych i liniowych	K_W02, K_W03, K_W14
zna rachunek zdań i predykatów, pojęcia spełnialności i prawdziwości oraz metody dowodzenia (indukcja, rezolucja, dedukcja)	K_W01, K_W02, K_W14

Umiejętności

posługuje się rachunkiem zdań i kwantyfikatorów	K_U02
potrafi przeprowadzać ścisłe rozumowanie z wykorzystaniem matematycznych metod dowodzenia (indukcji, rezolucji, dedukcji)	K_U01
potrafi stosować pojecia teoriomnogościowe do analizy i modelowania problemów w informatyce	K_U02, K_U22
potrafi precyzyjnie formułować pytania, służące pogłębieniu własnego zrozumienia danego tematu lub odnalezieniu brakujących elementów rozumowania	K_U01, K_U22

Kompetencje społeczne

rozumie i docenia znaczenie uczciwości intelektualnej w działaniach własnych i innych osób, postępuje etycznie	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

1. Wojciech Guzicki, Piotr Zakrzewski, Wykłady ze wstępu do matematyki. Wprowadzenie do teorii mnogości, PWN, Warszawa 2005.
2. Wojciech Guzicki, Piotr Zakrzewski, Wstęp do matematyki. Zbiór zadań, PWN, Warszawa 2005.
3. Kazimierz Kuratowski, Wstęp do teorii mnogości i topologii, PWN, Warszawa 2004.
4. Wiktor Marek, Janusz Onyszkiewicz, Elementy logiki i teorii mnogości w zadaniach, PWN, Warszawa 2005.
5. Helena Rasiowa, Wstęp do matematyki współczesnej, PWN, Warszawa 2007.
6. Jerzy Tiuryn, Wstęp do teorii mnogości i logiki, Skrypt Uniw. Warszawskiego, 1994.

17. Metody weryfikacji zakładanych efektów uczenia się

prezentacja rozwiązania zadania, egzamin, sprawdziany/kolokwia

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Ocena z ćwiczeń jest wystawiana na podstawie uzyskanej liczby punktów. Punkty otrzymuje się za przygotowane rozwiązania zadań (na podstawie deklaracji weryfikowanych w trakcie ćwiczeń), kartkówki i kolokwia.
Aby zaliczyć wykład należy zdobyć minimalną wymaganą liczbę punktów z końcowego egzaminu pisemnego. Szczegółowe warunki zaliczenia przedmiotu są określone w regulaminie przedmiotu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
repetytorium	30 godz.
ćwiczenia	30 godz.

Praca własna studenta

przygotowanie do ćwiczeń	45 godz.
przygotowanie do egzaminu	15 godz.
studiowanie tematyki wykładów i literatury	15 godz.
przygotowanie do sprawdzianów/kolokwiów	15 godz.

Sumarycznie

Łączna liczba godzin	180 godz.
Liczba punktów ECTS	8

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Logika dla informatyków (zaawansowana)** **Logic for Computer Science (advanced)**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOLZ**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **pierwszy**
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Przedmiot adresowany jest głównie do studentów I roku kierunku ISIM. O takich studentach zakłada się, że mają kulturę matematyczną wykraczającą ponad wymagania typowej szkoły średniej.
13.	Cele kształcenia dla przedmiotu Cel przedmiotu jest trojaki. • Po pierwsze jest to kurs „wstępu do matematyki”. Oznacza to, że przedmiot ma przygotować słuchaczy, pod względem wiedzy i umiejętności, do kontynuowania studiów na kierunku ISIM, w szczególności do, następującego w kolejnych dwóch semestrach, przedmiotu „Algebra”. • Po drugie jest to kurs logiki (dla informatyków). To znaczy ma uświadomić słuchaczom (na kilku wybranych przykładach) relację między językiem a semantyką. • Po trzecie (ale nie rozłączne z pozostałymi dwoma celami) przedmiot ma na celu przygotować słuchaczy do czytania tekstu matematycznego w tym sensie, że mają oni się nauczyć, że staranne przeczytanie tekstu, i nawet zrozumienie każdego zdania z osobna, nie jest dostatecznym warunkiem rozumienia konstrukcji matematycznej.

14. Treści programowe

Wszystkie treści przedmiotu „Logika dla Informatyków", a ponadto:

Z zakresu „wstępu do matematyki”:

- Pojęcia izomorfizmu i homomorfizmu, z licznymi przyładami.
- Konstrukcja ilorazowa, z licznymi przykładami (w tym konstrukcja liczb całkowitych, wymiernych i rzeczywistych).
- Liczby porządkowe i arytmetyka porządkowa, wraz z normalnie omawianymi w tym kontekście twierdzeniami (lemat Zorna, twierdzenie o dobrym uporządkowaniu).
- Przykłady dowodów terminacji procesów używające argumentów odwołujących się do liczb porządkowych.

Z zakresu „logiki dla informatyków”:

- logika I rzędu
- logika z operatorem punktu stałego (jako kolejny przykład użycia liczb porządkowych), uwaga o logice drugiego rzędu.
- Pojęcie semantyki.
- Pojęcie dowodu.
- Pojęcie poprawności i zupełności systemu dowodowego.
- Uwagi o systemach aksjomatycznych. Przykłady: arytmetyka Peano, teoria mnogości. Uwaga o niezupełności.
- Moc wyrażania logik. Gry Ehrenfeuchta–Fraïsségo.

W ramach ćwiczeń uczestnicy zajęć przeczytają fragment zaawansowanego tekstu matematycznego w języku angielskim (artykułu lub monografii).

15. Zakładane efekty uczenia się

Wiedza

zna wszystkie pojęcia z programu przedmiotu „Logika dla Informatyków", a spośród pojęć spoza tego programu pojęcia izomorfizmu, homomorfizmu i konstrukcję ilorazową	K_W01, K_W02, K_W03, K_W14
zna logikę pierwszego rzędu, ma świadomość istnienia innych logik oraz świadomość metody aksjomatycznej	K_W02, K_W03
zna podstawowe fakty związane z liczbami porządkowymi	K_W02, K_W03

Umiejętności

swobodnie posługuje się wszystkimi pojęciami z programu przedmiotu „Logika dla Informatyków”, a dość swobodnie pojęciami izomorfizmu, homomorfizmu i konstrukcją ilorazową	K_U01, K_U02
ma pewne doświadczenie w samodzielnym zmaganiu się z trudnym tekstem matematycznym	K_U22

Kompetencje społeczne

rozumie i docenia znaczenie uczciwości intelektualnej w działaniach własnych i innych osób, postępuje etycznie	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

1. Wojciech Guzicki, Piotr Zakrzewski, Wykłady ze wstępu do matematyki. Wprowadzenie do teorii mnogości, PWN, Warszawa 2005.
2. Wojciech Guzicki, Piotr Zakrzewski, Wstęp do matematyki. Zbiór zadań, PWN, Warszawa 2005.
3. Kazimierz Kuratowski, Wstęp do teorii mnogości i topologii, PWN, Warszawa 2004.
4. Wiktor Marek, Janusz Onyszkiewicz, Elementy logiki i teorii mnogości w zadaniach, PWN, Warszawa 2005.
5. Jerzy Tiuryn, Wstęp do teorii mnogości i logiki, Skrypt Uniw. Warszawskiego, 1994.

17. Metody weryfikacji zakładanych efektów uczenia się

prezentacja rozwiązania zadania, egzamin

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Egzamin pisemny (wspólny z przedmiotem „Logika dla informatyków") - warunki określone przez regulamin przedmiotu „Logika dla informatyków" . Zaliczenie ćwiczeń, według zasad określonych przez regulamin ćwiczeń.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
ćwiczenia	30 godz.

Praca własna studenta

przygotowanie do ćwiczeń	55 godz.
przygotowanie do egzaminu	15 godz.
studiowanie tematyki wykładów i literatury	40 godz.
przygotowanie do sprawdzianów/kolokwiów	15 godz.

Sumarycznie

Łączna liczba godzin	185 godz.
Liczba punktów ECTS	8

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Metody programowania** **Programming methodology**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOMP**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **pierwszy**
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 45 godzin, ćwiczenio-pracownia — 45 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Logika dla informatyków • Wstęp do informatyki **Wymagane kompetencje** • Podstawowa umiejętność wnioskowania logicznego • Wskazana bierna znajomość języka angielskiego
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest przygotowanie studentów do używania języków programowania jako formalnych narzędzi komunikacji między ludźmi, w szczególności jako technik, które mogą i powinny być stosowane do zapanowania nad złożonością intelektualną dużych systemów. W tym celu zajmiemy się zarówno konkretnymi narzędziami, których projektanci języków programowania dostarczają programistom, jak i ogólnymi zasadami konstrukcji języków programowania.

14.	Treści programowe

1. Elementy programowania
2. Rekursja i abstrakcja proceduralna
3. Wprowadzenie do abstrakcji danych
4. Dane hierarchiczne i własność domknięcia
5. Dane symboliczne i wielorakie reprezentacje danych
6. Składnia abstrakcyjna i podstawy ewaluacji
7. Środowiskowy model obliczeń
8. Procedury i domknięcia
9. Reprezentacja wartości w metajęzyku
10. Ewaluator metacykliczny
11. Zarządzanie złożonością: kontrakty
12. Zarządzanie złożonością: systemy typów
13. Obliczenia z danymi modyfikowalnymi
14. Elementy programowania współbieżnego
15. Programowanie w logice

15. Zakładane efekty uczenia się

Wiedza

zna wybrane modele obliczania wartości programów i podstawowe metody wnioskowania o ich poprawności	K_W02, K_W04, K_W14
rozumie pojęcie i cel stosowania abstrakcji danych, a także sposoby budowania indukcyjnych struktur danych	K_W02, K_W04, K_W14
ma przeglądową wiedzę na temat sposobów zarządzania złożonością większych programów	K_W04, K_W14

Umiejętności

rotrafi przedstawić rozwiązania problemów używając formalizmu funkcyjnego języka programowania	K_U03, K_U16, K_U19
potrafi zaprojektować reprezentację danych odpowiednią dla rozważanego problemu	K_U17, K_U19, K_U22
potrafi zadać semantykę prostego języka poprzez zdefiniowanie ewaluatora	K_U03, K_U17

Kompetencje społeczne

rozumie znaczenie języków programowania jako narzędzia precyzyjnej komunikacji międzyludzkiej	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

- H. Abelson, G.J. Sussman, J. Sussman, Struktura i interpretacja programów komputerowych, WNT, Warszawa 2002, https://mitpress.mit.edu/sicp/ (obowiązkowa)
- D.P. Friedman, M. Felleisen, The Little Schemer, MIT Press, 1995, https://mitpress.mit.edu/books/little-schemer (uzupełniająca)
- M. Felleisen, R. Findler, M. Flatt, S. Krishnamurthi, How to Design Programs, MIT Press, 2018, http://www.htdp.org/ (uzupełniająca)
- F. Bice, R. DeMaio, S. Florence, M. Lin, S. Lindeman, N. Nussbaum, E. Peterson, R. Plessner, D. Van Horn, M. Felleisen, C. Barski, Realm of Racket, no starch press, 2013, http://realmofracket.com/ (uzupełniająca)

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin pisemny lub pisemno-ustny, kolokwium, napisanie programu komputerowego, prezentacja rozwiązania zadania

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia ćwiczenio-pracowni należy:
- uzyskać wymaganą regulaminem przedmiotu liczbę punktów z kolokwium zaliczeniowego;
- uzyskać wymaganą regulaminem przedmiotu liczbę punktów za rozwiązania programistycznych zadań domowych;
- uzyskać wymaganą regulaminem przedmiotu liczbę zaliczonych obecności na ćwiczeniach.

Do zaliczenia niezbędne jest jednoczesne spełnienie wszystkich powyższych warunków.
Egzamin ma formę pisemną lub pisemno-ustną, ustaloną regulaminem przedmiotu.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	45 godz.
ćwiczenio-pracownia	45 godz.

Praca własna studenta

Konsultacje: w ramach indywidualnych potrzeb	0 godz.
Rozwiązanie programistycznych zadań domowych	60 godz.
Przygotowanie do ćwiczeń	30 godz.
Udział w kolokwiach i egzaminie	5 godz.
Utrwalenie materiału i przygotowanie do egzaminu	10 godz.

Sumarycznie

Łączna liczba godzin	195 godz.
Liczba punktów ECTS	9

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Analiza numeryczna (M)** **Numerical analysis (M)**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOANM**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **drugi**
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 60 godzin, ćwiczenia — 30 godzin, pracownia — 15 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu 1. Analiza matematyczna (ciągi, funkcje, granice, szeregi, różniczkowanie, szereg Taylora, całkowanie). 2. Algebra liniowa (algebra macierzy, układy równań liniowych, pojęcie bazy, jej wymiar i ortogonalizacja). 3. Umiejętność programowania w języku umożliwiającym obliczenia zmiennopozycyjne.
13.	Cele kształcenia dla przedmiotu Celem zajęć jest przedstawienie podstawowych metod i algorytmów rozwiązywania typowych zadań obliczeniowych. Omawiane zagadnienia mają wielorakie zastosowania m.in. w obliczeniach naukowych czy grafice komputerowej.
14.	Treści programowe 1. Analiza błędów. Arytmetyka numeryczna. Uwarunkowanie zadania. Algorytmy numerycznie poprawne. 2. Rozwiązywanie równań nieliniowych. Ogólna teoria metod iteracyjnych. Metody: bisekcji, Newtona i siecznych. 3. Interpolacja. Wzór interpolacyjny Lagrange'a. Reszta wzoru interpolacyjnego. Wzór interpolacyjny Newtona. Interpolacja Hermite'a. Interpolacja za pomocą funkcji sklejanych III stopnia. Krzywe Beziera i ich zastosowanie w grafice komputerowej. 4. Aproksymacja. Aproksymacja średniokwadratowa za pomocą wielomianów - wielomiany ortogonalne, twierdzenie o n-tym wielomianie optymalnym. Aproksymacja jednostajna - twierdzenie o alternansie, informacja o algorytmie Remeza konstrukcji wielomianu optymalnego, wielomiany prawieoptymalne. 5. Kwadratury. Kwadratura liniowa. Reszta i rząd kwadratury. Zbieżność ciągu kwadratur. Kwadratury interpolacyjne. Kwadratury Newtona-Cotesa. Wzory złożone: trapezów i Simpsona. Metoda Romberga. Kwadratury Gaussa. 6. Rozwiązywanie układów równań liniowych. Uwarunkowanie zadania. Rozkład macierzy kwadratowej na iloczyn macierzy trójkątnych. Metoda eliminacji Gaussa. Iteracyjne metody rozwiązywania układów równań liniowych.

15.	Zakładane efekty uczenia się

Wiedza

zna i rozumie podstawowe pojęcia, metody i algorytmy numeryczne (arytmetyka zmiennopozycyjna, teoria błędów, interpolacja i aproksymacja funkcji, rozwiązywanie równań nieliniowych, całkowanie numeryczne, metody numeryczne algebry liniowej)	K_W02, K_W06, K_W14
zna numeryczne podstawy grafiki komputerowej	K_W14

Umiejętności

potrafi zaimplementować poznane algorytmy numeryczne i stosować je do rozwiązywania podstawowych zadań matematyki obliczeniowe	K_U05, K_U14
potrafi samodzielnie analizować wybrane zagadnienia, dyskutować i prezentować wnioski przed grupą	K_U19, K_U22
potrafi wykorzystać zdobytą wiedzę z zakresu analizy numerycznej do formułowania i rozwiązywania zadań obliczeniowych stosując metody analityczne, symulacyjne i eksperymentalne oraz interpretować uzyskane wyniki i wyciągać wnioski	K_U05, K_U14, K_U22

Kompetencje społeczne

jest świadom ograniczeń własnej wiedzy i umiejętności w zakresie analizy numerycznej, adekwatnie ocenia poziom swoich kompetencji, zna swoje mocne i słabe strony oraz rozumie konieczność doskonalenia w tym zakresie	K_K01
jest świadom roli i znaczenia wiedzy z zakresu analizy numerycznej w rozwiązywaniu problemów o charakterze poznawczym i praktycznym, w tym dla funkcjonowania współczesnego społeczeństwa, zna niebezpieczeństwa z tym związane	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

Literatura

- A. Björck, G. Dahlquist, Metody numeryczne, PWN, 1987.
- M. Dryja, J.i M. Jankowscy, Przegląd metod i algorytmów numerycznych, cz. 1 i 2, WNT, 1988.
- D. Kincaid, W. Cheney, Analiza numeryczna, WNT, 2005.
- G. Dahlquist, A. Björck, Numerical Methods in Scientific Computing, Vol. I, SIAM, 2008.
- W. Gautschi, Numerical Analysis. An Introduction, Birkhäuser, 1997.
- G. Hämmerlin, K.-H. Hoffmann, Numerical Mathematics, Springer-Verlag, 1991.
- A. Quarteroni, R. Sacco, F. Saleri, Numerical Mathematics, Springer- Verlag, 2000.

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin pisemny, ćwiczenia prezentacja i analiza rozwiązań zadanych problemów, pisemne opracowanie sprawozdań z rozwiązaniem zadanych problemów

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- wykład egzamin pisemny
- ćwiczenia prezentacja i analiza rozwiązań zadanych problemów (na podstawie deklaracji), sprawdzian
- pracownia pisemne opracowanie sprawozdań z rozwiązaniem zadanych problemów (na podstawie deklaracji), sprawdzian

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	60 godz.
pracownia	15 godz.
ćwiczenia	30 godz.

Praca własna studenta

przygotowanie do egzaminu	30 godz.
przygotowanie do sprawdzianów/kolokwiów	15 godz.
przygotowanie raportu/prezentacji	10 godz.
praca nad projektem	50 godz.
przygotowanie do pracowni	30 godz.
studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do ćwiczeń	20 godz.

Sumarycznie

Łączna liczba godzin	290 godz.
Liczba punktów ECTS	12

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Matematyka dyskretna (M)** **Discrete Mathematics**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOMDM**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **drugi**
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 45 godzin, ćwiczenia — 45 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Ogólna kultura matematyczna lub • Logika dla informatyków • Analiza • Algebra
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest przedstawienie elementów matematyki, które są szczególnie użyteczne dla informatyków i nie są częścią standardowego wykładu algebry, analizy czy rachunku prawdopodobieństwa. W szczególności są to przydatne informatykowi elementy teorii liczb, kombinatoryki i teorii grafów. Wiedza w tym zakresie jest przydatna w konstruowaniu i analizie algorytmów, w tym weryfikacji poprawności i szacowaniu złożoności obliczeniowej.

14. Treści programowe

A. Elementy Algebry i Teorii Liczb

- Funkcje całkowitoliczbowe, arytmetyka modularna, operacje sufit i podłoga zaokrąglania liczb rzeczywistych, algorytm mergesort.
- Asymptotyka funkcji liczbowych z uwzględnieniem zastosowań w szacowaniu złożoności czasowej algorytmów.
- Podzielność liczb, algorytm Euklidesa.
- Liczby Fibonacciego.
- Liczby pierwsze i względnie pierwsze. Rozkład na czynniki. Funkcja Eulera. Chińskie twierdzenie o resztach. Twierdzenie Eulera

B. Kombinatoryka

- Rozmieszczenia, permutacje, kombinacje, podziały (zbioru, liczby), Lemat Burnside'a.
- Metody generowania prostych obiektów kombinatorycznych.
- Przykłady prostych problemów definiowanych rekurencyjnie.
- Rozwiązywanie równań rekurencyjnych, funkcje tworzące.
- Liczby Catalana.
- Zasada włączania i wyłączania.

C. Teoria grafów

- Definicja i przykłady grafów, grafy pełne, dwudzielne skierowane, stopień wierzchołka.
- Drogi i cykle w grafach: grafy spójne i dwudzielne.
- Drzewa - równoważność różnych definicji.
- Komputerowa reprezentacja grafów.
- Metody BFS i DFS przeszukiwania grafów.
- Minimalne drzewa rozpinające - algorytmy Kruskala i Prima-Dijkstry.
- Przechodnie domkniecie: algorytmy Dijkstry i Warshalla. Złożoność problemu.
- Cykle i drogi Eulera.
- Cykle i drogi Hamiltona tw. Ore i wielomianowa redukcja problemu drogi do cyklu i odwrotnie.
- Grafy planarne. Tw. Kuratowskiego i wzór Eulera.
- Przepływy w sieciach.
- Kolorowanie grafów: zastosowanie - planowanie sesji egzaminacyjnej. Algorytm sekwencyjny i twierdzenie o 5-kolorowaniu grafów planarnych.

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe sposoby zliczania i wyliczania obiektów kombinatorycznych	K_W02, K_W05, K_W14
zna metody rozwiązywania równań rekurencyjnych	K_W02, K_W05
zna podstawowe pojęcia, twierdzenia i algorytmy teorii liczb	K_W02, K_W05
zna notację asymptotyczną i sposoby szacowania i porównywania funkcji liczbowych	K_W02, K_W05, K_W14
zna podstawowe pojęcia, twierdzenia i algorytmy teorii grafów	K_W02, K_W05

Umiejętności

potrafi zliczać i wyliczać obiekty kombinatoryczne	K_U04, K_U22
potrafi rozwiązywać równania rekurencyjne	K_U04, K_U22
potrafi szacować i porównywać złożoności czasowe i pamięciowe algorytmów	K_U04, K_U22
umie zastosować pojęcia i twierdzenia teorii liczb	K_U04
potrafi stosować pojęcia, twierdzenia i algorytmy podstawowej teorii grafów	K_U04

Kompetencje społeczne

potrafi przejrzyście (w mowie i na piśmie) przedstawić rozumowania z dziedziny matematyki dyskretnej	K_K02, K_K06

16.	Literatura obowiązkowa i zalecana • Charles R.B. Wright, Kenneth A. Ross: Matematyka dyskretna • Ronald L. Graham, Donald E. Knuth, Oren Patashnik: Matematyka Konkretna • Witold Lipski: Kombinatoryka dla programistów • Victor Bryant: Aspekty kombinatoryki • Robin J. Wilson: Wprowadzenie do teorii grafów
17.	Metody weryfikacji zakładanych efektów uczenia się Egzamin pisemny, prezentacja zadań przy tablicy, opracowanie zadań pisemnych
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia ćwiczeń należy oddać na kartkach nie mniejszą od 1/3 poprawnych rozwiązań zadań z każdej listy zadań (można nie zaliczyć maksimum 3 list).
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	45 godz.
ćwiczenia	45 godz.

Praca własna studenta

przygotowanie do egzaminu	40 godz.
rozwiązywanie zadań z list	90 godz.
udział w egzaminie	4 godz.

Sumarycznie

Łączna liczba godzin	224 godz.
Liczba punktów ECTS	9

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Algorytmy i struktury danych (L)** **Algorithms and Data Structures**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOASDL**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **drugi**
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 45 godzin, ćwiczenio-pracownia — 60 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Kompetencje równoważne z zaliczeniem przedmiotów Wstęp do Informatyki i Matematyka Dyskretna
13.	Cele kształcenia dla przedmiotu Celem przedmiotu przedstawienie podstawowych struktur danych i algorytmów oraz osiągnięcie zakładanych efektów kształcenia.

14. Treści programowe

1. Przegląd metod projektowania efektywnych algorytmow: dziel i zwyciężaj, programowanie dynamiczne, metoda zachłanna.
2. Złożoność obliczeniowa algorytmu (pesymistyczna, oczekiwana, zamortyzowana). Przykłady analizy kosztu.
3. Sortowanie: Heapsort i Quicksort. Model drzew decyzyjnych i dolne ograniczenie na problem sortowania. Sortowanie w czasie liniowym: Countsort, Radixsort, Bucketsort.
4. Selekcja: algorytmy Hoarea i magicznych piątek.
5. Kolejki priorytetowe: kopce binarne, dwumianowe i Fibonacciego. Zastosowania w problemie najkrotszych ścieżek i minimalnego drzewa rozpinającego.
6. Scalanie. Drzewa turniejowe. Sortowanie zewnętrzne.
7. Wyszukiwanie i problem słownika. Drzewa wyszukiwań binarnych, zrownoważone drzewa wyszukiwań binarnych (AVL, 2-3-4-drzewa, drzewa czerwono-czarne). Optymalne drzewa wyszukiwań binarnych. Haszowanie. Wyszukiwanie pozycyjne.
8. Wyszukiwanie zewnętrzne - B-drzewa.
9. Problem sumowania zbiorow rozłącznych i jego zastosowania.
10. Algorytmy grafowe: przepływy w sieciach, skojarzenia.
11. Algorytmy na tekstach. Wyszukiwanie wzorca. Drzewa sufiksowe.
12. Geometria obliczeniowa. Lokalizacja punktu. Otoczka wypukła. Technika zamiatania.
13. Algorytmy algebraiczne i teorioliczbowe. FFT. Szybkie mnożenie liczb i wielomianow.
14. NP-zupełność. Algorytmy aproksymacyjne dla problemow obliczeniowo trudnych. Heurystyki dla problemow trudnych (algorytmy genetyczne, simulated annealing).
15. Modele obliczeń rownoległych: PRAM, tablica procesorow, hiperkostka. Algorytmy rownoległe. Klasa NC i problemy P-zupełne.
16. Specjalne modele obliczeń: sieci komparatorow, obwody logiczne.
17. Algorytmy randomizacyjne. Przykłady w dziedzinach: struktury danych, geometria obliczeniowa, algorytmy grafowe, algorytmy rownoległe.

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe struktury danych oraz ich właściwości	K_W02, K_W08
zna i rozumie wybrane techniki i paradygmaty konstrukcji algorytmów	K_W02, K_W07, K_W14
posiada wiedzę na temat znanych rozwiązań dla typowych zagadnień algorytmicznych	K_W02, K_W08, K_W14
rozumie podstawowe pojęcia i techniki z zakresu analizy złożoności obliczeniowej algorytmów	K_W02, K_W07

Umiejętności

potrafi zastosować poznane struktury danych do konstrukcji algorytmów dla wybranych zadań obliczeniowych	K_U06, K_U19, K_U21
umie dowodzić poprawności konstruowanych algorytmów	K_U19, K_U21
umie analizować złożoność obliczeniową (czasową i pamięciową) badanych algorytmów	K_U06, K_U13, K_U19, K_U21, K_U22
potrafi pisemnie przedstawić argumenty dotyczące poprawności i złożoności obliczeniowej wybranych rozwiązań algorytmicznych	K_U06, K_U13, K_U19, K_U21, K_U22

Kompetencje społeczne

potrafi opowiadać o problemach algorytmicznych w sposób gwarantujący zrozumienie treści przez słuchaczy	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Wprowadzenie do algorytmów, PWN, 2012.
- Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman; Algorytmy i struktury danych; Helion.
- Dexter Kozen, The Design and Analysis of Algortihms, Springer 1990.
- Jon Kleinberg, Eva Tardos, Algorithm Design, Pearson 2006.
- Notatki do wykładu.

17.	Metody weryfikacji zakładanych efektów uczenia się prezentacja rozwiązania zadania, napisanie programu komputerowego, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Aby zaliczyć przedmiot, należy uzyskać zaliczenie pracowni i zaliczenie ćwiczeń oraz zdać egzamin. Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie pracowni, liczbę punktów za napisanie programów rozwiązujących zadane problemy algorytmiczne. Liczba punktów za poszczególne programy zależy od ich poprawności i efektywności. Zależność ta jest szczegółowo opisana w regulaminie pracowni. Do zaliczenia ćwiczeń należy zdobyć wymaganą, podaną w regulaminie ćwiczeń, liczbę punktów, uzyskanych za deklaracje rozwiązań oraz ich prezentacje. Egzamin ma formę pisemną i składa się z dwóch części. Podczas pierwszej części, trwającej 2,5 godziny, studenci otrzymują 20 pytań lub bardzo prostych zadań, mających na celu sprawdzenie stopnia opanowania wiedzy. W drugiej części, trwającej 4,5 godziny, studenci otrzymują sześć zadań, spośród których powinni wybrać i rozwiązać trzy dowolne. Te zadania sprawdzają stopień opanowania umiejętności określone w zakładanych efektach uczenia się. Do zdania egzaminu należy uzyskać wymagane, określone w regulaminie przedmiotu, liczby punktów za poszczególne części egzaminu.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** wykład: 45 godz. ćwiczenio-pracownia: 60 godz. **Praca własna studenta** studiowanie notatek i literatury: 15 godz. rozwiązywanie zadań z list: 60 godz. udział w egzaminie: 7 godz. przygotowanie do egzaminu: 35 godz. rozwiązywanie zadań programistycznych: 60 godz. **Sumarycznie** Łączna liczba godzin: 272 godz. Liczba punktów ECTS: 12

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Algorytmy i struktury danych (M)** **Algorithms and Data Structures**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOASDL**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **drugi**
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 60 godzin, ćwiczeńo-pracownia — 60 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Kompetencje równoważne z zaliczeniem przedmiotów Wstęp do Informatyki i Matematyka Dyskretna
13.	Cele kształcenia dla przedmiotu Celem przedmiotu przedstawienie podstawowych struktur danych i algorytmów oraz osiągnięcie zakładanych efektów kształcenia.

14. Treści programowe

1. Przegląd metod projektowania efektywnych algorytmow: dziel i zwyciężaj, programowanie dynamiczne, metoda zachłanna.
2. Złożoność obliczeniowa algorytmu (pesymistyczna, oczekiwana, zamortyzowana). Przykłady analizy kosztu.
3. Sortowanie: Heapsort i Quicksort. Model drzew decyzyjnych i dolne ograniczenie na problem sortowania. Sortowanie w czasie liniowym: Countsort, Radixsort, Bucketsort.
4. Selekcja: algorytmy Hoarea i magicznych piątek.
5. Kolejki priorytetowe: kopce binarne, dwumianowe i Fibonacciego. Zastosowania w problemie najkrotszych ścieżek i minimalnego drzewa rozpinającego.
6. Scalanie. Drzewa turniejowe. Sortowanie zewnętrzne.
7. Wyszukiwanie i problem słownika. Drzewa wyszukiwań binarnych, zrownoważone drzewa wyszukiwań binarnych (AVL, 2-3-4-drzewa, drzewa czerwono-czarne). Optymalne drzewa wyszukiwań binarnych. Haszowanie. Wyszukiwanie pozycyjne.
8. Wyszukiwanie zewnętrzne - B-drzewa.
9. Problem sumowania zbiorow rozłącznych i jego zastosowania.
10. Algorytmy grafowe: przepływy w sieciach, skojarzenia.
11. Algorytmy na tekstach. Wyszukiwanie wzorca. Drzewa sufiksowe.
12. Geometria obliczeniowa. Lokalizacja punktu. Otoczka wypukła. Technika zamiatania.
13. Algorytmy algebraiczne i teorioliczbowe. FFT. Szybkie mnożenie liczb i wielomianow.
14. NP-zupełność. Algorytmy aproksymacyjne dla problemow obliczeniowo trudnych. Heurystyki dla problemow trudnych (algorytmy genetyczne, simulated annealing).
15. Modele obliczeń rownoległych: PRAM, tablica procesorow, hiperkostka. Algorytmy rownoległe. Klasa NC i problemy P-zupełne.
16. Specjalne modele obliczeń: sieci komparatorow, obwody logiczne.
17. Algorytmy randomizacyjne. Przykłady w dziedzinach: struktury danych, geometria obliczeniowa, algorytmy grafowe, algorytmy rownoległe.

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe struktury danych oraz ich właściwości	K_W02, K_W08
zna i rozumie wybrane techniki i paradygmaty konstrukcji algorytmów	K_W02, K_W07, K_W14
posiada wiedzę na temat znanych rozwiązań dla typowych zagadnień algorytmicznych	K_W02, K_W08, K_W14
rozumie podstawowe pojęcia i techniki z zakresu analizy złożoności obliczeniowej algorytmów	K_W02, K_W07

Umiejętności

potrafi zastosować poznane struktury danych do konstrukcji algorytmów dla wybranych zadań obliczeniowych	K_U06, K_U19, K_U21
umie dowodzić poprawności konstruowanych algorytmów	K_U19, K_U21
umie analizować złożoność obliczeniową (czasową i pamięciową) badanych algorytmów	K_U06, K_U13, K_U19, K_U21, K_U22
potrafi pisemnie przedstawić argumenty dotyczące poprawności i złożoności obliczeniowej wybranych rozwiązań algorytmicznych	K_U06, K_U13, K_U19, K_U21, K_U22

Kompetencje społeczne

potrafi opowiadać o problemach algorytmicznych w sposób gwarantujący zrozumienie treści przez słuchaczy	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Wprowadzenie do algorytmów, PWN, 2012.
- Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman; Algorytmy i struktury danych; Helion.
- Dexter Kozen, The Design and Analysis of Algortihms, Springer 1990.
- Jon Kleinberg, Eva Tardos, Algorithm Design, Pearson 2006.
- Notatki do wykładu.

17.	Metody weryfikacji zakładanych efektów uczenia się prezentacja rozwiązania zadania, napisanie programu komputerowego, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Aby zaliczyć przedmiot, należy uzyskać zaliczenie pracowni i zaliczenie ćwiczeń oraz zdać egzamin. Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie pracowni, liczbę punktów za napisanie programów rozwiązujących zadane problemy algorytmiczne. Liczba punktów za poszczególne programy zależy od ich poprawności i efektywności. Zależność ta jest szczegółowo opisana w regulaminie pracowni. Do zaliczenia ćwiczeń należy zdobyć wymaganą, podaną w regulaminie ćwiczeń, liczbę punktów, uzyskanych za deklaracje rozwiązań oraz ich prezentacje. Egzamin ma formę pisemną i składa się z dwóch części. Podczas pierwszej części, trwającej 2,5 godziny, studenci otrzymują 20 pytań lub bardzo prostych zadań, mających na celu sprawdzenie stopnia opanowania wiedzy. W drugiej części, trwającej 4,5 godziny, studenci otrzymują sześć zadań, spośród których powinni wybrać i rozwiązać trzy dowolne. Te zadania sprawdzają stopień opanowania umiejętności określone w zakładanych efektach uczenia się. Do zdania egzaminu należy uzyskać wymagane, określone w regulaminie przedmiotu, liczby punktów za poszczególne części egzaminu.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** wykład: 60 godz. ćwiczenio-pracownia: 60 godz. **Praca własna studenta** studiowanie notatek i literatury: 15 godz. rozwiązywanie zadań z list: 60 godz. udział w egzaminie: 7 godz. przygotowanie do egzaminu: 35 godz. rozwiązywanie zadań programistycznych: 60 godz. **Sumarycznie** Łączna liczba godzin: 302 godz. Liczba punktów ECTS: 13

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Języki formalne i złożoność obliczeniowa** **Formal Languages and Computational Complexity**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DOJFZO**
6.	Rodzaj przedmiotu/modułu **Obowiązkowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) **trzeci**
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 45 godzin, ćwiczenia — 45 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Zrealizowane przedmioty: • Logika dla informatyków • Analiza matematyczna • Algebra • Matematyka dyskretna
13.	Cele kształcenia dla przedmiotu Przedmiot dotyczy jednego z głównych obszarów teorii informatyki. Oczekuje się, że osoba, która go zaliczy, będzie miała świadomość wszechobecności, w sytuacjach związanych z praktyką informatyczną/algorytmiczną, podstawowych omawianych pojęć i zagadnień (takich jak automat skończony, problem rozstrzygalności, problem złożoności obliczeniowej, złożoność wielomianowa i złożoność NP, redukcja), a także będzie się umiała w praktyce sprawnie tymi pojęciami posługiwać. Ta zdolność do praktycznego posługiwania się stanowi priorytet, dlatego program przedmiotu nie obejmuje wielu zaawansowanych i specjalistycznych zagadnień, skupiając się na możliwie głębokim rozumieniu zagadnień elementarnych. Ponadto forma, w jakiej prowadzi się ćwiczenia tablicowe (w ramach których studenci prezentują na tablicy przygotowane przez siebie wcześniej rozwiązania zadań), przyczynia się do rozwoju kompetencji komunikacyjnych.

14.	Treści programowe A1. Deterministyczny automat skończony. Języki regularne. Lemat o pompowaniu. Twierdzenie o indeksie. A2. Niedeterminizm. Niedeterministyczny automat skończony. Determinizacja automatu. A3. Wyrażenia regularne. Rownoważność automatow skończonych i wyrażeń regularnych. Aspekty algorytmiczne: rozstrzyganie rownoważności wyrażeń regularnych jest możliwe ale zagadkowo czasochłonne. A4. Uwagi o automatach na obiektach innych niż słowa skończone: automaty na drzewach skończonych i na słowach nieskończonych. A5. Gramatyki bezkontekstowe. Przykłady. A6. Postać Chomsky'ego, lemat o pompowaniu i jego konsekwencje. A7. Automaty ze stosem. Postać Greibach gramatyk bezkontekstowych. Rownoważnośc gramatyk i automatow ze stosem. A8. Własności zamkniętości klasy językow bezkontekstowych. Niemożliwość determinizacji. R1. Zbiory rekurencyjne i rekurencyjnie przeliczalne. Funkcje rekurencyjne - częściowe i całkowite. Numeracja funkcji rekurencyjnych. Nierozstrzygalność problemu stopu. R2. Pojęcie redukcji. Twierdzenie Rice'a. Uwagi o implikacjach tw. Rice'a dla możliwości automatycznej weryfikacji programow. R3. Maszyna Turinga. Teza Churcha. R4. Nierozstrzygalność problemu słow dla semiprocesow Thuego. R5. Nierozstrzygalność problemu słow dla procesow Thuego (problemu słow w połgrupie). R6. Nierozstrzygalność problemu odpowiedniości Posta, i przykłady nierozstrzygalnych problemow dotyczących gramatyk. R7. Nierozstrzygalność teorii pierwszego rzędu liczb naturalnych z dodawaniem i mnożeniem (ewentualnie: z dodawaniem, mnożeniem i potęgowaniem). Uwagi o niemożliwości zaksjomatyzowania arytmetyki. Uwagi o dziesiątym problemie Hilberta. R8. Nierozstrzygalność rachunku predykatow pierwszego rzędu. C1. Klasa PTIME i PSPACE. Redukcje wielomianowe. Wielomianowa rownoważność 3SAT i 3COL. C2. Niedeterminizm i klasa NP. Przykłady. Przykłady językow z klasy co-NP, oraz z przecięcia NP i co-NP. Charakteryzacja NP jako klasy językow bedących projekcjami językow z P. C3. NP zupełność. Twierdzenie Cook'a. Więcej przykładow problemow NP- zupełnych. Uwagi o SAT-solverach. C4. PSPACE. Tw. Savitcha. C5. Problemy PSPACE-zupełne: QBF i totalność wyrażeń regularnych. Wyjaśnienie zagadki z wykładu A3. C6. Funkcje jednostronne. Uwagi o teoriozłożonościowych założeniach kryptografii z kluczem publicznym. C7. Słaba wersja twierdzenia o hierarchii czasowej: rożność EXPTIME i PTIME. Uwagi o sposobach uogolnienia tej słabej wersji. C8. Problemy wymagające dowodliwie czasu wykładniczego. Pebble games. Totalność wyrażeń regularnych. C9. Inne niż PTIME formalizacje intuicji "łatwej obliczalności". Uwagi o klasie FPT. Zrandomizowane algorytmy testowania pierwszości. Uwagi o komputerach kwantowych. C10. Przykład problemu rozstrzygalnego ale nieelementarnego: totalność wyrażeń regularnych z dopełnieniem.

15.	Zakładane efekty uczenia się

Wiedza

zna kilkunaście podstawowych zagadnień teorii informatyki, wymienionych w punkcie „Treści programowe”	K_W01, K_W02, K_W09, K_W14

Umiejętności

potrafi przeprowadzić elementarne rozumowania dotyczące omawianych pojęć, w szczególności rozumowanie (najczęściej polegających na konstrukcji odpowiedniej redukcji) dotyczące dolnych granic złożoności obliczeniowej prostych problemów	K_U02, K_U13, K_U21, K_U22
ma pogłębienie umiejętności prowadzenia rozumowania w paradygmacie nauk matematyczncyh, w szczególności odróżniania rozumowania poprawnego od niepoprawnego	K_U02, K_U13, K_U21, K_U22

Kompetencje społeczne

posiada umiejętność publicznego prezentowania swojego rozumowania i obrony jego poprawności	K_K02, K_K06

16. Literatura obowiązkowa i zalecana

[1] Michael Sipser, Wprowadzenie do teorii obliczen, WNT Warszawa 2009. Literatura dodatkowa
[2] Sanjeev Arora, Boaz Barak, Computational Complexity: A Modern Approach, Cambridge University Press, 2009.

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin pisemny i zaliczenie ćwiczeń tablicowych

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Egzamin odbywa się w trzech częściach, odpowiadających trzem obszarom tematycznym wykładu. Progi punktowe na poszczególne stopnie określone są w regulaminie przedmiotu. Zaliczenie ćwiczeń wymaga przygotowywania rozwiązań zadań przewidzianych na każde kolejne ćwiczenia i prezentowania tych rozwiązań na tablicy. Szczegółowe zasady odwołują się do „medody deklaracji” i są opisane w regulaminie przedmiotu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	45 godz.
wykład	45 godz.

Praca własna studenta

udział w egzaminie	7 godz.
przygotowanie do ćwiczeń (rozwiązywanie zadań)	120 godz.
przygotowanie do egzaminu	15 godz.

Sumarycznie

Łączna liczba godzin	232 godz.
Liczba punktów ECTS	9

2 Przedmioty podstawowe do wyboru

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Bazy danych** **Introduction to databases**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6BD**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracowni – 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Niezbędne kompetencje:** • Podstawy logiki i teorii mnogości (pojęcie relacji i rachunku zbiorów, formuły z kwantyfikatorami). • Podstawy programowania (znajomość podstawowych cech systemów informatycznych). • Umiejętność programowania w wybranym języku w stopniu wystarczającym do napisania aplikacji do projektu bazodanowego.
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie słuchaczy z typowymi systemami baz danych, głównie z systemami relacyjnymi, oraz opanowanie kluczowych umiejętności związanych z użytkowaniem i tworzeniem baz danych oraz stosowaniem systemów bazodanowych.

14.	Treści programowe

Program wykładu:

1. Model relacyjny. Podstawowe pojęcia (relacja) oraz języki zapytań (algebra relacji, rachunki relacyjne).
2. Teoria relacyjnych baz danych (zależności funkcyjne, postaci normalne, denormalizacja).
3. SQL jako standard relacyjnego języka zapytań i opisu danych. Zapytania SQL, elementy relacyjnej bazy danych (tabele, dziedziny, perspektywy, wyzwalacze i asercje, funkcje, użytkownicy i uprawnienia).
4. SQL w aplikacjach bazodanowych: osadzony SQL, programistyczny SQL, PLSQL.
5. Modelowanie konceptualne baz danych (diagramy E-R, UML). 6. Tworzenie bazy danych w oparciu o model konceptualny.
6. Elementy systemów zarządzania bazami danych (transakcje, optymalizacja, bezpieczeństwo danych).
7. Inne modele baz danych (m.in. grafowe bazy danych)

Program ćwiczeń i pracowni:

1. Zajęcia konwersatoryjne (ćwiczenia) pomagające w praktycznym opanowaniu pojęć z teorii bazy danych, zasad działania systemów zarządzania bazami danych oraz projektowania relacyjnych baz danych.
2. Zajęcia laboratoryjne (pracownia) pozwalające zapoznać się z profesjonalnymi systemami zarządzania bazami danych (PostrgeSQL itp.) i opanować język SQL.
3. Wykonanie kompletnego projektu niewielkiej bazy danych wraz z aplikacją użytkownika.

15. Zakładane efekty uczenia się

Wiedza

zna pojęcia modelu relacyjnego oraz zasady dobrego modelowania relacyjnych baz danych	K_W14
jest świadomy istnienia innych modeli, ich zalet i wad w porównaniu do modelu relacyjnego	K_W14
zna język SQL w zakresie opisu bazy danych i wyszukiwania informacji w bazie danych	K_W15
zna zasady działania systemów zarządzania bazami danych w zakresie optymalizacji, wielodostępu i bezpieczeństwa	K_W15

Umiejętności

projektuje modele relacyjnej bazy danych i formułuje zapytania w algebrze relacji i rachunkach relacyjnych	K_U02, K_U17
definiuje bazę danych w SQL z użyciem dostępnych i właściwie dobranych elementów i struktur tego języka	K_U13
formułuje efektywne i poprawne zapytania SQL	K_U02
projektuje bazy danych dostosowane do potrzeb użytkownika i spełniające wymogi dobrego i efektywnego projektu	K_U16
tworzy bazy danych w oparciu o podany projekt	K_U17
tworzy aplikacje bazodanowe zgodnie z zadaną specyfikacją	K_U17
potrafi zastosować posiadaną wiedzę do realizacji efektywnych rozwiązań w bazach danych	K_U19
potrafi jasno i zrozumiale przedstawiać własne pomysły (np. rozwiązanie zadania, element projektu bazy danych, fragment kodu), a także prosić o wyjaśnienia w przypadku problemów ze zrozumieniem pomysłów i rozwiązań proponowanych przez innych.	K_U23

Kompetencje społeczne

Jest świadomy ograniczeń własnej wiedzy i umiejętności, adekwatnie ocenia poziom swoich kompetencji, zna swoje mocne i słabe strony, rozumie konieczność doskonalenia swoich zawodowych kompetencji	K_K01
Jest świadom roli i znaczenia wiedzy w rozwiązywaniu problemów o charakterze poznawczym i praktycznym	K_K03
Aktywnie prezentuje krytyczną postawę wobec stwierdzeń, uwag i wniosków	K_K02
Wykazuje samodzielność myślenia i działania przy rozwiązywaniu problemów i wykonywaniu zadań w zakresie projektowania i tworzenia baz danych	K_K06

16.	Literatura obowiązkowa i zalecana • Jeffrey D. Ullman, Jennifer Widom, Podstawowy Kurs Systemów Baz Danych, WNT, Warszawa 1999; • Garcia-Molina H., Ullman J.D., Widom J., Implementacja systemów baz danych, WNT, 2003 (seria: Klasyka Informatyki); • Garcia-Molina H., Ullman J.D., Widom J., Database Systems: The Complete Book (suma dwóch powyzszych pozycji); ˙ • Thomas Connolly, Carolyn Begg, Database Systems, Addison Wesley 2002, także˙po polsku: ReadMe 2004; Date C. J., An Introduction to Database Systems, vol. II, Adison-Wesley Pub. Comp., równiez WNT – W-wa, (seria: Klasyka Informatyki), 2000; ˙ • R. Ramakrishnan, J. Gehrke, Database Management Systems, McGraw-Hill Education, 2003 • dokumentacja systemu PostgrSQL (http://www.postgresql.org/docs/)
17.	Metody weryfikacji zakładanych efektów uczenia się egzamin, prezentacja projektu, praktyczne sprawdziany SQL
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Zaliczenie ćwiczeń i pracowni na podstawie liczby punktów zdobytych za poniższe aktywności: • prezentacja i dyskusja rozwiązań zadań z listy opublikowanej przed zajęciami; • sprawdziany pisemne ze znajomości modelu relacyjnego i relacyjnych języków zapytań; • sprawdziany praktyczne ze znajomości języka SQL (tworzenie zapytań pod kontrolą nauczyciela w oparciu o udostępnioną dokumentację); • zaliczenie projektu bazy danych oraz prezentacja aplikacji; • egzamin pisemny ze znajomości modelu relacyjnego oraz podstaw działania systemów zarządzania bazami danych;
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** ćwiczenio-pracownia: 30 godz. wykład: 30 godz. **Praca własna studenta** przygotowanie raportu/prezentacji: 2 godz. przygotowanie do sprawdzianów/kolokwiów: 25 godz. przygotowanie do ćwiczeń: 10 godz. studiowanie tematyki wykładów i literatury: 20 godz. przygotowanie do pracowni: 20 godz. praca nad projektem: 30 godz. przygotowanie do egzaminu: 5 godz. **Sumarycznie** Łączna liczba godzin: 172 godz. Liczba punktów ECTS: 6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Podstawy grafiki komputerowej** **Fundamentals of Computer Graphics**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6PGK**
6.	Rodzaj przedmiotu/modułu **Informatyczny inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracowni – 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Niezbędne kompetencje:** • Znajomość algebry liniowej, analizy numerycznej, podstaw algorytmów i struktur danych. • Umiejętność dobrego programowania w C/C++.
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest przedstawienie podstawowych pojęć i algorytmów grafiki komputerowej oraz nabycie przez studentów umiejętności programowania kart graficznych. Celem pracowni jest nauka praktyki programowania w nowym OpenGL.
14.	Treści programowe • Programowanie kart graficznych przy pomocy API OpenGL • Przekształcenia w jednorodnym układzie współrzędnych • Efektywne reprezentacje rotacji w tym kwaterniony • Modelowanie wirtualnej kamery i obiektów 3D • Widoczność: algorytm z-bufora, śledzenie promieni i inne • Modelowanie i obliczanie oświetlenia • Rasteryzacja i antialiasing • Światło i kolor w grafice, modele koloru • Teksturowanie 2D/3D, tekstury proceduralne

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe pojęcia i algorytmy grafiki komputerowej	K_W14
rozumie ograniczenia i podstawy działania GPU	K_W15

Umiejętności

umie konstruować macierze w jednorodnym układzie współrzędnych dla przekształceń pomiędzy zadanymi układami	K_U14
potrafi programować karty graficzne przy pomocy API OpenGL	K_U16, K_U19
umie programować shadery w języku GLSL	K_U16, K_U17

Kompetencje społeczne

rozumie znaczenie grafiki komputerowej we współczesnym społeczeństwie	K_K04

16. Literatura obowiązkowa i zalecana

- P. Shirley - "Fundamentals of Computer Graphics", A.K.Peters, Natick Massachusetts 2002.
- J.F.Hughes, A.van Dam, M.Mcguire, D.F.Sklar, J.D.Foley, S.K.Feiner, K.Akeley - "Computer Graphics - Principles and Practice", 3rd Edition 2014.

17. Metody weryfikacji zakładanych efektów uczenia się

- napisanie programu komputerowego
- prezentacja rozwiązania zadania
- egzamin

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu, liczbę punktów za zadania/projekty programistyczne.
Aby zaliczyć egzamin konieczne jest zdobycie wymaganej liczby punktów.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	10 godz.
studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do pracowni	20 godz.
egzamin	2 godz.
praca nad projektami programistycznymi	30 godz.

Sumarycznie

Łączna liczba godzin	152 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Programowanie funkcyjne** **Functional Programming**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6PF**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracowni – 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Metody programowania **Niezbędne kompetencje:** • Umiejętność programowania w języku wyższego poziomu • Pożądana znajomość języka Scheme
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie studentów z mechanizmami i metodami programowania funkcyjnego w językach z typizacją statyczną (OCaml, Haskell). Paradygmat ten obecnie staje się bardzo popularny (również w językach imperatywnych), ponieważ wysoki poziom abstrakcji ułatwia składanie komponentów, a bezstanowość jest wykorzystywana w wysokopoziomowych mechanizmach programowania współbieżnego, które pozwalają pisać efektywne programy, wykorzystujące procesory wielordzeniowe.
14.	Treści programowe 1. System typów z polimorfizmem parametrycznym 2. Funkcje wyższych rzędów 3. Algebraiczne typy danych i mechanizm dopasowania do wzorca 4. Ewaluacja gorliwa i leniwa, rekursja ogonowa 5. Efekty obliczeniowe w języku funkcyjnym 6. Abstrakcyjne typy danych, moduły 7. Klasy typów jako mechanizm abstrakcji, monady

15.	Zakładane efekty uczenia się

Wiedza

rozumie specyfikę programowania funkcyjnego	K_W14
zna używane w językach funkcyjnych mechanizmy abstrakcji	K_W14
zna typowe dla programowania funkcyjnego mechanizmy językowe	K_W14

Umiejętności

potrafi wykorzystać mechanizmy paradygmatu funkcyjnego w implementacji programów	K_U03, K_U16

16. Literatura obowiązkowa i zalecana

- materiały do wykładu, udostępniane przez prowadzącego
- Minsky Y., Madhavapeddy A., Hickey J., Real World OCaml, https://v1.realworldocaml.org/
- E.Chailloux, P.Manoury, B.Pagano. Developing Applications with Objective Caml. http://caml.inria.fr/pub/docs/oreilly-book/
- O'Sullivan B., Goerze J., Stewart D., Real World Haskell, http://book.realworldhaskell.org/

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin pisemny, napisanie programu komputerowego, realizacja zadań przy komputerze

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia pracowni konieczne jest zaimplementowanie i wyjaśnienie wymaganych programów, zgodnie z podanymi specyfikacjami (ocenianych punktowo lub w równoważny sposób).
Egzamin ma formę pisemną, aby go zaliczyć konieczne jest zdobycie wymaganej liczby punktów.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	20 godz.
przygotowanie do pracowni	40 godz.
studiowanie tematyki wykładów i literatury	30 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Sieci komputerowe** **Computer networks**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6SK**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykładu – 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Algebra lub Matematyka dyskretna • Systemy operacyjne lub Systemy komputerowe **Niezbędne kompetencje:** • Znajomość grup i pierścieni. • Znajomość tematyki procesów, sygnałów i komunikacji międzyprocesowej. • Obsługa wiersza poleceń w systemie Linux, umiejętność pisania prostych skyptów w powłoce. • Dobra umiejętność programowania w C lub C++.
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zaznajomienie słuchaczy z mechanizmami, protokołami i algorytmicznymi podstawami efektywnej i bezpiecznej komunikacji w sieciach komputerowych, ze szczególnym uwzględnieniem Internetu.

14.	Treści programowe 1. Podstawowe pojęcia, protokoły, model warstwowy. 2. Warstwa sieciowa: adresowanie CIDR, routing, protokoły IP i ICMP, tworzenie tablic routingu oparte o stan łączy i wektory odległości, systemy autonomiczne, budowa routera, NAT. 3. Warstwa łącza danych i fizyczna: Ethernet, sieci bezprzewodowe, dostęp do współdzielonego nośnika (CSMA/CD, CSMA/CA). 4. Warstwa transportowa: niezawodny transport, mechanizmy okna przesuwnego, kontrola przepływu, kontrola przeciążenia, protokoły UDP i TCP. 5. Warstwa zastosowań: DNS, HTTP, poczta elektroniczna, sieci P2P, serwery proxy, współpraca z NAT. 6. Programowanie gniazd: gniazda surowe, datagramowe i strumieniowe. 7. Podstawy kodowania i kryptografii: kody detekcyjne (CRC) i korekcyjne, szyfrowanie, podpisy cyfrowe, RSA, SSL, PGP. 8. Podstawy bezpieczeństwa sieci.
15.	Zakładane efekty uczenia się **Wiedza** Zna podstawowe mechanizmy, protokoły i algorytmiczne podstawy sieci komputerowych: model warstwowy, routing, model klient-serwer, niezawodny transport, mechanizmy okna przesuwnego, kontrola przepływu, kontrola przeciążenia, protokoły dostępu do współdzielonego medium. — K_W14 Zna implementacje powyższych mechanizmów we współczesnym Internecie: protokoły TCP, UDP, IP, Ethernet. — K_W15 Zna elementy kryptografii, podstawowe słabości stosowanych protokołów i typy ataków sieciowych. — K_W14, K_W15 **Umiejętności** Konfiguruje działanie prostej sieci lokalnej złożonej z komputerów, przełączników i routerów. — K_U18 Jest w stanie zdiagnozować i rozwiązać problemy z brakiem połączenia, połączeniem złej jakości czy też wadliwą konfiguracją routingu. — K_U18 Posługuje się standardowymi narzędziami szyfrowania danych, potrafi podsłuchiwać nieszyfrowany ruch sieciowy. — K_U18, K_U19 Pisze programy komunikujące się przez sieć wykorzystując interfejs gniazd. — K_U16 **Kompetencje społeczne** Rozumie znaczenie Internetu w funkcjonowaniu współczesnego społeczeństwa, rozumie możliwości i zagrożenia z tym związane. — K_K02
16.	Literatura obowiązkowa i zalecana • James F. Kurose, Keith W. Ross, Sieci komputerowe. Ujęcie całościowe, Helion, 2018. • Andrew S. Tanenbaum, Sieci komputerowe, Helion, 2012. • W. Richard Stevens, Unix programowanie usług sieciowych, tom 1, WNT, 2002, rozdziały 1-9 i 25.
17.	Metody weryfikacji zakładanych efektów uczenia się prezentacja rozwiązania zadania, napisanie programu komputerowego, realizacja zadań przy komputerze, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Warunkiem zaliczenia ćwiczenio-pracowni jest uzyskanie minimum 40 punktów łącznie za pisemne przygotowanie zadań domowych, aktywną realizacje zadań warsztatowych przy komputerze i za projekty programistyczne. Co najmniej 10 punktów należy uzyskać za realizację projektów programistycznych. Warunkiem zaliczenia wykładu jest zdobycie co najmniej 60 punktów na egzaminie pisemnym. Szczegółowe warunki zaliczenia przedmiotu określone są w regulaminie przedmiotu.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

rozwiązywanie zadań domowych	15 godz.
studiowanie tematyki wykładu oraz przygotowanie do egzaminu	20 godz.
pisanie projektów programistycznych	40 godz.
udział w egzaminie	3 godz.
przygotowywanie się do zajęć warsztatowych	15 godz.

Sumarycznie

Łączna liczba godzin	153 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Systemy operacyjne** **Operating Systems**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DI1SO**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin, pracownia — 15 godzin, repetytorium — 15 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Kurs: Podstawowy warsztat informatyka • Kurs: Wstęp do programowania w języku C • Architektury systemów komputerowych **Niezbędne kompetencje** • Umiejętność obsługi wiersza poleceń systemu uniksowego • Dobra znajomość języka C • Podstawowa wiedza o przerwaniach i wyjątkach procesora, tłumaczeniu adresów i pamięci wirtualna oraz konsolidacji
13.	Cele kształcenia dla przedmiotu Celem wykładu jest zapoznanie słuchaczy z wybranymi aspektami systemu uniksowego z punktu widzenia programisty systemowego i administratora. Na przykładzie wywołań systemowych, biblioteki języka C i narzędzi standardu POSIX.1 zostaną przedstawione najważniejsze osiągnięcia w dziedzinie systemów operacyjnych.

14.	Treści programowe 1. Jądro, wywołania systemowe, zmiana kontekstu, wywłaszczanie. 2. Procesy: stan i środowisko wykonania, tworzenie i kończenie, oczekiwanie na zakończenie, ładowanie programów i hierarchia. 3. Sygnały i nielokalne skoki. 4. Zarządzanie zadaniami, sesje, terminal, organizacja powłoki uniksowej. 5. Pliki: rodzaje, operacje, ścieżki, pliki urządzeń, deskryptory plików. 6. System plików: struktury dyskowe. 7. Buforowanie plików. 8. Tożsamość, autoryzacja i uwierzytelnianie. 9. Komunikacja międzyprocesowa. 10. Pamięć wirtualna, odwzorowania plików w pamięć. 11. Zarządzanie stertą: algorytmy przydziału pamięci. 12. Programowanie sieciowe TCP/IP, architektura klient-serwer. 13. Wielowątkowość i problemy z współbieżnością. 14. Synchronizacja: muteksy i zmienne warunkowe.
15.	Zakładane efekty uczenia się

Wiedza

Posiada przeglądową wiedzę na temat komponentów systemu operacyjnego.	K_W14
Zna mechanizmy, polityki i algorytmy, które wykorzystują jądra SO w celu realizacji usług dla programów użytkownika.	K_W14, K_W15
Rozumie wpływ architektury systemu komputerowego na implementację SO.	K_W15

Umiejętności

Potrafi użyć środków synchronizacji do konstrukcji poprawnych programów współbieżnych.	K_U19
Umie wykorzystać abstrakcje oferowane przez jądro SO przy projektowaniu oprogramowania.	K_U19
Potrafi czytać i analizować kod jądra SO z użyciem wyspecjalizowanych narzędzi.	K_U19
Umie śledzić interakcję programów użytkownika z jądrem SO w celu znajdowania usterek i problemów z wydajnością.	K_U18
Potrafi zaproponować bezpieczną i wydajną metodę przechowywania danych z użyciem usług SO.	K_U17
W projektowaniu oprogramowania posługuje się zasadą separacji mechanizmu od polityki.	K_U17

Kompetencje społeczne

Potrafi ocenić przydatność danego SO do implementacji projektu w pracy zawodowej.	K_K06
Pomaga członkom zespołu wybrać właściwe interfejsy i usługi SO do realizacji zadania.	K_K06

16.	Literatura obowiązkowa i zalecana • Operating Systems: Three Easy Pieces Remzi Andrea Arpaci-Dusseau • Advanced Programming in the UNIX Environment (3rd Edition) Richard Stevens, Stephen Argo
17.	Metody weryfikacji zakładanych efektów uczenia się egzamin pisemny, projekt programistyczny, prezentacja rozwiązania zadania
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia ćwiczeń i pracowni należy zdobyć co najmniej połowę liczby punktów za zadania ćwiczeniowe i programistyczne. Na pracowni będą co najmniej trzy projekty programistyczne do wykonania: powłoka uniksowa, algorytm przydziału pamięci, system plików. Na punkty z poszczególnych aktywności mogą być narzucone minimalne progi zaliczeniowe. Egzamin końcowy jest w formie pisemnej i wymaga zdobycia co najmniej połowy punktów. Zadania mogą mieć następujące formy: opisanie problemu, prezentacja działania algorytmu, test wielokrotnego wyboru.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

repetytorium	15 godz.
pracownia	15 godz.
ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

udział w egzaminie	3 godz.
przygotowanie do egzaminu	10 godz.
rozwiązywanie zadań z list	80 godz.
rozwiązywanie zadań programistycznych	60 godz.

Sumarycznie

Łączna liczba godzin	243 godz.
Liczba punktów ECTS	8

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Sztuczna inteligencja** **Artificial Intelligence**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6SI**
6.	Rodzaj przedmiotu/modułu **Informatyczny inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia – 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Logika dla informatyków • Wstęp do informatyki lub Algorytmy i struktury danych • Analiza matematyczna **Niezbędne kompetencje:** • Umiejętność programowania w języku wyższego poziomu • Pożądana wstępna znajomość języka Python
13.	Cele kształcenia dla przedmiotu Podstawowym celem przedmiotu jest zapoznanie studentów z technikami stosowanymi do rozwiązywania problemów, które są z jednej strony trudne do rozwiązania przy użyciu standardowych technik algorytmicznych, a z drugiej są efektywnie rozwiązywane przez ludzi, korzystających ze swojej inteligencji. Zajęcia koncentrują się na następujących pojęciach: modelowanie świata, przeszukiwanie przestrzeni rozwiązań, wnioskowanie i uczenie się z przykładów bądź z symulacji.

14.	Treści programowe

1. Modelowanie rzeczywistości za pomocą przestrzeni stanów.
2. Przeszukiwanie w przestrzeni stanów: przeszukiwanie wszerz i w głąb, iteracyjne pogłębianie, przeszukiwanie dwustronne, algorytm A*, właściwości i tworzenie funkcji heurystycznych wspomagających przeszukiwanie.
3. Przeszukiwanie metaheurystyczne: hill climbing, symulowane wyżarzanie, beam search, algorytmy ewolucyjne.
4. Rozwiązywanie więzów: modelowanie za pomocą więzów, spójność więzów i algorytm AC-3, łączenie propagacji więzów z przeszukiwaniem z nawrotami, specjalistyczne języki programowania z więzami na przykładzie Prologa z więzami.
5. Strategie w grach: gry dwuosobowe, algorytm minimax, odcięcia alfa-beta, przykłady heurystycznej oceny sytuacji na planszy w wybranych grach, losowość w grach, algorytm Monte Carlo Tree Search.
6. Elementy teorii gier: strategie czyste i mieszane, rozwiązywanie prostych gier typu dylemat więźnia.
7. Modelowanie za pomocą logiki zdaniowej, wnioskowanie forward-chaining i backward-chaining, rozwiązywanie problemów spełnialności formuły w CNF (WalkSAT, DPLL).
8. Modelowanie niepewności świata: sieci bayesowskie, procesy decyzyjne Markowa, algorytmy value iteration oraz policy iteration. Elementy uczenia ze wzmocnieniem: TD-learning oraz Q-learning.
9. Podstawy uczenia maszynowego: idea uczenia się z przykładów, generalizacja, niebezpieczeństwo przeuczenia. Wybrane metody: regresja liniowa i logistyczna, wielowarstwowe sieci neuronowe (MLP), algorytm k-NN, drzewa decyzyjne i lasy losowe.

15. Zakładane efekty uczenia się

Wiedza

rozumie, czym zajmuje się Sztuczna inteligencja, rozumie również, na czym polega specyficzność metod tej dziedziny	K_W16
posiada przeglądową wiedzę o różnych dziedzinach sztucznej inteligencji	K_W14
zna różne metody modelowania świata, z uwzględnieniem niepewności	K_W14
zna algorytmy przeszukiwania przestrzeni stanów oraz przeszukiwania drzew gry	K_W14
zna podstawowe algorytmy wnioskowania	K_W14
zna podstawowe metody uczenia maszynowego (z nadzorem oraz ze wzmocnieniem)	K_W14

Umiejętności

umie modelować różne zagadnienia jako zadania przeszukiwania (lub przeszukiwania z więzami)	K_U18
umie stosować i modyfikować różne algorytmy przeszukiwania (w tym również przeszukiwania w grach)	K_U19
umie modelować niepewność świata za pomocą narzędzi z rachunku prawdopodobieństwa (ze szczególnym uwzględnieniem metod Monte Carlo)	K_U19
umie stosować podstawowe metody uczenia maszynowego (w tym również metody uczenia ze wzmocnieniem)	K_U19

Kompetencje społeczne

rozumie znaczenie algorytmów sztucznej inteligencji dla funkcjonowania współczesnego społeczeństwa, rozumie możliwości i niebezpieczeństwa z tym związane	K_K02
umie prezentować swoje idee w sposób dostosowany do wiedzy słuchaczy	K_K05

16. Literatura obowiązkowa i zalecana

- Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach.
- Richard S. Sutton and Andrew G. Barto, Reinforcement Learning: An Introduction.
- Prateek Joshi, Artificial Intelligence with Python.

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin pisemny, prezentacja projektu, prezentacja rozwiązania zadania, napisanie i prezentacja programu komputerowego

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia ćwiczenio-pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za zadania ćwiczeniowe, pracowniowe i opcjonalny projekt. Punkty za wszystkie wyżej wymienione aktywności liczą się łącznie. Egzamin ma formę pisemną, aby go zaliczyć konieczne jest zdobycie wymaganej liczby punktów.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

udział w egzaminie	3 godz.
samodzielne rozwiązywanie zadań pracowniowych i projektowych	50 godz.
studiowanie tematyki wykładów i literatury oraz przygotowanie do egzaminu	15 godz.
przygotowywanie się do ćwiczeń (w tym czytanie materiałów dodatkowych)	25 godz.

Sumarycznie

Łączna liczba godzin	153 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Wybrane elementy praktyki projektowania oprogramowania** **Selected Aspects of Practical Software Development**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DI1WEPPO**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracowni – 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu
13.	Cele kształcenia dla przedmiotu Celem zajęć jest przedstawienie studentom praktyki projektowania oprogramowania. Zajęcia w zwięzły sposób łączą podstawową wiedzę z zakresu baz danych i inżynierii oprogramowania w obszarze projektowania obiektowego w wybranych współczesnych realiach technologicznych.
14.	Treści programowe 1. podstawy projektowania obiektowego 2. język UML 3. język Javascript - programowanie modularne i obiektowe 4. język Typescript 5. technologia node.js 6. frameworki Express i Socket.IO 7. podstawy języka SQL i komunikacji z relacyjnymi bazami danych 8. testowanie aplikacji webowych

15.	Zakładane efekty uczenia się

Wiedza

zna podstawy projektowania obiektowego	K_W15
zna podstawy technologii baz danych	K_W15
zna wybrany przemysłowy język wytwarzania aplikacji	K_W15
zna możliwości wybranej przemysłowej technologii wytwarzania aplikacji (w tym aplikacji internetowych)	K_W15

Umiejętności

potrafi zaprojektować strukturę relacyjnej bazy danych w wybranej technologii	K_U19
potrafi zaprojektować, zaimplementować i wdrożyć aplikację internetową wykorzystującą typowe elementy technologiczne	K_U16, K_U17
potrafi projektować diagramy UML z wykorzystaniem wybranego narzędzia przemysłowego	K_U19

16. Literatura obowiązkowa i zalecana

- D. Flanagan - JavaScript: The Definitive Guide, 6th Edition
- S. Stefanov - Javascript Patterns
- E. Hahn - Express in Action
- M. Fogus - Functional Javascript
- A. Beaulieu - Learning SQL

17. Metody weryfikacji zakładanych efektów uczenia się

- napisanie i prezentacja programu komputerowego
- prezentacje rozwiązania zadania
- prezentacja projektu programistycznego
- egzamin pisemny

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Wykład, pracownia programistyczna - listy zadań pokrywające poszczególne partie materiału. Student zdobywa minimalną liczbę punktów określoną w regulaminie zajęć.
Egzamin pisemny sprawdzający wiedzę w poszczególnych partiach materiału.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	20 godz.
przygotowanie do pracowni	40 godz.
studiowanie tematyki wykładów i literatury	20 godz.
udział w egzaminie	2 godz.

Sumarycznie

Łączna liczba godzin	142 godz.
Liczba punktów ECTS	6

3 Przedmioty informatyczne

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Advanced Data Mining** **Advanced Data Mining**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-SAdvDatMin**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – zastosowania informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Niezbędne kompetencje:** • znajomość podstaw algebry liniowej • znajomość podstaw rachunku prawdopodobieństwa i statystyki • znajomość podstaw eksploracji danych lub determinacja do samodzielnego ich poznania w pierwszych tygodniach semestru
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie studentów z zaawansowanymi algorytmami eksploracji danych, dotyczącymi głównie systemów rekomendujących, redukcji wymiarowości, eksploracji danych temporalnych i systemów wspomagania decyzji.
14.	Treści programowe Wybrane algorytmy eksploracji danych dla: • systemów rekomendujących, zwłaszcza oparte na faktoryzacji macierzy, • redukcji wymiarowości, zwłaszcza oparte na zanurzeniach sąsiedztw punktów, • przetwarzania danych temporalnych, • systemów wspomagania decyzji.

15. Zakładane efekty uczenia się

Wiedza

rozumie wybrane zaawansowane podejścia eksploracji danych	K_W14
rozumie problemy związane z eksploracją złożonych danych, w tym nieustrukturalizowanych, uwzględniając także ich zmienność w czasie	K_W15
posiada przeglądową wiedzę o różnych podejściach zaawansowanej eksploracji danych	K_W14
zna wybrane algorytmy grupowania i klasyfikacji złożonych danych, w tym nieustrukturalizowanych, uwzględniając także ich zmienność w czasie	K_W07
zna wybrane algorytmy uczenia maszynowego i głębokiego do eksploracji złożonych danych	K_W14
rozumie znaczenie algorytmów zaawansowanej eksploracji danych dla funkcjonowania współczesnego społeczeństwa	K_W16

Umiejętności

umie formułować zagadnienia praktyczne jako zadania zaawansowanej eksploracji danych	K_U18
umie stosować i modyfikować algorytmy zaawansowanej eksploracji danych	K_U18, K_U19
umie stosować metody uczenia maszynowego i głębokiego do eksploracji złożonych danych	K_U18, K_U19

Kompetencje społeczne

umie prezentować swoje idee w sposób dostosowany do wiedzy słuchaczy	K_K05

16. Literatura obowiązkowa i zalecana

wybrane artykuły naukowe z czasopism i konferencji z dziedziny eksploracji danych, inteligencji obliczeniowej i uczenia maszynowego

17. Metody weryfikacji zakładanych efektów uczenia się

prezentacja, dyskusja

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Zaliczenie ćwiczenio-pracowni wymaga zdobycia określonej w regulaminie przedmiotu liczby punktów za zadania, projekt i ewentualne dodatkowe aktywności (minireferaty). Zaliczenie wykładu wymaga zdobycia określonej liczby punktów na egzaminie.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do prezentacji projektu i egzaminu	10 godz.
przygotowywanie się do zajęć (w tym czytanie materiałów dodatkowych)	20 godz.
rozwiązywanie zadań i realizacja projektu	60 godz.
udział w egzaminie: y	2 godz.

Sumarycznie

Łączna liczba godzin	152 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Advanced Distributed Algorithms** **Advanced Distributed Algorithms**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-AdvDstAlg**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Na wykładzie przyjmuje się, że studenci znają materiał z przedmiotu Algorytmy i Struktury Danych, posiadają też podstawową wiedzę z rachunku prawdopodobieństwa.
13.	Cele kształcenia dla przedmiotu Celem wykładu jest zapoznanie studentów z najpopularniejszymi obecnie modelami obliczeń rozproszonych oraz problemami algorytmicznymi w nich występującymi. W szczególności wykład skoncentrowany będzie na rozwiązywaniu problemów grafowych w środowisku rozproszonym, a także na problemach algorytmicznych związanych komunikacją i koordynacją współpracujących jednostek, odpornościa na błędy, współdzieleniem łączy komunikacyjnych i innych zasobów. Rozważane będą klasyczne modele obliczeń rozproszonych (Local, Congest, pamięć dzielona) jak i nowsze (Congested Clique, radio/wireless networks, MapReduce).
14.	Treści programowe Wykład obejmować będzie m.in. następujące problemy w różnych modelach obliczeń rozproszonych: • Rozproszone kolorowanie grafu (sieci) • Wyznaczenie składowych spójności grafu, (minimalnych) drzew spinających, BFS • Wyznaczanie najkrótszych ścieżek, problemy rutingu • Wyznaczanie maksymalnego zbioru niezależnego, minimalnego zbioru dominującego Wyznaczanie minimalnych/maksymalnych cięć w grafie • Klastrowanie i dekompozycja sieci Rozgłaszanie (broadcasting) i zbieranie (gathering) informacji, elekcja lidera, łamanie symetrii w sieciach (ad hoc). • Problemy algorytmiczne w środowisku z pamięcią dzieloną.

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe modele obliczeń rozproszonych, ich zalety i wyzwania związane z korzystaniem z nich	K_W02
zna podstawowe miary jakości algorytmów rozproszonych	K_W14, K_W15
zna najważniejsze problemy algorytmiczne występujące w środowisku rozproszonym	K_W14
posiada podstawy umożliwiające śledzenie i rozumienie najnowyszych osiągnięć w obszarze algortymów rozproszonych	K_W14

Umiejętności

potrafi konstruować algorytmy rozproszone spełniające zadane warunki	K_U18
potrafi analizować algorytmy rozproszone stosując standardowe uwzględniając różne miary ich jakości	K_U18
potrafi korzystać z literatury specjalistycznej w dziedzinie	K_U20, K_U22

Kompetencje społeczne

potrafi dzielić się zdobytą wiedzą specjalistyczną z innymi, prezentując ją w formie przystępnej nie tylko dla wąskiej grupy specjalistów	K_K06

16. Literatura obowiązkowa i zalecana

- David Peleg. Distributed Computing: A Locality-Sensitive Approach Society for Industrial and Applied Mathematics (SIAM), 2000, ISBN 0-89871-464-8 Hagit
- Attiya, Jennifer Welch. Distributed Computing: Fundamentals, Simulations and Advanced Topics McGraw-Hill Publishing, 1998, ISBN 0-07-709352 6 Juraj
- Hromkovic, Ralf Klasing, Andrzej Pelc, Peter Ruzicka, Walter Unger.
- Dissemination of Information in Communication Networks Springer-Verlag, Berlin Heidelberg, 2005, ISBN 3-540-00846-2
- Skrypty do wymienionych w opisie przedmiotu wykładów na różnych uczelniach.

17. Metody weryfikacji zakładanych efektów uczenia się

Egzamin w formie ustnej.
Ćwiczenia:

- prezentacja samodzielnie przygotwanych rozwiązań zadań/problemów podawanych przez wykładowcę po wykładach;
- przedstawienie w formie prezentacji (wskazane przygotowanie materiałów multimedialnych/slajdów) aktualnych wyników naukowych dostępnych w postaci specjalistycznych publikacji naukowych.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia ćwiczeń należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów zdobywanych za: **przygotowanie oraz za prezentacje zadań z list zadań podanych przez wykładowcę** przedstawienie w formie prezentacji wybranych rezultatów badań naukowych dostępnych w formie publikacji naukowej.
Egzamin ma formę ustną, aby go zaliczyć konieczne jest zdobycie wymaganej liczby punktów, przyznawanej przez egzaminatora, odzwierciedlającej stopień znajomości zagadnień wylosowanych do przedstawienia, wyjaśnienia i/lub umiejętności rozwiązania wskazanych przez egzaminatora problemów algorytmicznych.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do egzaminu	30 godz.
przygotowanie do ćwiczeń	30 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Algorytmy online** **Online algorithms**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DZ7AO**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Algorytmy i struktury danych (M) • Rachunek prawdopodobieństwa **Niezbędne kompetencje:** • Umiejętność konstrukcji algorytmów, analizy ich poprawności i złożoności. • Znajomość analizy zamortyzowanej. • Swobodne posługiwanie się pojęciem zmiennej losowej. • Umiejętność konstrukcji algorytmów randomizowanych i analizy ich oczekiwanej złożoności.
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zaznajomienie słuchaczy ze podstawami problemów online, współczesnymi technikami konstrukcji algorytmów online i analizy konkurencyjnej.
14.	Treści programowe • Definicja konkurencyjności, przykładowe zagadnienia. • Algorytmy zaznaczające: problem pamięci podręcznej, wpływ randomizacji, metryczne systemy zadań. • Analiza zamortyzowana za pomocą funkcji potencjału: problem reorganizacji listy. • Funkcje potencjału dla algorytmów randomizowanych. • Zasada minimaksowa dla dowodzenia dolnych ograniczeń. • Metoda podwajania: zastosowania do problemów eksploracji terenu i szeregowania. • Techniki programowania liniowego: problem pokrywania zbiorami, routing. • Problem k-server: dolne ograniczenia, algorytm dla drzew. • Technika factor-revealing LP: problem migracji pliku. • Funkcje pracy: optymalny algorytm dla metrycznych systemów zadań. • Problem routingu i rozłącznych ścieżek. • Przybliżanie dowolnych metryk za pomocą drzew, zastosowania.

15. Zakładane efekty uczenia się

Wiedza

Zna definicję konkurencyjności i jej podstawowe własności.	K_W07, K_W14
Rozumie wpływ randomizacji na możliwość konstruowania lepszych algorytmów online.	K_W07,K_W13, K_W14
Zna zasadę minimaksową i jej znaczenie dla randomizowanej konkurencyjności.	K_W07, K_W14
Zna analizę zamortyzowaną w odniesieniu do algorytmów online.	K_W07, K_W14
Zna wybrane techniki analizy algorytmów online (zasadę zaznaczania, funkcje pracy, metodę podwajania, techniki programowania liniowego) zastosowane do klasycznych i współczesnych problemów online (problem pamięci podręcznej, reorganizacja listy, szeregowanie zadań, pokrywanie zbiorami).	K_W07,K_W13, K_W14

Umiejętności

Potrafi modelować konkretne zagadnienia jako problemy online.	K_U03
Potrafi konstruować algorytmy online wykorzystując wybrane techniki: zasadę zaznaczania, funkcje pracy, metodę podwajania, techniki programowania liniowego.	K_U13
Potrafi analizować jakość algorytmu online wykorzystując wybrane techniki: analizę zamortyzowaną, metody programowania liniowego.	K_U06
Potrafi dowodzić dolnych ograniczeń na konkurencyjność, w tym konkurencyjność randomizowaną wykorzystując zasadę minimaksową.	K_U06

Kompetencje społeczne

Potrafi krytycznie analizować prezentowane przez innych rozumowania.	K_K06
Zna wartość i ograniczenia analizy konkurencyjnej w odniesieniu do praktycznych zagadnień.	K_K02

16. Literatura obowiązkowa i zalecana

- Alan Borodin, Ran El-Yaniv: "Online Computation and Competitive Analysis". Cambridge University Press, 1998.
- Rajeev Motwani, Prabhakar Raghavan: "Randomized Algorithms". Cambridge University Press, 1995.
- Amos Fiat, Gerhard J. Woeginger: "Online Algorithms, The State of the Art". Springer, 1998
- Niv Buchbinder, Joseph Naor: "The Design of Competitive Online Algorithms via a Primal–Dual Approach". Foundations and Trends in Theoretical Computer Science, 2007

17. Metody weryfikacji zakładanych efektów uczenia się

- pisemne rozwiązywanie zadań domowych
- prezentacja rozwiązania zadania
- egzamin ustny

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- Egzamin sprawdzający znajomość algorytmów, modeli i technik dowodowych.
- Listy zadań (których rozwiązania prezentowane są przez słuchaczy na zajęciach) dotyczące wprowadzanych zagadnień, technik konstrukcji algorytmów, technik dowodzenia poprawności i efektywności algorytmów.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	30 godz.
udział w egzaminie	2 godz.
studiowanie tematyki wykładów i literatury	15 godz.
przygotowanie do ćwiczeń, w tym pisemne rozwiązywanie zadań	45 godz.

Sumarycznie

Łączna liczba godzin	152 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Algorytmy tekstowe** **Algorithms on strings**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DZ7AT**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – zastosowania informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Znajomość podstawowych technik konstruowania oraz analizy algorytmów i struktur danych, w tym: programowania dynamicznego, dziel-i-zwyciężaj, analizy zamortyzowanej. Umiejętność programowania w dowolnym języku wysokiego poziomu takim jak C++, Java.
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie studentów z narzędziami przydatnymi w konstruowaniu efektywnych algorytmów operujących na tekstach takimi jak kombinatoryczne własności typu okresowość, własności grafu odległości edycyjnej czy też grafu nakładek, struktury danych takie jak drzewa sufiksowe, tablice sufiksowe, RMQ, także w zwięzłych wersjach, czy też różne metody kompresji. Drugorzędnym celem przedmiotu jest także zaimplementowanie niektórych z tych narzędzi i sprawdzenie ich działania w praktyce.
14.	Treści programowe 1. Analiza i zastosowania algorytmów wyszukiwania wzorca: dokładnego, z niezgodnościami, z błędami. 2. Problemy podobieństwa tekstów (wspólne podciągi, nadciągi, odległość edycyjna, dopasowanie). 3. Struktury danych dla tekstów: drzewa i tablice sufiksowe, graf podsłów. 4. Zwięzłe struktury danych i ich zastosowanie w strukturach danych dla tekstów. 5. Różne metody kompresji (Huffmana, LZ78, LZ, kompresja gramatykowa), indeksowanie i wyszukiwanie wzorca w skompresowanych tekstach. 6 Conditional lower bounds dla problemów na tekstach. 7 Problem superstringu.

15. Zakładane efekty uczenia się

Wiedza

Zna kombinatoryczne własności słów typu lemat o okresowości, własności grafu odległości edycyjnej czy grafu nakładek	K_W07, K_W14
Zna liniowy algorytm konstrukcji drzewa sufiksowego i tablicy sufiksowej	K_W07, K_W14
Zna podstawowe wyniki mówiące o trudności problemów na tekstach i hipotezy, na których są one oparte	K_W14
Zna wybrane techniki używane do konstruowania zwięzłych struktur danych	K_W07

Umiejętności

Potrafi konstruować proste algorytmy przetwarzające teksty oparte na strukturach typu drzewo sufiksowe	K_U06
Potrafi skorzystać z gotowych bibliotek implementujących zwięzłe struktury danych	K_U15, K_U16
Potrafi czytać po angielsku teksty specjalistyczne na temat algorytmów tekstowych, typu publikacje na konferencjach CPM, SPIRE	K_U15, K_U20

16. Literatura obowiązkowa i zalecana

1. A. Apostolico and Z. Galil, editors, Pattern Matching Algorithms, Oxford University Press, New York, 1997
2. M. Crochemore and W. Rytter, Text Algorithms, Oxford University Press, New York, 1994
3. D. Gusfield, Algorithms on Strings, Trees, and Sequences, Cambridge University Press, New York, 1997
4. E. Ohlebusch, Bioinformatics Algorithms: Sequence Analysis, Genome Rearrangements, and Phylogenetic Reconstruction. Oldenbusch Verlag, 2013

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin, sprawdziany/kolokwia, prezentacja projektu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- Ćwiczenia: sprawdzian w połowie semestru oraz aktywność na zajęciach.
- Wykład: egzamin pisemny.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do egzaminu	30 godz.
przygotowanie raportu/prezentacji	20 godz.
przygotowanie do ćwiczeń	40 godz.

Sumarycznie

Łączna liczba godzin	180 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **An Introduction to Finite and Classical Model Theory** **An Introduction to Finite and Classical Model Theory**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **??**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Logika dla informatyków • Algebra **Niezbędne kompetencje:** • Umiejętność pisania i czytania dowodów matematycznych
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest wprowadzenie studentów w wybrane tematy teorii modeli skończonych. Studenci zapoznają się z technikami używanymi do porównywania siły wyrazu oraz ustalania złożoności obliczeniowej problemów spełnialności wybranych fragmentów logiki pierwszego rzędu.

14. Treści programowe

1. Wprowadzenie do tematyki + przykłady. Twierdzenie o zwartości z jego dowodem przy pomocy twr. Goedela. Konsekwencje twierdzenia o zwartości i niewyrażalność w FO.
2. Filtry i ultrafiltry (czyli brakujące kawałki materiału z logiki dla informatyków). Ultraprodukty i twierdzenie Łosia. Dowód twierdzenia o zwartości przy pomocy twierdzenia Łosia.
3. Ważne własności teorio-modelowe logik: interpolacja Craiga, projektywna definiowalność Beth, twierdzenia o zachowywaniu Łosia-Tarskiego i twierdzenie o zachowywaniu przez homomorphizmy. Porażki w świecie skończonych modeli.
4. Prawo zero-jedynkowe FO, czyli o tym że prawdopodobieństwo że losowa struktura spełnia phi to 0 lub 1. Niewyrażalność przy pomocy praw 0/1. Graf Rado i trochę na temat teorii omega-kategorycznych.
5. Gry Ehrenfeuchta-Fraisse'go jako uniwersalna metoda pokazywania niewyrażalności. Gry na porządkach liniowych. Typy, back-and-forth i dowód twierdzenia Fraisse'go. Modyfikacje gier E-F na przykładach gier z kamieniami i gier dla MSO.
6. FO wyraża tylko lokalne właśności: lokalności Hanf'a i Gaifman'a oraz postać normalna Gaifmana. Dowód twierdzenia Hanfa. Zastosowania lokalności do dowodzenia niewyrażalności oraz do efektywnych algorytmów parametryzowanych do ewaluacji zapytań w grafach o ograniczonym stopniu.
7. Order-invariant First-Order Logic, czyli trochę na temat piękna skończonych modeli. Parę słów o logice dla P.

15. Zakładane efekty uczenia się

Wiedza

Zna definicje podstawowych rozstrzygalnych fragmentów logiki pierwszego rzędu (logik) motywowanych informatyką. Wie o możliwych zastosowaniach takich logik w różnych obszarach informatyki (bazy danych, reprezentacja wiedzy, automatyczna weryfikacja).	K_W03, K_W14
Zna podstawowe metody pokazywania rozstrzygalności/nierozstrzygalności oraz ustalania złożoności takich logik. Rozumie zależność pomiędzy siłą wyrazu logiki a jej złożonością obliczeniową.	K_W03, K_W14
Zna podstawowe techniki porównywania siły wyrazu logik i potrafi zastosować.	K_W01

Umiejętności

Potrafi wykorzystywać poznane metody do badania rozstrzygalności różnych logik i do ustalania ich siły wyrazu.	K_U01, K_U02

Kompetencje społeczne

Potrafi samodzielnie rozwiązywać zadania związane z rozstrzygalnością fragmentów logiki pierwszego rzędu przy użyciu poznanych na wykładzie technik.	K_K03

16. Literatura obowiązkowa i zalecana

Książka przeglądowa:

- Leonid Libkin, Elements of Finite Model Theory
- Neil Immerman, Descriptive Complexity
- Serge Abiteboul, Foundations of databases
- Hubert Comon, Max Dauchet, Remi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Loding, Sophie Tison, Marc Tommasi: Tree Automata Techniques and Applications

Oprócz tego pewna liczba artykułów naukowych, których pełna lista będzie podana w czasie wykładu. M.in.:

- Mikolaj Bojanczyk, Michal Pilipczuk: Definability equals recognizability for graphs of bounded treewidth. LICS 2016: 407-416
- Roman Kontchakov, Michael Zakharyaschev: An Introduction to Description Logics and Query Rewriting. Reasoning Web 2014: 195-244
- Magdalena Ortiz, Mantas Simkus: Reasoning and Query Answering in Description Logics. Reasoning Web 2012: 1-53
- Martin Otto, Elementary proof of the van Benthem - Rosen characterisation theorem
- P. Blackburn, J. van Benthem, F. Wolter, Handbook of Modal Logic, Elsevier, 2007

17.	Metody weryfikacji zakładanych efektów uczenia się egzamin ustny, prezentacja rozwiązań zadań przy tablicy
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Aby zaliczyć ćwiczenia należy zdobyć podaną w regulaminie liczbę punktów. Punkty dostaje się za obecność na ćwiczeniach, prezentację przy tablicy rozwiązań zadań z list ćwiczeniowych oraz udział w dyskusji na temat rozwiązań. Egzamin ma formę ustną i polega na prezentacji dowodów kilku twierdzeń, przedstawionych wcześniej na wykładzie, wybranych z listy ogłoszonej przez wykładowcę.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** ćwiczenia — 30 godz. wykład — 30 godz. **Praca własna studenta** przygotowanie do ćwiczeń — 30 godz. studiowanie tematyki wykładów i literatury — 30 godz. egzamin — 1 godz. przygotowanie do egzaminu — 11 godz. **Sumarycznie** Łączna liczba godzin — 132 godz. Liczba punktów ECTS — 6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Architektury systemów komputerowych** **Computer Architecture for Programmers**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6ASK**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 15 godzin, ćwiczenio-pracownia — 30 godzin, repetytorium — 15 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Kurs: Wstęp do programowania w języku C • Logika cyfrowa (opcjonalnie) **Niezbędne kompetencje** • Umiejętność obsługi wiersza poleceń systemu uniksowego • Dobra znajomość języka C
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest wprowadzenie słuchaczy w tematykę budowy i zasady działania komputerów z punktu widzenia programisty. Po zakończeniu przedmiotu studenci będą rozumieć: reprezentację programów i danych w pamięci, wpływ architektury komputera na wydajność programów oraz jak pisać kod wolny od podatności na ataki.

14.	Treści programowe

1. Reprezentacja danych w pamięci komputera i reprezentacja pośrednia programów: kod trójadresowy, graf przepływu sterowania.
2. Dziwne właściwości arytmetyki całkowitoliczbowej i zmiennopozycyjnej (standard IEEE754).
3. Składnia asemblera: mnemoniki, operandy, dyrektywy.
4. Translacja konstrukcji języka C do asemblera (wyrażenia, instrukcje sterujące, typy rekordowe i tablicowe)
5. Konwencja wołania procedur i organizacja stosu wywołań (ABI)
6. Konsolidacja i ładowanie: ELF, sekcje, relokacje, układ pamięci, biblioteki i konsolidator dynamiczny (ld.so).
7. Ataki na oprogramowanie: błąd przepełnienia bufora, wstrzykiwanie kodu, technika ROP, techniki niwelowania podatności (ASLR, WX).
8. Organizacja pamięci operacyjnej i masowej (magnetycznej i półprzewodnikowej) - czas dostępu.
9. Komunikacja procesora z urządzeniami peryferyjnymi: przerwania, MMIO, DMA.
10. Pamięć podręczna: zasada lokalności dostępów, organizacja pamięci podręcznej, polityki wymiany i zapisu.
11. Analiza i optymalizacja programów pod kątem lokalności dostępów do pamięci.
12. Pamięć wirtualna: motywacja, przestrzeń adresowa, zbiór roboczy i rezydentny, błąd strony, stronicowanie na żądanie.
13. Translacja adresów: deskryptory stron, hierarchiczna tablica stron, TLB, przełączanie przestrzeni adresowych.
14. Mikroarchitektura: przetwarzanie superskalarne, OoO, spekulacja, przewidywanie skoków.

15. Zakładane efekty uczenia się

Wiedza

Wie jak programy komunikują się ze sprzętem i systemem operacyjnym	K_W08, K_W15
Rozumie wpływ architektury maszyny na implementację kompilatorów i systemów operacyjnych	K_W15
Wie jak wykorzystać organizację maszyny do implementacji wydajnych programów	K_W08, K_W15

Umiejętności

Umie pisać programy zachowujące się poprawnie w obliczu ograniczeń maszynowej reprezentacji typów numerycznych	K_U16
Potrafi wybrać i zastosować właściwe narzędzia i metodologie do mierzenia wydajności kodu programów	K_U19
Potrafi przeprowadzić analizę kodu pod względem wykorzystania pamięci podręcznej i przeorganizować go celem optymalizacji	K_U06
Umie przeanalizować kod maszynowy generowany przez kompilator pod względem poprawności i wydajności	K_U06
Potrafi użyć narzędzi profilujących do znalezienia fragmentów programu wymagających optymalizacji	K_U19
Umie przeanalizować kod programu pod kątem występowania podatności na ataki	K_U18
Potrafi wykonać inżynierię wsteczną fragmentów kodu maszynowego celem ustalenia pełnionych przez nie funkcji	K_U18
Pomaga członkom zespołu w pisaniu oprogramowania wolnego od usterek związanych z niskopoziomowymi aspektami działania komputerów	K_U23

Kompetencje społeczne

Potrafi właściwie wskazać te części projektu komercyjnego, które opłaca się optymalizować	K_K07

16. Literatura obowiązkowa i zalecana

- Computer Systems: A Programmer's Perspective (3rd edition) Randal E. Bryant, David R. O'Hallaron

17. Metody weryfikacji zakładanych efektów uczenia się

Egzamin pisemny, prezentacja rozwiązań zadań przy tablicy, napisanie i prezentacja niedużych programów, znalezienie luki w podatnym oprogramowaniu, analiza zachowania programów.

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia ćwiczeń i pracowni należy zdobyć co najmniej połowę liczby punktów za zadania ćwiczeniowe i programistyczne. Punkty za wszystkie wyżej wymienione aktywności liczą się łącznie. Na punkty z poszczególnych aktywności mogą być narzucone minimalne progi zaliczeniowe. Egzamin końcowy jest w formie pisemnej i wymaga zdobycia co najmniej połowy punktów. Zadania mogą mieć następujące formy: analiza kodu programu, prezentacja działania mechanizmu sprzętowego, pisanie niewielkich fragmentów kodu, test wielokrotnego wyboru.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

repetytorium	15 godz.
ćwiczenio-pracownia	30 godz.
pracownia	15 godz.
wykład	30 godz.

Praca własna studenta

rozwiązywanie zadań programistycznych	60 godz.
udział w egzaminie	3 godz.
przygotowanie do egzaminu	10 godz.
rozwiązywanie zadań z list	80 godz.

Sumarycznie

Łączna liczba godzin	243 godz.
Liczba punktów ECTS	8

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Geometric Optimization** **Geometric Optimization**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DI2GeoOptA**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 20 godzin, ćwiczenia — 10 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Completed courses:** • algorithms and data structures **Additional skills that will be usefull:** • linear programming • standard tools from probability theory
13.	Cele kształcenia dla przedmiotu The course will provide students with algorithmic techniques that are in useful in handling geometric optimization problems. The main focus will be on embedding techniques that allow to approximately preserved distances while reducing the complexity of the instance of the studied optimization problem.
14.	Treści programowe 1. Metric Embedding into normed spaces 2. Dimension Reduction 3. Doubling dimension, Epsilon-nets, VC-dimension 4. Nearest neighbor search and its approximation 5. Embedding into trees and Probabilistic embedding 6. Euclidean TSP 7. Approximate Sparsest Cut

15.	Zakładane efekty uczenia się

Knowledge

Understands properties of selected metric spaces and norms	K_W14
Knows standard algorithmic techniques based on metric embeddings	K_W14

Skills

Can construct algorithms utilizing properties of problem speciffic metric spaces	K_U06, K_U18, K_U20
Is capable of proving selected properties of geometric algoritms	K_U06, K_U18, K_U20

16. Literatura obowiązkowa i zalecana

1. Geometric approximation algorithms (Author- Sariel Har-Peled)
2. Algorithmic geometry (Author- J.D. Boissonnat, Mariette Yvinec and Herve Bronniman)

17. Metody weryfikacji zakładanych efektów uczenia się

Exam,
exercise solving,
oral presentations (optional)

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Exercise solving.
Written final exam.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	20 godz.
ćwiczenia	10 godz.

Praca własna studenta

preparing solutions for exercises	10 godz.
Preparing for the exam	10 godz.
participating in the exam	2 godz.
literature study	20 godz.
additional consultations if necessary	0 godz.

Sumarycznie

Łączna liczba godzin	72 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Inżynieria oprogramowania** **Software engineering**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6IO**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Gotowość do pracy zespołowej
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zdobycie podstawowych: praktycznej wiedzy i umiejętności wymaganych w pracy na stanowisku inżynier oprogramowania. Wykład ma na celu przekazanie studentom podstawowej, praktycznej wiedzy z zakresu inżynierii oprogramowana. Przedstawia informacje dotyczące wykorzystania dziedzin wiedzy inżynierii. Istnieje możliwość wygłoszenia paru wykładów przez specjalistów z czołowych zagranicznych i polskich firm np: VOLVO IT, Capgemini. Podczas ćwiczeń/pracowni studenci zespołowo realizują, zgodnie z najlepszymi praktykami 10 podstawowych dyscyplin inżynierii oprogramowania projekt konkretnego systemu informatycznego. Projekt zawiera również fragmenty kodu. Podczas zajęć laboratoryjnych wytwarzana są diagramy dot. fragmentów projektowanego systemu oraz fragmenty kodu. Realizowany projekt ma za zadanie przygotowanie wg standardów światowych do pracy w przemyśle wytwarzania oprogramowania.
14.	Treści programowe • Wprowadzenie - historia inżynierii oprogramowania – kodeks etyki informatyka (2g.) • Omówienie dyscyplin podstawowych i pokrewnych inżynierii oprogramowania (2g.) • Studium wykonalności (2g.) • Zarządzanie wymaganiami (2g.) • Projektowanie oprogramowania (2g.) • Kodowanie (2g.) • Testowanie oprogramowania (2g.) • Jakość oprogramowania (2g.) • Eksploatacja oprogramowania (2g.) • Zarządzanie konfiguracją oprogramowania (2g.), • Procesy w życiu oprogramowania (2g.), • zarządzanie zespołowym wytwarzaniem oprogramowania (2g.), • Narzędzia do zarządzania konfiguracjami i wersjami oprogramowania (2g.). • Kolokwium (4g.)

15.	Zakładane efekty uczenia się

Wiedza

Zna metody inżynierii oprogramowania dotyczące analizy i specyfikacji problemu informatycznego (wzorce projektowe, architektura oprogramowania, analiza i projektowanie obiektowe). Zna metody i narzędzia wytwarzania i rozwijania oprogramowania (narzędzia do analizy wymagań i modelowania, narzędzia do testowania, narzędzia do podglądu kodu, narzędzia do zarządzania projektem konfiguracjami i wersjami oprogramowania). Ma wiedzę na temat cyklu życia projektu informatycznego, walidacji i weryfikacji, technik utrzymywania oprogramowania (refaktoryzacji) i potrafi ją stosować w przypadku podstawowych systemów informatycznych	K_W15
Ma podstawową wiedzę dotyczącą prawnych, społecznych i etycznych aspektów pracy informatyka, ze szczególnym uwzględnieniem zagadnień inżynierii oprogramowania	K_W16
Ma podstawową wiedzę o strukturze, działaniu i cyklach życia systemów informatycznych oraz o oprogramowaniu urządzeń komputerowych	K_W15
Ma podstawową wiedzę o strukturze, działaniu i cyklach życia systemów informatycznych oraz o programowaniu urządzeń komputerowych	K_W15

Umiejętności

Potrafi projektować pisać, uruchamiać i testować programy w wybranych paradygmatach i środowiskach programistycznych. Projektuje oprogramowanie zgodnie z metodyką obiektową, posługuje się wzorcami projektowymi i potrafi stworzyć model obiektowy prostego systemu w języku UML. Potrafi wykonać prostą analizę funkcjonowania systemu informatycznego, stworzyć i przeprowadzić plan testowania, wziąć udział w inspekcji oprogramowania	K_U16
Przy formułowaniu i rozwiązywaniu zadań inżynierskich dostrzega ich aspekty systemowe i pozatechniczne, w szczególności potrafi dokonać wstępnej analizy ekonomicznej podejmowanych działań inżynierskich, w tym szacowania czasochłonności implementowania wybranych metod działania	K_U18

Kompetencje społeczne

Jest świadom roli i znaczenia wiedzy w rozwiązywaniu problemów o charakterze poznawczym i praktycznym typowych dla inżynierii oprogramowania	K_K02
Wykazuje gotowość do pełnienia, w sposób odpowiedzialny i respektujący zasady etyki zawodowej, ról zawodowych wymagających kompetencji zdobywanych w ramach studiów na kierunku informatyka, w szczególności jest gotów do formułowania opinii o zagadnieniach związanych z inżynierią oprogramowania	K_K03

16. Literatura obowiązkowa i zalecana

- Jaszkiewicz A.: Inżynieria oprogramowania
- Sacha K.: Inżynieria oprogramowania
- Sommerville i.: Inżynieria oprogramowania
- Martin R.C.: Zwinne wytwarzanie oprogramowania. Najlepsze wzorce, zasady i praktyki.
- Cohn M.: Agile. Metodyki zwinne w wytwarzaniu oprogramowania.

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin, kolokwium, prezentacja projektu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- wykład: egzamin (treści weryfikowane na egzaminie, np. modele procesu wytwarzania oprogramowania, zarządzanie jakością, metody testowania),
- ćwiczenia: omówienie rozwiązań zadań,
- projekt: omówienie dokumentacji projektowej oraz prezentacja fragmentów opracowanej aplikacji

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie projektu godz	20 godz.
przygotowanie do egzaminu godz	20 godz.
studiowanie tematyki wykładów i literatury godz	12 godz.
przygotowanie do ćwiczeń godz	30 godz.

Sumarycznie

Łączna liczba godzin	142 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kombinatoryka** **Combinatorics**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DZ6KMB**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Matematyka dyskretna
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie studentów z technikami kombinatorycznymi potencjalnie przydatnymi w rozwiązywaniu problemów algorytmicznych.
14.	Treści programowe • Liczby szczególne • Wzory inwersyjne • Zliczanie istotnie różnych obiektów • Wzór Eulera-Maclaurina i wzór Stirlinga • Teoria Ramseya • Metoda probabilistyczna Erdosa
15.	Zakładane efekty uczenia się **Wiedza**

Zna zagadnienia związane z liczbami szczególnymi	K_W05, K_W14
Zna zaawansowane zagadnienia zliczania związane z wzorami inwersyjnymi i teorią Polya	K_W05, K_W14
Posiada znajomość teorii Ramsey'a	K_W05, K_W14
Zna techniki związane z metodą probabilistyczną	K_W05,K_W13, K_W14

Umiejętności

Umie rozwiązywać proste problemy wymagające użycia liczb szczególnych	K_U04, K_U13
Umie dla prostych problemów użyć wzorów inwersyjnych i twierdzenia Polya	K_U04, K_U13
Umie rozwiązywać problemy używając naiwnej metody probabilistycznej	K_U04, K_U13

16.	Literatura obowiązkowa i zalecana • R.L. Graham, D.E. Knuth, O. Patashnik, Matematyka konkretna, PWN 1996. • W.Lipski, W.Marek, Analiza kombinatoryczna, PWN 1988. • Z.Palka, A.Ruciński, Niekonstruktywne metody matematyki dyskretnej, PWN, Warszawa 1996.
17.	Metody weryfikacji zakładanych efektów uczenia się Prezentacja rozwiązań zadań, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Warunkiem zaliczenia jest uzyskanie minimum punktowego za zaprezentowane na tablicy rozwiązania zadań. Warunkiem zaliczenia egzaminu jest zdobycie minimalnej wymaganej liczby punktów z egzaminu końcowego. Szczegółowe warunki zaliczenia są określone w regulaminie przedmiotu.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** ćwiczenia: 30 godz. wykład: 30 godz. **Praca własna studenta** przygotowanie do egzaminu - godz: 8 godz. studiowanie literatury do przedmiotu - godz: 40 godz. udział w egzaminie - godz: 2 godz. rozwiązywanie zadań z list - godz: 40 godz. **Sumarycznie** Łączna liczba godzin: 150 godz. Liczba punktów ECTS: 6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Komunikacja człowiek-komputer** **Human-Computer Interaction**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP9HCI**
6.	Rodzaj przedmiotu/modułu **Informatyczny inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Przedmiot „Komunikacja człowiek-komputer" jest wybitnie interdyscyplinarny (zob. „Opis przedmiotu" powyżej sylabusa). **Niezbędne lub pożądane kompetencje** • Zainteresowana osoba powinna zdradzać otwartość na humanistyczne aspekty najwyższej warstwy oprogramowania i zastosowań informatyki. • Zainteresowana osoba powinna umieć i (lub) chcieć analizować i poddawać refleksji przedmioty w otoczeniu, w tym zewnętrzną otoczkę programów komputerowych. • Zainteresowana osoba powinna być dociekliwa i umieć dociekać "prawd" o użyteczności przedmiotów (w tym programów komputerowych) w kontaktach z potencjalnymi, rzeczywistymi, typowymi lub wyrafinowanymi użytkowniczkami i użytkownikami. • Prócz oczywistych kwalifikacji informatycznych, w tym technik inżynierii oprogramowania (osobny wykład) i niektórych aspektów AI (odrębny wykład), zainteresowania psychologią, socjologią, antropologią, lingwistyką, fizjologią, ergonomią, wzornictwem przemysłowym, a nawet filozofią i sztuką dają osobie studiującej KCK (HCI) przewagę twórczą i produkcyjną w czasie zajęć. • Pożądana jest elementarna znajomość któregoś z języków tworzenia interaktywnych stron WWW (np. HTML, osobny wykład), aplikacji okienkowych (inny wykład), a także wprawne operowanie którymś z systemów konstruowania makiet interfejsów (ćwiczonym i opanowywanym samodzielnego jako „wartość dodana”22 zajęć). • Elementarna wiedza o cyklu wytwarzania oprogramowania (osobny wykład z inżynierii oprogramowania).

13.	Cele kształcenia dla przedmiotu Uczestniczka lub uczestnik kursu ma okazję poznać elementy składające się na dobry interfejs, w rozumieniu wywierania jak najkorzystniejszych wrażeń na użytkowniku docelowym, opanować podstawową wiedzę psychologiczną i socjologiczną w zakresie przydatnym w projektowaniu GUI, zrozumieć związki między udanym interfejsem a sukcesem ekonomicznym, poznać podstawowe zasady ergonomii i komunikacji werbalnej i wizualnej (a także dźwiękowej) z zastosowaniem w interfejsach komputerowych, reguły operowania przestrzenią, barwą, krojami pisma i innymi materiałami do budowy interfejsów oraz prześledzić techniki designerskie pomocne w tworzeniu interfejsów nowoczesnych komputerów i skomputeryzowanych urządzeń. Opanowuje praktycznie wybrane narzędzie projektowania makiet interfejsów graficznych.
14.	Treści programowe Niżej podano tytuły wykładów. Szczegółowe wypunktowanie poruszanych zagadnień jest sukcesywnie uwidaczniane w trakcie semestru na stronie wykładu w witrynie SKOS. 1. Wstęp 2. Wyszukiwanie potrzeb 3. Analiza potrzeb [i] ludzi 4. Prototypowanie czas zacząć 5. Naucz się krytykować 6. Kształtowanie interfejsów 7. Zmysły mylą 8. Barwą, krojem i układem 9. Układ, liczy się układ 10. Inwentaryzacja GUI 11. GUI — dokończenie remanentu 12. Materiały do 17 wykładów studenckich
15.	Zakładane efekty uczenia się **Wiedza** Uczestniczka lub uczestnik kursu nabiera nowego, racjonalnego, krytycznego spojrzenia i pojmowania graficznych interfejsów użytkownika — K_W15 Ma świadomość zasad rządzących konstruowaniem udanych GUI — K_W15 Poznaje miary jakościowe przydatne w ocenie interfejsów (listy kontrolne, w sumie ponad 200 pozycji szczegółowych) — K_W15 Opanowuje elementarne, potrzebne w projektowaniu interfejsów wiadomości z zakresu psychologii, socjologii, neurologii, kognitywistyki, typografii i estetyki — K_W15 Rozumie, co to znaczy design i czym zajmują się osoby projektujące interfejsy komputerowe i inne — K_W15, K_W16 Ma świadomość, jak wiele we współczesnym obrazkowym świecie zależy także od doboru słów i jasności opisów werbalnych w interfejsach — K_W16 **Umiejętności** Potrafi uwzględniać w procesie projektowania i kształtowania interfejsów komputerowych zasady działania pamięci człowieka oraz możliwości i ograniczenia zmysłów ludzkich — K_U19 Potrafi analizować projektowane i gotowe interfejsy, zarówno komputerów, jak i innych skomputeryzowanych przedmiotów i maszyn — także przedmiotów niebędących komputerami — K_U21 Potrafi wykorzystać w projektowaniu interfejsów elementarną wiedzę z zakresu ergonomii — K_U22 Potrafi krytycznie oceniać interfejsy — zarówno komputerowe, jak i inne, dotyczące maszyn, przedmiotów i urządzeń (także codziennego użytku) — K_U21 Portafi sporządzić wykaz potrzeb — wcześniej je dostrzegając (wydobywając) — na podstawie obserwacji, wywiadów, ankiet, list kontrolnych — K_U19 Potrafi wynajdywać możliwości ulepszeń w istniejących interfejsach — K_U19 Potrafi testować jakość i przydatność poszczególnych interfejsów — K_U19 Potrafi operować barwą, odcieniem, kształtem, przestrzenią, proporcjami i rozmieszczeniem elementów interfejsów, "gospodarować pustką" itp — K_U19 Potrafi w sposób elementarny dobierać kroje pisma i sporządzić projekt aspektów typograficznych interfejsu — K_U19 Umie mówić i pisać jasno i poprawnie po polsku — K_U01

16. Literatura obowiązkowa i zalecana

Istnieje wiele dobrych podręczników KCK (HCI). Niektóre są dostępne bezpłatnie w Sieci. Prócz tego Sieć zawiera mnóstwo artykułów poświęconych projektowaniu witryn i ogólniej — kwestiom designerskim interfejsów komputerowych. Oto niektóre lektury, oprócz dodatkowych, proponowanych na wykładzie:

1. Lynch P. J., Horton S.: Web Style Guide. 3rd Edition. Yale University Press 2008. Książka dostępna w Sieci bez opłat.
2. Van Duyne D. K., Landay J. A., Hong J. I.: The Design of Sites: Patterns for Creating Winning Web Sites. Prentice Hall Professional 2007.
3. Leventhal L. M., Barnes J. A.: Usability Engineering: Process, Products, and Examples. Pearson/Prentice Hall 2008.
4. Dzieła klasyków dziedziny, D. Normana i J. Nielsena – dostępne w Sieci.
5. Inne wartościowe artykuły, przemyślenia, instruktaże i normy, m.in. na stronach firm Microsoft i Google.
6. Kilkadziesiąt zestawów materiałów do wykładów studenckich, dostępnych na stronie wykładu w Sieci.

17. Metody weryfikacji zakładanych efektów uczenia się

- Ocena wypowiedzi, pomysłów i uwag osób studiujących, dokonywana podczas ćwiczeń.
- Ocena (krytyka) etapów realizacji projektów na pracowni.
- Ocena prezentowanych zadań specjalnych i dodatkowych.
- Ocena wykładów studenckich, poszerzających i unowocześniających dziedzinę; dla ochotników.
- Egzamin pisemny (wybór z ponad 360 pytań).

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Warunki i forma zaliczenia poszczególnych komponentów przedmiotu
Zaliczenie ćwiczeń i związanej z wykładem pracowni wymaga:

- aktywnego, twórczego uczestnictwa w ćwiczeniach i pracowni;
- opracowania wartościowej i pełnej makiety wybranego interfejsu (w dążeniu do ulepszenia istniejącej aplikacji lub zupełnie nowego);
- wykazania się umiejętnością konstruktywnych ocen i krytyki przedmiotów użytkowych, w tym aplikacji komputerowych;
- umiejętnego przedstawienia i przekazania własnego wyrobu (GUI).

Dodatkowo, w miarę możliwości czasowych, studenci mogą przygotować na koniec semestru własne wykłady wzbogacające, a zwłaszcza unowocześniające, wiedzę o trendach, modach i praktyce w dziedzinie KCK (HCI).

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do ćwiczeń	15 godz.
praca nad projektem	30 godz.
studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do pracowni	15 godz.
udział w egzaminie	2 godz.

Sumarycznie

Łączna liczba godzin	152 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Konstrukcja kompilatorów** **Compiler construction**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DI1KK**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Metody programowania
13.	Cele kształcenia dla przedmiotu Podstawowym celem przedmiotu jest zapoznanie studentów z budową współczesnych kompilatoróworaz z technikami stosowanymi w ich konstruowaniu.
14.	Treści programowe • Wstęp: struktura kompilatora • Leksowanie • Parsowanie • Analiza semantyczna • Generowanie kodu pośredniego • Analiza statyczna i optymalizacja kodu • Generowanie kodu wynikowego • Wsparcie dla języków wysokopoziomowych

15.	Zakładane efekty uczenia się

Wiedza

po zaliczeniu przedmiotu student będzie znał budowę standardowego kompilatora oraz podstawowe pojecia i techniki niezbędne do konstruowania poszczególnych modułów, w tym • wyrażenia regularne • gramatyki bezkontekstowe • reguły typowania • analizę przepływu danych • techniki optymalizacji kodu	K_W04, K_W08, K_W14, K_W15

Umiejętności

umie skonstruować kompilator prostego języka imperatywnego	K_U16, K_U18, K_U19

16. Literatura obowiązkowa i zalecana

Modern Compiler Implementation in ML, Appel Compilers: Principles, Techniques, and Tools, Aho Ullman Engineering: A Compiler, Cooper Torczon Advanced Compiler Design and Implementation, Muchnick The Garbage Collection Handbook, Jones

17. Metody weryfikacji zakładanych efektów uczenia się

- egzamin
- projekty programistyczne polegające na doimplementowaniu do przygotowanego szablonu poszczególnych funkcjonalności kompilatora

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia ćwiczenio-pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za zadania ćwiczeniowe i pracownie. Punkty w obu wymienionych kategoriach liczą się osobno. Egzamin ma formę pisemną; egzamin poprawkowy może mieć formę ustną. Do jego zaliczenia trzeba uzyskać określoną liczbę punktów.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do ćwiczeń	10 godz.
przygotowanie do pracowni	10 godz.
studiowanie tematyki wykładów i literatury	10 godz.
praca nad projektem	50 godz.
przygotowanie do egzaminu	10 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kryptografia** **Cryptography**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DZ6KRP**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – zastosowania informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Matematyka dyskretna **Zalecane dodatkowo** • Algorytmy i struktury danych
13.	Cele kształcenia dla przedmiotu Celem wykładu jest zapoznanie uczestnikow z nowoczesnymi metodami służącymi do ochrony prywatnosci danych elektronicznych, autentyfikacji użytkownikow systemów komputerowych, zabezpieczaniu przed nieuprawnionymi modyfikacjami danych i innymi tego typu zastosowaniami opartymi na technikach kryptograficznych. Znaczenie tego typu metod ujawnia się szczególnie ostro w epoce powstawania globalnych sieci komputerowych, gdzie systemy operacyjne nie gwarantują już bezpieczeństwa.

14. Treści programowe

- Klasyczne szyfry symetryczne i ich kryptoanaliza. Bezpieczeństwo doskonałe i one-time pad
- Współczesne szyfry blokowe (DES, IDEA, AES) i ich podstawowe tryby użycia (ECB, CBC, CFB, OFB, CTR)
- Kryptoanaliza różnicowa i liniowa
- Bezpieczeństwo trybu CBC
- Generowanie dużych liczb pierwszych, test Millera-Rabina
- Algorytmy faktoryzacji i znajdowania logarytmu dyskretnego
- Protokół Diffie-Hellmana. Szyfry asymetryczne RSA i ElGamala. Semantyczne bezpieczeństwo algorytmu ElGamala
- Generatory pseudolosowe
- Funkcje haszujące
- Algorytmy podpisu cyfrowego
- Zobowiązania bitowe i dzielenie tajemnic
- Protokoły z wiedzą zerową

15. Zakładane efekty uczenia się

Wiedza

zna zagadnienia dotyczące zasad konstrukcji szyfrów blokowych i tryby ich pracy	K_W14
zna zagadnienia teorioliczbowe związane z kryptografią asymetryczną	K_W14
zna kryteria które powinny spełniać generatory pseudolosowe i funkcje skrótu	K_W14
zna zagadnienia dotyczące podpisów cyfrowych	K_W15
zna zagadnienia dotyczące dowodów z wiedzą zerową	K_W15

Umiejętności

umie zidentyfikować właściwe sposoby dobrania szyfru asymetrycznego wraz z trybem jego działania	K_U19
umie prawidłowo stosować generatory pseudolosowe, funkcje haszujące i podpisy cyfrowe	K_U19
umie tworzyć proste dowody z wiedzą zerową	K_U19

16. Literatura obowiązkowa i zalecana

- Douglas Stinson: Kryptografia. W teorii i praktyce.
- Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone: Handbook of applied cryptography.
- Neal Koblitz: Algebraiczne aspekty kryptografii.

17. Metody weryfikacji zakładanych efektów uczenia się

Prezentacja rozwiązań zadań, egzamin pisemny

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Warunkiem zaliczenia ćwiczeń jest uzyskanie minimum punktowego za zaprezentowane przy tablicy zadania.
Warunkiem zaliczenia egzaminu jest zdobycie minimalnej wymaganej liczby punktów z egzaminu końcowego. Szczegółowe warunki zaliczenia są określone w regulaminie przedmiotu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	8 godz.
studiowanie literatury do przedmiotu	40 godz.
udział w egzaminie	2 godz.
rozwiązywanie zadań z list	40 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Logika cyfrowa** **Digital logic**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-LogCyf**
6.	Rodzaj przedmiotu/modułu **Informatyczny inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Logika dla informatyków • Wstęp do informatyki **Niezbędne kompetencje:** • Umiejętność programowania w dowolnym języku programowania • Znajomość podstaw rachunku zdań
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest nauczenie zasad rządzących funkcjonowaniem elektroniki cyfrowej oraz typowych rozwiązań stosowanych przy projektowaniu układów. Zajęcia koncentrują się na następujących pojęciach: logika kombinacyjna i sekwencyjna, algebra Boole'a, automaty skończone. Przedmiot ma charakter praktyczny, jego ważnym elementem jest nauczenie posługiwania się językiem opisu sprzętu.

14.	Treści programowe • Wprowadzenie do układów cyfrowych • Technologie implementacji układów cyfrowych • Elementy SystemVeriloga jako języka opisu sprzętu • Optymalizacja funkcji logicznych • Układy arytmetyczne • Podstawowe układy kombinacyjne • Zatrzaski i przerzutniki • Liczniki • Układy pamięci ROM i RAM • Układy programowalne PAL, PLA, podstawy CPLD i FPGA • Automaty Moore'a i Mealy'ego • Układy sekwencyjne • Podstawy architektury RISC V • Budowa prostego procesora
15.	Zakładane efekty uczenia się **Wiedza** zna podstawowe elementy budowy układów cyfrowych - bramki logiczne - oraz ich związek z algebrą Boole'a — K_W03, K_W02 posiada wiedzę na temat reprezentowania danych w formie cyfrowej — K_W02 zna podstawowe układy bramkowe kombinacyjne i sekwencyjne, w tym: multipleksery, przerzutniki, liczniki, pamięci — K_W14 zna pojęcie automatu skończonego, automatu Moore'a i Mealy'ego — K_W14 rozumie działanie podstawowych implementacji procesorów (jednocyklowej, wielocyklowej i potokowej) — K_W15 **Umiejętności** potrafi ocenić efektywność (długość ścieżki krytycznej) i rozmiar (liczba bramek) układu cyfrowego — K_U18 potrafi zaprojektować układ cyfrowy zoptymalizowany ze względu na szybkość, koszt lub zużycie energii — K_U18, K_U19 potrafi projektować i minimalizować automaty skończone oraz implementować z ich pomocą algorytmy jako układy sekwencyjne z wydzieloną ścieżką sterowania i danych — K_U13 potrafi modelować układy kombinacyjne i sekwencyjne w języku SystemVerilog — K_U03, K_U16
16.	Literatura obowiązkowa i zalecana **Literatura obowiązkowa:** • Fundamentals of Digital Logic with Verilog Design; Brown, Vranesic; McGraw-Hill [roz. 1-7] • Digital Design: With an Introduction to the Verilog HDL, VHDL, and SystemVerilog (6th Edition); Mano, Ciletti; Pearson • Computer Organization and Design, the Hardware/Software Interface, RISC-V Edition [roz. 4] **Literatura zalecana:** • Verilog HDL (2nd Edition); Palnitkar • A Verilog HDL Primer, Third Edition; Bhasker • Standard IEEE 1364-2005 (Verilog 2005) • Standard IEEE 1800-2012 (SystemVerilog 2012) • Lessons in Electric Circuits; Kuphaldt [tom IV]
17.	Metody weryfikacji zakładanych efektów uczenia się prezentacja rozwiązania zadania, napisanie programu w języku opisu sprzętu, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia ćwiczenio-pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za zadania ćwiczeniowe i pracowniowe. Końcowa punktacja jest liczona przy użyciu średniej ważonej, aby waga ćwiczeń i pracowni wynosiła 50. Zadania ćwiczeniowe sprawdzane są podczas prezentacji przy tablicy. Na punktację zadań pracownianych składa się wyliczana automatycznie ocena za poprawność oraz indywidualna ocena prowadzącego. Egzamin ma formę pisemną, do jego zaliczenia konieczne jest zdobycie odpowiedniej liczby punktów.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

studiowanie tematyki wykładów i literatury	20 godz.
przygotowanie do pracowni	30 godz.
przygotowanie do ćwiczeń	30 godz.
przygotowanie do egzaminu	7 godz.
udział w egzaminie	3 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Machine Learning** **Machine Learning**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-ML**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – zastosowania informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Completed courses:** • "Wstęp do informatyki" lub "Algorytmy i struktury danych" • Soft requirement of "Sztuczna Inteligencja" and "Rachunek Prawdopodobieństwa i Statystyka" **Required Competences:** • Python programming, good to know numpy and scipy numerical libraries
13.	Cele kształcenia dla przedmiotu Presentation of fundamentals of machine learning. Presentation of main machine learning techniques: naive Bayes classification, decision trees, and linear regression. Demonstration of practical application of machine learning to real-world problems.
14.	Treści programowe 1. Fundamentals of statistical reasoning. 2. Naive Bayes classifier. 3. Logistic regression. 4. Decision trees. 5. Basics of neural networks. 6. Bias-variance tradeoff, model regularization and ensembling. 7. Boost classifiers, gradient boosting. 8. PAC learning theory. 9. Matrix factorization, recommender systems.

15. Zakładane efekty uczenia się

Knowledge

Student knows the difference between supervised learning, unsupervised learning, reinforcement learning.	K_W14
Student knows application domains of machine learning problems (regression, classification, clustering, frequent itemset mining, recommendations)	K_W15
Student knows and understands the probabilistic formulation of machine learning models	K_W14
Student knows the details of classical machine learning techniques: naive Bayes classifier, linear regression, decision trees	K_W14, K_W15
Student understands the theoretical (PAC) and practical requirements for success of machine learning projects (understands model bias and variance, knows the importance of proper training, development and test set creation, how to properly measure model performance)	K_W14, K_W15

Skills

Student can design and execute a machine learning project	K_U16, K_U17
Student can implement machine learning algorithms	K_U16, K_U17
Student can choose machine learning techniques suitable for the given problem and appropriate given the nature of the problem, the availability and quality of data, and run-time and training time requirements (costs, computational complexity, need for understandability)	K_U12, K_U19
Student can read and write algorithm description and technical documentation in English	K_U20

Social competences

Student understands importance of machine learning and artificial intelligence methods for the functioning of society, understands associated opportunities and dangers	K_K02, K_K04
Student shows independent thinking and actions when solving machine learning problems, is aware of the possibility of making mistakes by himself and others, shows criticism of the obtained results	K_K06

16. Literatura obowiązkowa i zalecana

1. Kevin Murphy, Machine Learning: A Probabilistic Perspective
2. Christopher Bishop, Pattern Analysis and Machine Learning

Supplementary reading:

- Trevor Hastie, Robert Tibshirani, Jerome Friedman: The Elements of Statistical Learning
- Leo Breiman: Classification and Regression Trees
- online materials for Stanford course CS229.

17. Metody weryfikacji zakładanych efektów uczenia się

final exam, lab and written assignments, final project

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Lecture

final exam

Lab assignments

The student must obtain at least 60 of points for assignments (about 6 sets per semester) and at least 60 points for the final project (final month of the course). Typical projects deal with implementing a recent research paper, applying methods from the course to new data or research projects settled individually with the instructor.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

exam	2 godz.
work on the project	20 godz.
preparation of a report for the final project	5 godz.
studying for the final exam	10 godz.
preparation for lab assignments	30 godz.
preparation for written assignments	25 godz.
studying for the lecture and reading literature	10 godz.

Sumarycznie

Łączna liczba godzin	162 godz.
Liczba punktów ECTS	7

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Neural Networks and Natural Language Processing** **Neural Networks and Natural Language Processing**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-NNaNLP**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – zastosowania informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Completed courses:** • Machine Learning or Sztuczna Inteligencja • Soft requirement on „Rachunek Prawdopodobieństwa i Statystyka” **Required Competencies:** • Knowledge of Python (ideally with the numpy and scipy numerical libraries) • Basics of probability and statistics (random variables, normal distribution, maximum likelihood estimation).
13.	Cele kształcenia dla przedmiotu Presentation of state-of-the-art deep learning techniques and their areas of applicability with the strong emphassis on natural language processing.
14.	Treści programowe • A recap on neural network operation and implementation in PyTorch. • Practical aspects of neural network application: regularization, on-line and batch learning, hyperparameter choice. • Image processing using neural networks: convolutional networks for classification, detection and segmentation of images. • Neural Networks for language modelling: feed forward networks (incl. Word2Vec), Convolutional Models, Recurent Neural Networks (incl. LSTM, GRU, and QRNN), Transformers • Sequence tagging (Part-of-Speech tagging, and Named Entity Recognition) using Conditional Random Fields (LSTM-CRF) • Encoder-decoder Models (with or without attention) for sequences • BERT and its variants and different pretraining strategies • Neural parsing and unsupervised parsing as a way of discovering the structure of language

15. Zakładane efekty uczenia się

Knowledge

The student knows the theoretical foundations of fitting parameters of a neural network using the maximum likelihood principle	K_W13, K_W14
Student knows numerical optimization algorithms required for neural network training and understands the trade-offs of batch and on-line training	K_W06, K_W07
Student knows popular deep learning frameworks and natural language processing libraries	K_W15
Student understands feed-forward, convolutional and recurrent neural networks	K_W14, K_W15
Student knows how to solve several NLP problems using neural nets	K_W15
Student understands importance of deep learning and artificial intelligence methods for the functioning of society, understands associated opportunities and dangers	K_W16

Skills

Student can understand recent deep learning literature	K_U15, K_U20
Student can execute a deep learning project, including properly writing a problem statement and measuring performance indicators	K_U16, K_U19
Student can fit neural networks using numerical optimization techniques	K_U16, K_U18
Student can implement basic feed-forward, convolutional and recurrent neural networks	K_U16, K_U17
Student can implement state-of-the-art deep learning models described in scientific literature	K_U15, K_U16, K_U17, K_U20
Student can use popular NLP libraries for the wide range of tasks	K_U16
Student can read and write algorithm description and technical documentation in English	K_U20

Social competences

Student shows independent thinking and actions when solving deep learning problems, is aware of the possibility of making mistakes by himself and others, shows criticism of the obtained results	K_K06

16. Literatura obowiązkowa i zalecana

- Goodfellow, I., Bengio, Y. and Courville, A., 2016. Deep learning. MIT Press.
- Dan Jurafsky and James H. Martin, Speech and Language Processing (3rd ed. draft)

17. Metody weryfikacji zakładanych efektów uczenia się

final exam, graded assignment lists, final project

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Lecture

- final exam

Laboratories

- assignment lists (about 5)
- larger programming project implementation and presentation (such as applying deep learning to new data and network improvements; other project topics are possible after instructor's approval)

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

work on the project	30 godz.
exam	2 godz.
preparation of a report for the final project	5 godz.
studying for the final exam	10 godz.
preparation for lab assignments	30 godz.
studying for the lecture and reading literature	10 godz.
preparation for written assignments	15 godz.

Sumarycznie

Łączna liczba godzin	162 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Optymalizacja kombinatoryczna** **Combinatorial optimization**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DZ8OK**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zaliczenie przedmiotów:** • Matematyka dyskretna • Algorytmy i struktury danych
13.	Cele kształcenia dla przedmiotu Podstawowym celem przedmiotu jest zapoznanie studentów z głównymi problemami optymalizacji kombinatorycznej dotyczącymi skojarzeń, matroidów oraz przepływów. W ramach przedmiotu omawiana jest zarówno matematyczna struktura tych problemów, jak i analizowane są algorytmy je rozwiązujące.
14.	Treści programowe 1. Skojarzenia: zastosowania, własności, algorytmy, twierdzenia dotyczące struktury. W szczególności algorytmy dla wersji ważonych i nieważonych w odpowiednio grafach dwudzielnych i niedwudzielnych. Twierdzenie Edmondsa-Gallai. 2. Skojarzenia z preferencjami jednostronnymi i dwustronnymi (stabilne, rango-maksymalne). 3. Matroidy: zastosowania, powiązania między różnymi wariantami, podstawowe pojęcia teoretyczne. Algorytm dla intersekcji dwóch matroidów. 4. Przepływy i rozłączne ścieżki: własności i algorytmy.
15.	Zakładane efekty uczenia się **Wiedza** zna główne problemy optymalizacji kombinatorycznej oraz podstawowe algorytmy dla nich — K_W14 **Umiejętności** potrafi zastosować algorytmy optymalizacji kombinatorycznej — K_U18

16.	Literatura obowiązkowa i zalecana • Schrijver "Combinatorial Optimization" Papadimitriou, Steiglitz "Combinatorial Optimization" Lawler "Combinatorial Optimization. Networks and matroids."
17.	Metody weryfikacji zakładanych efektów uczenia się egzamin prezentacja rozwiązań zadań na tablicy
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Egzamin ma formę pisemną. Aby go zaliczyć, trzeba zdobyć wymaganą liczbę punktów. W czasie zajęć studenci prezentują rozwiązania zadań z list. Uzyskane w ten sposób punkty są podstawą do zaliczenia ćwiczeń.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	10 godz.
przygotowanie do ćwiczeń	40 godz.
studiowanie tematyki wykładów i literatury	40 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Optymalizacja kombinatoryczna w warunkach niepewności** **Combinatorial Optimization under uncertainty**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zaliczenie przedmiotów:** • Rachunek prawdopodobieństwa • Algorytmy i struktury danych **Niezbędne kompetencje:** • biegłość w dyskretnym rachunku prawdopodobieństwa
13.	Cele kształcenia dla przedmiotu Podstawowym celem jest zapoznanie studentów z paradygmatem konstrukcji algorytmów, które działają na losowych danych.
14.	Treści programowe 1. Modelowanie procesów losowych za pomocą programów liniowych. 2. Konstrukcja i analiza dyskretnych procesów losowych. 3. Zapoznanie studentów z 3 głównymi modelami niepewności — model bayesowski, model sekretarki, model online.

15.	Zakładane efekty uczenia się

Wiedza

rozumie, czym zajmuje się Optymalizacja Kombinatoryczna w warunkach niepewności, rozumie również, na czym polega specyficzność metod tej dziedziny	K_W14
posiada przeglądową wiedzę o różnych klasach problemów z dziedziny Optymalizacji Kombinatorycznej w warunkach niepewności	K_W14
zna różne metody modelowania świata, z uwzględnieniem niepewności	K_W13
zna algorytmy i metody ich analizy dla problemu stochastycznego próbkowania	K_W13
zna algorytmy i metody ich analizy dla problemów sekretarkowych	K_W14
zna algorytmy i metody ich analizy dla problemów online takich jak RANKING dla problemu skojarzeń dwudzielnych online	K_W14

Umiejętności

umie modelować różne zagadnienia jako program liniowy lub wypukły/wklęsły	K_U18
umie stosować i modyfikować różne algorytmy zaokrąglania programów liniowych na potrzeby zaprojektowania odpowiedniego algorytmu stochastycznego	K_U18, K_U19
umie modelować niepewność świata za pomocą narzędzi z rachunku prawdopodobieństwa	K_U11, K_U12

16. Literatura obowiązkowa i zalecana

- każdy wykład będzie miał dedykowane notatki i odnośniki do literatury naukowej

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin ustny, sporządzenie jakościowych notatek z wykładu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Egzamin ma formę ustną. Na egzaminie student będzie musiał opisać jeden z kluczowych algorytmów przedstawionych na wykładzie wraz z jego pełną analizą.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

udział w egzaminie	2 godz.
przygotowanie do egzaminu	10 godz.
przygotowanie do ćwiczeń	40 godz.
studiowanie tematyki wykładów i literatury	40 godz.

Sumarycznie

Łączna liczba godzin	152 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Programowanie obiektowe** **Object-oriented programming**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6PO**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Wstęp do programowania lub Algorytmy i struktury danych **Niezbędne kompetencje** • Podstawowa umiejętność programowania w języku wyższego poziomu
13.	Cele kształcenia dla przedmiotu Głównym celem przedmiotu jest zapoznanie studentów z fundamentami programowania obiektowego: klasa, obiekt, dziedziczenie, metody wirtualne, ukrywanie implementacji czy wzorce projektowe, oraz praktyczne ich wykorzystanie w programowaniu. W takcie zajęć będą też omówione metody projektowania oprogramowania wykorzystujące techniki obiektowe.
14.	Treści programowe • Różne style programowania • Klasy i obiekty, konstruktory, dziedziczenie • Metody wirtualne, przeciążanie metod • Programowanie generyczne • Dziedziczenie a implementacja interfejsu • Wybrane wzorce projektowe: singleton, template, strategy • Modelowanie obiektowe za pomocą notacji UML • Analiza i projektowanie obiektowe • Paradygmaty programowania wywodzące się z programowania obiektowego • Obiekty rozproszone i trwałość obiektów • Dobre praktyki programowania obiektowego: zasada podstawienia Liskov, zasada pojedynczej odpowiedzialności

15. Zakładane efekty uczenia się

Wiedza

zna i rozumie podstawowe mechanizmy obiektowe (dziedziczenie, enkapsulacja, abstrakcja, metody wirtualne)	K_W04
zna podstawy UML w zakresie diagramów klas	K_W15
zna przynajmniej 2 języki obiektowe w stopniu podstawowym	K_W15
zna podstawy projektowania obiektowego	K_W15

Umiejętności

umie korzystać z obiektowych klas bibliotecznych i tworzyć własne klasy biblioteczne	K_U03
potrafi określić i korzystać z podstawowych zależności między obiektami (wzorce projektowe)	K_U03, K_U19
umie zaprojektować obiektowo aplikację	K_U17
umie opracować i przedstawić swój projekt	K_U17

16. Literatura obowiązkowa i zalecana

- M. Weisfeld; Myślenie obiektowe w programowaniu
- H. Ledgard; Mała księga programowania obiektowego
- Peter Coad, Edward Yourdon; Object-oriented analysis
- B. Meyer; Programowanie zorientowane obiektowo
- I. Graham; Metody obiektowe w teorii i praktyce

17. Metody weryfikacji zakładanych efektów uczenia się

zaprogramowanie i prezentacja programu komputerowego, implementacja i prezentacja większego projektu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia ćwiczenio-pracowni należy zdobyć wskazaną w regulaminie liczbę punktów za implementację zadań.
Do zaliczenia egzaminu konieczna jest realizacja projektu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

samodzielna implementacja zadań: godz	35 godz.
prezentacja projektu	1 godz.
studiowanie dokumentacji i literatury dodatkowej	20 godz.
opracowanie i implementacja projektu: godz	30 godz.

Sumarycznie

Łączna liczba godzin	146 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Przetwarzanie Obrazów / Digital Image Processing** **Digital Image Processing**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DI2ImgProcP**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – zastosowania informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Completed courses:** • Algebra • Calculus **Required competences:** • programming skills: preferred language C/C++ • some knowledge of Computer Graphics
13.	Cele kształcenia dla przedmiotu The aim of the course is to teach students the methods used in today's image editors in order to better understand how they work as well as their limits. When we need to do something that exceeds the capacity of these tools we have to learn how to reimplement and modify classical methods to get required results. Knowledge of both classical image processing methods and a new field of Computational Photography should enable us to develop and implement better new algorithms.
14.	Treści programowe 1. Color Science and High Dynamic Range Imaging 2. Per pixel operations, histograms, combining images 3. Geometrical operations eg. distortion correction 4. Linear filtering and Fourier transforms 5. Edge detection 6. Nonlinear filters eg. median, bilateral 7. Morphological operators: dylation and erosion 8. Multiresolution representations 9. Keypoint detection and image registration 10. Image segmentation, graphcuts and texture synthesis 11. Gradient domain image processing

15. Zakładane efekty uczenia się

Knowledge

knows color and high dynamic range image representations	K_W15
knows selected API for image processing pipeline	K_W15
knows how to use Fourier Transform to construct linear filters and analyse their properties	K_W14, K_W15
knows the problems of geometrical operations caused by sampling	K_W14, K_W15
knows the methods for image registration and alignment	K_W15
knows selected segmentation algorithms including graphcut method	K_W14
knows gradient domain image processing method and its applications	K_W15

Skills

is able to implement image processing pipeline with image loading, displaying, modifying and saving using selected API	K_U16
is able to efficiently implement linear filtering using FFT	K_U13, K_U16
is able to make analysis of linear filter to avoid aliasing artifacts	K_U19
is able to use provided implementation of graphcut method to solve and implement new tasks	K_U18, K_U19
is able to understand and implement publications from conferences which extend presented algorithms	K_U15, K_U20

16. Literatura obowiązkowa i zalecana

- B.Jahne - "Digital Image Processing", 5th edition, Springer-Verlag 2002.
- Richard Szeliski, Computer Vision: Algorithms and Applications, 2010 available online: http://szeliski.org/Book/

17. Metody weryfikacji zakładanych efektów uczenia się

- programming assignment
- exercises with presentation of solutions on the blackboard
- programming project
- exam

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

To pass the excercises-lab the student has to get the sum of points stated in the rules of the subject. The points sum up from excercises, programming tasks and programming project. Final grade from the subject comes from the exam in which excercise-lab grade can be taken into account according to the rules of the subject.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenio-pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

exam	2 godz.
programming project work	20 godz.
preparation for labs	30 godz.
studing topics from lectures and literature	30 godz.
preparation for exam	10 godz.

Sumarycznie

Łączna liczba godzin	152 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Rachunek prawdopodobieństwa dla informatyków** **Probability Theory for Computer Scientists**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP7RPI**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 45 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Algebra • Analiza matematyczna
13.	Cele kształcenia dla przedmiotu Celem przedmioty jest zapoznanie studentów z podstawowymi pojęciami i technikami probabilistycznymi. Wiele z nich posiada zastosowania w analizie i konstruowaniu współczesnych algorytmów, co będzie ilustrowane licznymi przykładami zastosowania rachunku prawdopodobieństwa do zagadnień typowo informatycznych. Wykład dedykowany jest studentom informatyki, którzy chcieliby poznać arkana rachunku prawdopodobieństwa, by stosować je w analizie i konstruowaniu algorytmów grafowych, programowaniu liniowym, geometrii obliczeniowej, algorytmów teorio-liczbowych.

14. Treści programowe

- Pierwsza część wykładu poświęcona będzie klasycznemu pojęciu prawdopodobieństwa oraz pojęciu i własnościom zmiennych losowych.
- W drugiej części wykładu zajmiemy się nieco bardziej zaawansowanymi zagadnieniami, jak twierdzenia graniczne, procesy Markowa.
- Wprowadzone pojęcia ilustrowane będą przykładami zastosowania w informatyce, na przykład w konstruowaniu efektywnych algorytmów sortowania, problemów kolekcjonera, lematu Lovasza.

Plan wykładu

1. pojęcie prawdopodobieństwa
2. prawo włączeń-wyłączeń
3. podstawowe schematy kombinatoryczne
4. prawdopodobieństwo klasyczne i geometryczne
5. niezależność zdarzeń losowych
6. prawdopodobieństwo warunkowe i całkowite
7. pojęcie zmiennej losowej i jej rozkładu
8. podstawowe rozkłady probabilistyczne
9. wartość oczekiwana, wariancja, kowariancja
10. tw. Markowa, tw. Czebyszewa, tw. Chernoffa
11. twierdzenia graniczne
12. estymatory
13. przedziały ufności
14. łańcuchy Markowa
15. rozkłady warunkowe
16. metody probabilistyczne (lemat Lovasza).

15. Zakładane efekty uczenia się

Wiedza

zna pojęcie prawdopodobieństwa i potrafi stosować formuły kombinatoryczne do jego wyliczenia	K_W13
zna pojęcie prawdopodobieństwa dyskretnego i ciągłego oraz pojęcie wartości oczekiwanej	K_W13
zna pojęcie zmiennej losowej, potrafi liczyć wartość oczekiwaną i wariancję, zna podstawowe rozkłady probabilistyczne	K_W13
zna centralne twierdzenie graniczne i twierdzenie Poissona oraz nierówności Czebyszewa, Markowa i Chernoffa	K_W13
zna pojęcie procesu stochastycznego, próbkowania i estymacji	K_W13

Umiejętności

potrafi definiować i analizować zdarzenia i zmienne losowe oraz obliczać ich prawdopodobieństwo, wartości oczekiwane i odchylenie standardowe	K_U11
umie stosować centralne twierdzenie graniczne i twierdzenie Poissona oraz nierówności Czebyszewa, Markowa i Chernoffa	K_U11
potrafi stosować twierdzenia graniczne oraz stosować oszacowania szukanych prawdopodobieństw za pomocą nierówności Markowa, Czebyszewa, Chernoffa	K_U11
potrafi stosować podstawowe narzędzia statystyczne, w tym skonstruować i zanalizować przedziały ufności dla średniej, zna pojęcie procesu stochastycznego	K_U11

16. Literatura obowiązkowa i zalecana

- Miztenmacher, Upfal “Probability and Computing”
- Feller, Wstęp do Rachunku Prawdopodobieństwa”, Tom 1.

17. Metody weryfikacji zakładanych efektów uczenia się

Egzamin, zadania domowe referowane na zajęciach, Kolokwium

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Egzamin, zadania domowe referowane na zajęciach, Kolokwium

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	45 godz.
ćwiczenia	30 godz.

Praca własna studenta

samodzielne studiowanie tematyki wykładów i literatury	20 godz.
egzamin	3 godz.
przygotowanie do ćwiczeń (konwersatorium)	35 godz.
przygotowanie do sprawdzianów i egzaminu	20 godz.

Sumarycznie

Łączna liczba godzin	153 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Rachunek prawdopodobieństwa i statystyka** **Probability and Statistics**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6RPS**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Algebra • Analiza matematyczna **Niezbędne kompetencje:** • Wstępna znajomość rachunku całkowego • Znajomość operacji macierzowych
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest: • Powiązanie intuicyjnego pojęcia prawdopodobieństwa z formalnymi określeniami. • Zapoznanie słuchaczy z typowymi rozkładami probabilistycznymi. • Część zasadnicza - przejście od próbki pochodzącej ze znanego rozkładu do statystyk decyzyjnych. • Podstawowe informacje o analizie wariancji i regresji liniowej, związki między tymi sposobami analizowania danych.

14. Treści programowe

1. Przestrzeń probabilistyczna, pojęcie zmiennej losowej, jej rozkład i opis.
2. Przykłady zmiennych losowych typu dyskretnego i typu ciągłego. Charakterystyki zmiennych losowych - momenty.
3. Rozkłady zmiennych losowych wielowymiarowych (rozkład dwuwymiarowy, rozkład warunkowy, rozkład brzegowy, niezależność dwóch zmiennych losowych); Macierze kowariancji i korelacji. Wielowymiarowy rozkład normalny i szczególny przypadek dwuwymiarowy.
4. Funkcje dwuwymiarowych zmiennych losowych. Wyznaczanie gęstości i dystrybuanty funkcji zmiennych losowych.
5. Funkcja charakterystyczna i jej własności. Związek funkcji charakterystycznej z momentami zmiennej.
6. Populacja i próbka. Rozkłady którym podlega próbka (chi-kwadrat, t-Studenta, F). Centralne twierdzenie graniczne.
7. Wielowymiarowy rozkład normalny.
8. Regresja liniowa i analiza wariancji - wstęp.

15. Zakładane efekty uczenia się

Wiedza

rozumie pojęcia zmiennej losowej i przestrzeni probabilistycznej	K_W13
zna podstawowe charakterystyki opisowe zmiennej losowej	K_W13
rozumie pojęcie niezależności zmiennych losowych i potrafi połączyć to z rozkładem łącznym (wielowymiarowym)	K_W13
zna pojęcie funkcji tworzącej momenty	K_W13
rozumie zasady testowania hipotez, zna pojęcia statystyki testowej, obszaru krytycznego	K_W13, K_W16
posiada przeglądową wiedzę o różnych rodzajach regresji oraz o analizie wariancji	K_W13, K_W14
rozumie znaczenie obliczeń statystycznych w wymiarze ekonomicznym i kulturowym	K_W16

Umiejętności

potrafi obliczyć wielkości opisowe zmiennej losowej: wartość oczekiwaną, wariancję, korelację	K_U11
umie wyznaczyć rozkład łączny i funkcje zmiennych losowych	K_U11
potrafi wykorzystać funkcją tworzącą momenty do identyfikacji rozkładu	K_U11
zna typowe testy parametryczne i rozumie ich konstrukcję	K_U11
posiada wiedzę o związku regresji z aproksymacją średniokwadratową	K_U11
umie przeprowadzić typowe rodzaje analizy wariancji	K_U11
rozumie związek analizy wariancji z regesją liniową	K_U11

Kompetencje społeczne

jest świadomy zagrożeń wynikających z gromadzenia danych	K_K08

16. Literatura obowiązkowa i zalecana

1. Tadeusz Gernstenkorn, Tadeusz Śródka: Kombinatoryka i rachunek prawdopodobieństwa. PWN, 1974.
2. Małgorzata Majsnerowska: Wprowadzenie do rachunku prawdopodobieństwa z zadaniami.Wydawnictwo BTC, 2009.
3. Mieczysław Sobczyk: Statystyka. Wydawnictwo Uniwersytetu Marii Curie-Skłodowskiej, 1994.
4. Jacek Jakubowski, Rafał Sztencel: Wstęp do teorii prawdopodobieństwa. Script, 2001.

17. Metody weryfikacji zakładanych efektów uczenia się

- Listy zadań - około 10 zadań tygodniowo.
- Zadania do obliczeń komputerowych.
- Kolokwium.
- Egzamin.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Zaliczenie wykładu - egzamin, zadania obliczeniowe w trakcie semestru.
Zaliczenie ćwiczeń - listy zadań, kolokwium w trakcie semestru.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do ćwiczeń	45 godz.
praca nad zadaniami obliczeniowymi	5 godz.
przygotowanie do egzaminu	15 godz.
kolokwium	3 godz.
przygotowanie do kolokwium	10 godz.
czytanie dodatkowych materiałów	10 godz.
egzamin	3 godz.

Sumarycznie

Łączna liczba godzin	151 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Systemy komputerowe** **Computer Systems**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP14ASKSO**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Logika dla informatyków • Wstęp do informatyki lub Algorytmy i struktury danych Pomocna będzie elementarna znajomość języka C.
13.	Cele kształcenia dla przedmiotu Celem wykładu jest przedstawienie studentom najważniejszych algorytmów i struktur danych wykorzystywanych w systemach komputerowych, głównie w sprzęcie, systemach operacyjnych i narzędziach systemowych.

14. Treści programowe

1. Sprzętowa implementacja operacji arytmetycznych: szybkie algorytmy dodawania (sumatory carry-lookahead i Ling'a), mnożenia (drzewa Wallace'a i kodowanie Booth'a) i dzielenia (non-restoring division, SRT division).
2. Niskopoziomowa reprezentacja programów (kod trójadresowy, graf przepływu sterowania), analiza przepływu danych (obliczenia stałopunktowe).
3. Architektura współczesnego procesora (przetwarzanie potokowe i out-of-order). Graf przepływu danych i algorytm Tomasulo.
4. Spekulatywne wykonywanie instrukcji, przewidywanie skoków (perceptron, TAGE), atak Spectre.
5. Instrumentacja kodu, profilowanie (graf wywołań funkcji) i optymalizacja kodu (algorytm local value numbering).
6. Struktury pamięci podręcznych, algorytmy cache-oblivious.
7. Algorytmy szeregowania wątków i balansowania obciążenia procesorów (algorytm ULE, kolejka kalendarzowa).
8. Translacja adresów, wielopoziomowa i odwrócona tablica stron.
9. Pamięć wirtualna, algorytmy zastępowania stron (algorytm WSCLOCK, aging, Clock-Pro).
10. Algorytmy dynamicznego przydziału pamięci (buddy systems, boundary tags). Algorytmy odśmiecania (mark and sweep, algorytm kopiujący).
11. Struktury danych systemów plików (grupy bloków, i-node, B-drzewo, journaling). Planowanie operacji I/O (budget-fair queueing).
12. Przetwarzanie współbieżne. Implementacja środków synchronizacji (synchronizacja pamięci podręcznych, exponential backoff).
13. Kontrola przepływności w protokole TCP. Obliczanie sum kontrolnych i kodów korekcyjnych.
14. Sieci w układach scalonych: topologia i trasowanie pakietów.

15. Zakładane efekty uczenia się

Wiedza

Wie jak programy komunikują się ze sprzętem i systemem operacyjnym	K_W08
Rozumie wpływ architektury maszyny na implementację kompilatorów i systemów operacyjnych	K_W08, K_W15
Posiada przeglądową wiedzę na temat komponentów systemu operacyjnego.	K_W15
Zna mechanizmy, polityki i algorytmy, które wykorzystują jądra systemów operacyjnych w celu realizacji usług dla programów użytkownika.	K_W15

Umiejętności

Umie pisać programy zachowujące się poprawnie w obliczu ograniczeń maszynowej reprezentacji typów numerycznych	K_U16
Potrafi wybrać i zastosować właściwe narzędzia i metodologie do mierzenia wydajności kodu programów	K_U19
Potrafi przeprowadzić analizę kodu pod względem wykorzystania pamięci podręcznej i przeorganizować go celem optymalizacji	K_U06
Potrafi użyć środków synchronizacji do konstrukcji poprawnych programów współbieżnych.	K_U03
Pomaga członkom zespołu w pisaniu oprogramowania wolnego od usterek związanych z niskopoziomowymi aspektami działania komputerów	K_U23

Kompetencje społeczne

Potrafi właściwie wskazać te części projektu komercyjnego, które opłaca się optymalizować	K_K07

16. Literatura obowiązkowa i zalecana

Wybrane przez wykładowcę fragmenty poniższych publikacji:

- Hennessy, John L., and David A. Patterson. Computer Architecture: A Quantitative Approach (6th ed.).
- Israel Koren, Computer Arithmetic Algorithms (2nd ed.).
- Flemming Nielson, Hanne R. Nielson, and Chris Hankin. Principles of Program Analysis.
- D. Harris, S. Harris, Digital Design and Computer Architecture (2nd ed.)
- Randal Bryant, David O'Hallaron. Computer Systems: A Programmer's Perspective (3rd ed.).
- Bruce Jacob, Spencer Ng, and David Wang. Memory Systems: Cache, DRAM, Disk (2nd ed.)
- Abraham Silberschatz, Peter Baer Galvin, Greg Gagne. Operating System Concepts (10th ed.)
- Maurice J. Bach. The design of the UNIX operating system.

17.	Metody weryfikacji zakładanych efektów uczenia się prezentacja rozwiązań zadań, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Warunkiem zaliczenia ćwiczeń jest uzyskanie minimum punktowego za przygotowane rozwiązania zadań (na podstawie deklaracji weryfikowanych w trakcie ćwiczeń). Aby zaliczyć wykład, należy zdobyć co najmniej połowę punktów z końcowego egzaminu pisemnego. Szczegółowe warunki zaliczenia przedmiotu są określone w regulaminie przedmiotu.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	25 godz.
egzamin	2 godz.
rozwiązywania zadań z list	30 godz.
analiza i weryfikacja notatek z zajęć oraz innych źródeł informacji	30 godz.

Sumarycznie

Łączna liczba godzin	147 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Systemy wbudowane** **Embedded systems**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP11SW**
6.	Rodzaj przedmiotu/modułu **Informatyczny inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Wstęp do informatyki • Wskazane: systemy operacyjne • Wskazane: logika cyfrowa **Niezbędne kompetencje:** • Umiejętność programowania w języku C • Wskazane: znajomość podstaw elektroniki
13.	Cele kształcenia dla przedmiotu Celem wykładu jest nauczenie podstaw programowania mikrokontrolerów oraz projektowania układów elektronicznych z mikrokontrolerami. Na zajęciach student pozna, w jaki sposób wykorzystać peryferia mikrokontrolerów do sterowania różnorodnymi urządzeniami zewnętrznymi.

14.	Treści programowe • Wprowadzenie do systemów wbudowanych • Podstawy architektury AVR • Podstawy systemu FreeRTOS • Prototypowanie układów na płytce stykowej • Projektowanie układów elektronicznych z mikrokontrolerem • Obsługa portów GPIO mikrokontrolera • Przetworniki analogowo-cyfrowe • Fala prostokątna, modulacja PWM • Obsługa przerwań na mikrokontrolerach • Zarządzanie energią • Protokoły komunikacyjne UART, SPI, I2C • Elementy przełączające • Podstawy wzmacniaczy • Sterowanie silnikami • Podstawy teorii sterowania
15.	Zakładane efekty uczenia się **Wiedza** Rozumie na podstawowym poziomie działanie najważniejszych rodzajów elementów elektronicznych — K_W15 Posiada przeglądową wiedzę na temat obwodów peryferyjnych występujących w popularnych mikrokontrolerach: liczników, generatorów PWM, transceiverów, przetworników ADC — K_W08, K_W15 **Umiejętności** Potrafi konstruować prototypy układów elektronicznych z mikrokontrolerem na podstawie schematu przy użyciu płytki prototypowej — K_U16, K_U17 Potrafi programować mikrokontrolery AVR ATmega przy użyciu języka C oraz wykorzystywać system czasu rzeczywistego FreeRTOS — K_U16, K_U17 Potrafi przeczytać notę katalogową układu elektronicznego i na jej podstawie zaimplementować sterownik tego układu — K_U17 Potrafi dobrać właściwą metodę komunikacji (np. protokół UART, SPI lub I2C, sygnał analogowy, sygnał PWM) pomiędzy mikrokontrolerem a urządzeniem lub innym mikrokontrolerem — K_U03 **Kompetencje społeczne** Rozumie znaczenie konwencji stosowanych w schematach elektronicznych dla komunikacji międzyludzkiej — K_K05
16.	Literatura obowiązkowa i zalecana **Literatura obowiązkowa:** • Make: AVR programming; Elliot Williams; Maker Media, Inc. • Designing Embedded Hardware; John Catsoulis; O'Reilly [rozdziały 4, 6-9, 13, 15] **Literatura zalecana:** • Lessons in Electric Circuits; Tony R. Kuphaldt
17.	Metody weryfikacji zakładanych efektów uczenia się budowa prototypu układu elektronicznego z mikrokontrolerem, oprogramowanie mikrokontrolera, egzamin pisemny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za zadania. Zadania są rozwiązywane w domu, ich rozwiązania (złożone z prototypu układu i programu na mikrokontroler) prezentowane są na pracowni. Egzamin ma formę pisemną, do jego zaliczenia konieczne jest zdobycie odpowiedniej liczby punktów. Osoby, które osiągnęły bardzo dobre wyniki na ćwiczeniach mogą uzyskać zwolnienie z egzaminu.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
wykład	30 godz.

Praca własna studenta

studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do pracowni	50 godz.
udział w egzaminie	3 godz.
przygotowanie do egzaminu	7 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Teoria informacji i teoria kodowania** **Information and Coding Theory**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DI2InfTh**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Algebra • Analiza matematyczna • Rachunek prawdopodobieństwa i statystyka (lub podobny przedmiot dający biegłość w posługiwaniu się pojęciami zmiennej losowej, wartości oczekiwanej itp.) • Mile widziane, choć nie wymagane: Algorytmy i struktury danych • Mile widziane, choć nie wymagane: Matematyka dyskretna **Niezbędne kompetencje:** • Znajomość podstawowych pojęć z rachunku prawdopodobieństwa i statystyki (zmienna losowa, wartość oczekiwana, niezależność) i umiejętność posługiwania się nimi. • Znajomość pojęć algebry liniowej (wektor, macierz, mnożenie macierzy, wartości i wektory własne).
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zaznajomienie studentów z podstawowymi pojęciami teorii informacji. Ponadto ćwiczenia powinny nauczyć studentów swobodnego operowana tymi pojęciami.

14. Treści programowe

1. Miary informacji: entropia, dywergencja, informacja wspólna, łańcuchy Markowa, nierówność przetwarzania danych
2. Kompresja bezstratna: kody o zmiennej długości.
3. Kodowanie uniwersalne, przykłady: LZ77, kompresja gramatykowa.
4. Kompresja stratna: kwantyzacja, kompresja dźwięku i obrazu, transformata Karhunen–Loève, dyskretna transformata kosinusowa (DCT)
5. Kanały komunikacji: dyskretne, ciągłe.
6. Przepustowość kanałów komunikacji: twierdzenie Shannona.
7. Kody korekcyjne, kody polarne.
8. Inne zastosowania teorii informacji: uczenie maszynowe, generatory pseudolosowe.

15. Zakładane efekty uczenia się

Wiedza

Zna podstawowe pojęcia i twierdzenia teorii informacji	K_W14
Zna twierdzenia teorii kodowania mówiące o ograniczeniach górnych algorytmów kompresji bezstratnej i stratnej	K_W14
Zna zastosowania teorii informacji w innych działach matematyki i informatyki	K_W14

Umiejętności

Umie posługiwać się twierdzeniami teorii informacji i kodowania w celach szacowania rozmiaru wyjścia algorytmów kompresji i szacowania średniej informacji generowanej przez źródła stochastyczne	K_U19
Umie wykorzystać teorię informacji do szacowania dolnych granic	K_U14
Potrafi modelować problemy przy użyciu pojęć teorii informacji	K_U13

16. Literatura obowiązkowa i zalecana

- Thomas M. Cover, Joy A. Thomas, Elements of Information Theory. Wiley Series in Telecommunications, 1991.
- Gareth A. Jones, Mary J. Jones, Information and Coding Theory. Springer, 2000.
- David Salomon, Data Compression: The Complete Reference, Springer, 1998

17. Metody weryfikacji zakładanych efektów uczenia się

- ćwiczenia: prezentacja zadań przy tablicy
- egzamin końcowy (w zależności od liczby słuchaczy: pisemny lub ustny)

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Zaliczenie ćwiczeń wiąże się z uzyskaniem wymaganej liczby punktów, przyznanej za prezentacje zadań przy tablicy.
Egzamin jest pisemny. Zaliczenie go wiąże się z uzyskaniem wymaganej liczby punktów.
W przypadku małej liczby uczestników przeprowadzony zostanie egzamin ustny. Zaliczenie go polega na przedstawieniu wybranych (przez egzaminatora) zagadnień związanych z treściami przedmiotu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

udział w egzaminie	3 godz.
przygotowanie do ćwiczeń	60 godz.
przygotowanie do egzaminu	20 godz.

Sumarycznie

Łączna liczba godzin	143 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Testowanie gier** **Game Testing**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-TestGier**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zaliczone przedmioty:** zaliczony kurs jednego z języków programowania **Niezbędne kompetencje:** elementarna znajomość języka C++, Java. Większość zadań programistycznych wymaga podstawowej znajomości Javy.
13.	Cele kształcenia dla przedmiotu Celem przedmiotu Testowanie gier jest pomoc w doskonaleniu umiejętności wystarczająco dobrego testowania gier komputerowych.

14. Treści programowe

Wykłady

- Podstawy testowania gier
- Poziomy i typy testów
- Dwie podstawowe zasady testowania gier
- Rola testera gier
- Znaczenie wystarczająco dobrego testowania gry
- Jakość gry komputerowej
- Plan testowania gry komputerowej
- Proces testowania gry
- Testing by the Numbers, Combinational Testing
- Test Flow Diagrams
- Cleanroom Testing, Test Tree
- Ad Hoc Testing oraz Gameplay testing
- Defect Triggers
- Regeression testing and Test Reuse
- Exploratory Game Testing

Ćwiczenio-pracowania

- Omówienie zakresu i warunków zaliczenia ćwieczenio-pracowni
- Metody opisu wymagań funkcjonalnych
- Metody opisu wymagań niefunkcjonalnych
- Definiowanie i planowanie przypadków testowych
- Tworzenie opisu procesu testowego
- Testowanie bezpieczeństwa i ochrony danych
- Ocena zgodności interfejsu użytkownika z 10-ciom heurystykami Nielsena
- Automatyczne testowanie interfejsu użytkownika
- Testowanie jednostkowe
- Techniki i narzędzia Mockup
- Testowanie integracyjne
- Testowanie wydajności
- Add Number Method
- Basic Test Plan Template
- Testy kombinacyjne
- Test Flow Digrams (TFD)

15. Zakładane efekty uczenia się

Wiedza

ma poszerzoną, uporządkowaną i podbudowaną teoretycznie wiedzę w obrębie testowania gier	K_W15
zna i rozumie pojęcie specyfikacji gry, poprawność i testowalności wymagań i zasad planowania wykonania testów gry	K_W15, K_W16
ma podstawową wiedzę na temat tworzenia i weryfikacji oraz walidacji przypadków testowych	K_W15
zna podstawowe metody projektowania i analizy podstawy testów	K_W15
zna podstawowe metody planowania procesu testowego gry	K_W15

Umiejętności

potrafi testować testować gry	K_U16, K_U18
umie określić cele testowania gry	K_U19
potrafi definiować i analizować testowalność wymagań gry określonych przez jej specyfikacje	K_U19
analizuje poprawność i złożoność przypadku testowego i procesu testowego	K_U19
umie stosować podstawowe metody i typy testów gry	K_U19
ma umiejętność używania popularnych automatycznych narzędzia testowania	K_U16

Kompetencje społeczne

jest świadom roli i znaczenia wiedzy na temat testowania oprogramowania, w szczególności gier komputerowych	K_K02

16.	Literatura obowiązkowa i zalecana • strony www w języku angielskim ISTQB (International Software Testing Qualifications Board) Certified Tester Foundation Level (CTFL) i wyższe poziomy • strony www Stowarzyszenia Jakości Systemów Informatycznych • książka: Schultz Ch. P., Bryant R. D.: Game Testing All in One, Third Edition Mercury Learning and information, 2017
17.	Metody weryfikacji zakładanych efektów uczenia się • prezentacja rozwiązań zadań podczas ćwiczeń i w pracowni, • weryfikacja i walidacja zadań przygotowanych i wykonanych przez studentów, • test pisemny i egzamin.
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Aby uzyskać zaliczenie ćwiczeń i pracowni, student musi zebrać określoną minimalną liczbę punktów do rozwiązania zestawów zadań (na podstawie deklaracji zweryfikowanych podczas ćwiczeń/pracowni) oraz uzyskać pozytywne oceny z dwóch kolokwiów. Aby zaliczyć przedmiot, należy zaliczyć ćwiczenia i pracownie i uzyskać co najmniej 60 punktów za końcowy egzamin pisemny.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** ćwiczenio-pracownia: 30 godz. wykład: 30 godz. **Praca własna studenta** testy i egzamin: 6 godz. rozwiązywanie zestawów zadań: 30 godz. przygotowanie się do testu: 4 godz. przygotowanie się do egzaminu: 16 godz. lektura materialu: 25 godz. **Sumarycznie** Łączna liczba godzin: 141 godz. Liczba punktów ECTS: 6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Wstęp do informatyki** **Introduction to computer science**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP6WI**
6.	Rodzaj przedmiotu/modułu **Informatyczny podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Elementarna znajomość języka C lub Python (począwszy od wykładu 5)
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest przedstawienie struktury logicznej i zasad działania komputera, cyfrowej reprezentacji informacji oraz sposobu wykonywania programów przez komputer. Istotnym celem jest też umożliwienie studentom nabycia podstawowych umiejętności z zakresu konstruowania i analizy algorytmów, w tym weryfikacji poprawności i szacowania złożoności obliczeniowej. W ramach przedmiotu umożliwia się poznanie podstawowych struktur danych i ich implementacji, a także nabycie umiejętności stosowania takich struktur danych do rozwiązywanych problemów.

14.	Treści programowe • Pojęcia: problem, specyfikacja, algorytm, program komputerowy. Budowa komputera – schemat logiczny. • Elementy programowania w języku niskopoziomowym na przykładzie maszyny RAM • Reprezentacja danych w pamięci komputera: binarna stałopozycyjna i zmiennopozycyjna reprezentacja liczb, arytmetyka binarna; reprezentacja tekstu, obrazu, dźwięku. • Programowanie w wysokopoziomowym języku strukturalnym, zstępująca metoda programowania. • Kompilacja, translacja, interpretacja programów. Zarządzanie pamięcią podczas uruchamiania programu. • Podstawowe konstrukcje programistyczne: iteracja, zagnieżdżanie, licznik, sumator, rekurencja. • Elementy analizy algorytmów: poprawność, własność stopu, złożoność czasowa i pamięciowa, miary złożoności; notacja asymptotyczna. • Podstawowe struktury danych i ich zastosowania: tablica, lista wiązana, stos, kolejka; drzewo przeszukiwań binarnych. Abstrakcyjne typy danych. • Sortowanie przy pomocy porównań. • Podstawowe techniki algorytmiczne: metoda nawrotów, metoda dziel i zwyciężaj, programowanie dynamiczne, algorytmy zachłanne. • Komputerowa reprezentacja grafów. • Algorytmy grafowe: przeszukiwanie w głąb, wszerz; składowe spójności. • Gramatyki bezkontekstowe jako narzędzie do opisu składni języków programowania. Notacja BNF i EBNF. • Pojęcie rozstrzygalności problemów, przykłady problemów nierozstrzygalnych.
15.	Zakładane efekty uczenia się **Wiedza** zna i rozumie pojęcie specyfikacji problemu algorytmicznego, zgodności algorytmu ze specyfikacją i sposobu wykonywania programów przez komputer — K_W02 posiada podstawową wiedzę na temat cyfrowej reprezentacji danych - liczby, tekst, obrazy, grafy — K_W02 zna podstawowe metody projektowania i analizy algorytmów — K_W02 zna podstawowe struktury danych i wykonywane na nich operacje, a także koncepcję abstrakcyjnych struktur danych — K_W02 **Umiejętności** potrafi formułować precyzyjną specyfikację problemu — K_U02 konstruuje algorytmy rozwiązujące standardowe problemy sformułowane w postaci specyfikacji — K_U17 potrafi analizować poprawność i złożoność obliczeniową prostych algorytmów, w tym rekurencyjnych — K_U06 wykorzystuje podstawowe techniki algorytmiczne, struktury danych i obiekty abstrakcyjne (np. grafy) do rozwiązywania problemów — K_U03 potrafi interpretować i zapisywać elementy składni języków programowania w postaci gramatyk bezkontekstowych — K_U03 posługuje się cyfrową reprezentacją danych: liczb, obrazów, grafów — K_U03
16.	Literatura obowiązkowa i zalecana • Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman; Algorytmy i struktury danych; Helion, [rozdziały 1-3, 5.1 – 5.2, 6, 8.1-8.3, 9, 10] • David Harel, Rzecz o istocie informatyki – algorytmika, WNT, 2008. [rozdziały 1 – 6, częściowo 7 i 8] • A.V. Aho, J.D. Ullman, Wykłady z informatyki z przykładami w języku C, Helion, Gliwice 2003 [rozdziały 1-3, 5, 6, 9.1 – 9.4, 11.1 – 11.4] • Ryuhei Uehara, First Course in Algorithms Through Puzzles, Springer, 2019. [rozdziały 1 - 5, poza 4.4]
17.	Metody weryfikacji zakładanych efektów uczenia się prezentacja rozwiązania zadania, kolokwium, egzamin pisemny

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Warunkiem zaliczenia ćwiczeń jest uzyskanie minimum punktowego łącznie za przygotowane rozwiązania zadań (na podstawie deklaracji weryfikowanych w trakcie ćwiczeń) oraz dwa kolokwia. Aby zaliczyć wykład, należy zdobyć co najmniej połowę punktów z końcowego egzaminu pisemnego. Szczegółowe warunki zaliczenia przedmiotu są określone w regulaminie przedmiotu.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

udział w kolokwiach i egzaminie	5 godz.
analiza i weryfikacja notatek z zajęć oraz innych źródeł informacji	15 godz.
nabycie umiejętności posługiwania się symulatorem maszyny RAM –	5 godz.
przygotowanie do kolokwiów	15 godz.
przygotowanie do egzaminu	20 godz.
rozwiązywania zadań z list	30 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	6

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Złożoność obliczeniowa** **Computational complexity**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DZ6ZO**
6.	Rodzaj przedmiotu/modułu **Informatyczny zaawansowany – teoria informatyki**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Kompetencje równoważne z zaliczeniem przedmiotu Języki Formalne i Złożoność Obliczeniowa.
13.	Cele kształcenia dla przedmiotu Celem wykładu jest przedstawienie kluczowych koncepcji i twierdzeń teorii złożoności obliczeniowej, które powinien znać każdy dobrze wykształcony magister informatyki.

14. Treści programowe

1. Uporządkowanie wiedzy nabytej na przedmiocie Języki Formalne i Automaty.

Następnie przejdziemy do omawiania zagadnień z poniższej listy.

1. Pamięć vs Czas
   o Twierdzenie Hopcrofta, Paula, Valianta: o tym, że pamięć jest cenniejszym zasobem niż czas obliczeń i że można przekonać się o tym grając w kamyki
2. Dopełnienie vs Niedeterminizm
   o Twierdzenie Immermana-Szelepcsenyi'ego: o tym, że niedeterministycznie można nie tylko zgadywać istnienie rozwiązań, ale także ich nieistnienie
3. Determinizm vs Niedeterminizm
   o Twierdzenie Paula, Pippengera, Szemeredi'ego i Trottera: o tym, że umiejętność zgadywania jest istotnie pomocna (przynajmniej wtedy, gdy mamy mało czasu)
4. Struktura zbiorów NP-trudnych
   o Twierdzenie Mahaneya: o tym, że zbiory NP-zupełne nie mogą mieć mało elementów, no chyba, że P=NP (w co nikt nie wierzy)
5. Dowody interakcyjne
   o Twierdzenie Shamira (IP=PSPACE): m.in. o tym, że ograniczony (w mocy obliczeniowej) król Artur może skorzystać podczas gry w szachy z podpowiedzi nieskończenie mądrego Merlina i, co ważniejsze, nie dać mu się oszukać
6. Trudność zliczania
   o Twierdzenie Valianta: o znaczeniu „-1" w definicji wyznacznika macierzy albo inaczej o tym , że obliczanie permanentu macierzy jest nieporównanie trudniejsze od obliczania wyznacznika (no chyba, że P=NP:-)
   o Twierdzenie Tody, o tym, że umiejętność szybkiego zliczania cykli Hamiltona w grafie ma daleko większe konsekwencje niż mogłoby się to wydawać
   o Twierdzenie Valianta, Vazirani'ego o tym, że szukanie igły w stogu siana jest tak samo trudne niezależnie od liczby igieł w stogu (przynajmniej wtedy, gdy igłami są podstawienia spełniające formułę SAT, a stóg tworzą podstawienia niespełniające)
   8. Algorytmy probabilistyczne:
   o Twierdzenie Beigela, Reingolda, Spielmana: o tym jak policzyć przekonanych w więcej niż w 50
7. Złożoność problemu rozpoznawania liczb pierwszych
   o Twierdzenie Pratta (PRIMES $\in$ NP) o tym, że istnieją proste dowody pierwszości liczb,
   o Twierdzenie Agrawala-Kayala-Saxeny (PRIMES $\in$ P) o tym, że warto uczyć się algebry, by dowodów na pierwszość nie trzeba było zgadywać
8. Spójność w grafach
   o Twierdzenie Reingolda: o tym, że łatwiej jest sprawdzać połączenia, gdy drogi są dwukierunkowe
9. Moc probabilistycznej weryfikacji
   o Twierdzenie Arory, Safry o tym, że do nabrania przekonania o spełnialności formuły wystarczy wylosować stałą liczbę bitów z pewnego dowodu.

Większość z powyższych twierdzeń udowodnimy. W niektórych przypadkach będziemy jednak musieli zadowolić się przedstawieniem jedynie idei dowodu.

15. Zakładane efekty uczenia się

Wiedza

Zna kluczowe koncepcje i twierdzenia teorii złożoności obliczeniowej	K_W09, K_W14
Zna i rozumie metody dowodzenia wybranych, kluczowych twierdzeń złożoności obliczeniowej	K_W09, K_W14
Posiada wiedzę na temat klasycznych otwartych problemów teorii złożoności obliczeniowej	K_W09, K_W14
Zna inspiracje teorii złożoności obliczeniowej wyrosłe z praktyki oraz wpływ osiągnięć teorii złożoności obliczeniowej na praktykę	K_W09, K_W14

Umiejętności

Potrafi przedstawiać skomplikowane dowody i konstrukcje matematyczne stosowane w zakresie informatyki teoretycznej	K_U01, K_U13
Umie dotrzeć do materiałów źródłowych i czytać trudne prace naukowe	K_U15, K_U20

16. Literatura obowiązkowa i zalecana

- Sanjeev Arora and Boaz Barak, Computational Complexity. A Modern Approach. Cambridge University Press, 2009.
- Liczne artykuły źródłowe, umieszczone na stronie wykładu.
- Notatki do wykładu.

17.	Metody weryfikacji zakładanych efektów uczenia się Egzamin pisemny i ustny, prezentacja rozwiązania
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Podstawą zaliczenia ćwiczeń jest aktywność studentów na ćwiczeniach, premiowana punktami. Do premiowanej aktywności zalicza się m.in. prezentację rozwiązań zadań z list, dyskusję nad rozwiązaniami, znajdowanie błędów w wywodach innych studentów i prowadzącego ćwiczenia. Egzamin składa się z dwóch części: pisemnej i ustnej. Podczas części pisemnej studenci otrzymują 10 pytań lub bardzo prostych zadań, mających na celu sprawdzenie stopnia opanowania podstawowych pojęć i faktów. Podczas części ustnej sprawdzany jest stopień zrozumienia prezentowanych na wykładzie zagadnień, w tym dowodów prezentowanych twierdzeń. Ze względu na trudność materiału, podczas tej części studenci w czasie (około) półgodzinnego przygotowania do odpowiedzi na wcześniej wylosowane pytania mogą korzystać z dowolnych materiałów (w tym dostarczonych przez wykładowcę, własnych oraz z zasobów internetu).
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	30 godz.
wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu	35 godz.
rozwiązywanie zadań z list	60 godz.
udział w egzaminie	3 godz.
studiowanie notatek i literatury	15 godz.

Sumarycznie

Łączna liczba godzin	173 godz.
Liczba punktów ECTS	6

4 Kursy narzędzi informatycznych

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Algorytmika Olimpijska** **Olympic algorithms**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKAO**
6.	Rodzaj przedmiotu/modułu **Kurs zaawansowany**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenio-pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Niezbędne zaliczone przedmioty: • Wstęp do informatyki Zalecane zaliczone przedmioty: • Algorytmy i struktury danych Niezbędne kompetencje: • elementarna znajomość języka C++ lub Python Zalecane kompetencje: • obycie w startach w zawodach algorytmicznych (np. Olimpiada Informatyczna)
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zaznajomienie studentów ze specyfiką organizacji zawodów algorytmicznych, w szczególności w zakresie wymyślania i opracowywania zadań. Studenci powinni zdawać sobie sprawę z technicznych aspektów i teoretycznych ograniczeń w zakresie oceny rozwiązań zawodników. Umiejętności te mogą pomóc w zaangażowaniu się w organizację zawodów - np. Olimpiady Informatycznej. Drugim celem przedmiotu jest wprowadzenie do działalności dydaktycznej ze szczególnym uwzględnieniem nauki algorytmiki dla uczniów szkół podstawowych i uczniów liceów. W ramach przedmiotu studenci otrzymają informację na temat przykładowych działań, w które mogą się zaangażować po uzyskaniu kompetencji nabytych podczas nauki przedmiotu.

14. Treści programowe

W zakresie przygotowywania zadań algorytmicznych:
- potencjalne błędy w zadaniach algorytmicznych na konkursach algorytmicznych, analiza konkretnych przypadków archiwalnych,
- redagowanie treści zadań algorytmicznych,
- proces przygotowywania programów wzorcowych, częściowych i błędnych na potrzeby systemu online judge,
- przygotowywanie zestawu testów (danych wejściowych) do oceny rozwiązań zawodników,
- przygotowywanie narzędzi pomocnicznych na potrzeby oceny rozwiązań (np. weryfikatory poprawności odpowiedzi, generatory testów, walidatory wejść),
- aspekty techniczne sprawdzania rozwiązań, zasada działania systemu online judge, ograniczenia teoretyczne takich systemów,
- inspiracje dla tworzenia własnych problemów algorytmicznych.

W zakresie dydaktyki informatyki:
- przegląd sposobów rozpoczęcia pracy z uczniami bez umiejętności programowania,
- zadania algorytmiczne bez komputera, elementy myślenia komputacyjnego,
- podstawy maszyny RAM i sposoby nauczania uczniów programowania w maszynie RAM,
- dydaktyka programowania,
- zajęcia pozalekcyjne: obozy, sparingi, konkursy, prelekcje, organizacja i sposoby angażowania uczniów w działania dodatkowe,
- ocenianie uczniów,
- przykładowe wzorcowe prezentacje zagadnień informatycznych,
- przygotowanie notatek i materiałów dodatkowych dla uczniów.

15. Zakładane efekty uczenia się

Wiedza

posiada wiedzę na temat elementów niezbędnych do przygotowania zadania na zawody algorytmiczne	K_W07, K_W08
zna dostępne narzędzia, w których można przygotowywać zadania	K_W15
zna pojęcia związane z maszyną RAM	K_W02

Umiejętności

potrafi przygotować komplet materiałów zadania algorytmicznego do użycia w systemie online judge	K_U17
potrafi przygotować treść zadania w LaTeX i Markdown, również w języku angielskim	K_U20
potrafi zaprezentować trudne zagadnienia informatyczne w sposób interesujący i przekonujący dla szerokiej grupy osób, również tych bez wiedzy i doświadczenia	K_U21
potrafi przygotować materiały edukacyjne wysokiej jakości	K_U21
potrafi zauważać problemy algorytmiczne w swoim otoczeniu i konkretyzować je do pomysłów nadających się na zadania na konkurs	K_U22

Kompetencje społeczne

potrafi opowiadać o problemach algorytmicznych w sposób zoptymalizowany na jak najlepsze zrozumienie treści przez słuchaczy	K_K05
zna i opracowuje własne konkretne metody dydaktyczne pomocne w pracy z uczniami na różnym poziomie zaawansowania i o różnych potrzebach	K_K04

16. Literatura obowiązkowa i zalecana

Cormen Thomas H., Leiserson Charles E., Rivest Ronald L, Clifford Stein: Wprowadzenie do algorytmów

17. Metody weryfikacji zakładanych efektów uczenia się

- prezentacja rozwiązania zadania,
- opracowanie i przedstawienie prezentacji na zadany temat,
- napisanie programu komputerowego,
- realizacja zadań przy komputerze.

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Do zaliczenia kursu konieczne jest uzyskanie co najmniej 50 maksymalnej liczby punktów. Około 50 maksymalnej liczby punktów można otrzymać na każdych ćwiczeniach lub pracowni za prezentację rozwiązań zadań na listach ćwiczeniowych (w systemie deklaracji) lub prezentację przygotowanych przez siebie programów i materiałów. Około 50 maksymalnej liczby punktów można otrzymać za wykonanie projektu, prezentacji i materiału edukacyjnego. Student wybiera projekt, temat prezentacji i temat przygotowywanego materiału z dostępnej listy. Szczegółowe warunki zaliczenia określone są w regulaminie przedmiotu.
19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
ćwiczenio-pracownia	30 godz.

Praca własna studenta

przygotowanie rozwiązań na pracownię	20 godz.
przygotowanie do ćwiczeń - samodzielne rozwiązywanie zadań i przygotowanie do zaprezentowania rozwiązań	25 godz.
studiowanie notatek z wykładów	5 godz.
praca nad projektem	15 godz.
praca nad prezentacją	10 godz.
praca nad materiałem edukacyjnym	5 godz.

Sumarycznie

Łączna liczba godzin	140 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Introduction to cloud computing** **Introduction to cloud computing**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-ItCC**
6.	Rodzaj przedmiotu/modułu **Kurs podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin pracownia — 15 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu • General technical culture to learn technology independently. • Knowledge of the English language to allow. – Work with documentation and teaching materials, – presentation of own project, – group discussion.
13.	Cele kształcenia dla przedmiotu The primary goal of the course is to familiarize students with the cloud services currently offered on the market and to teach how to use them (at a basic level).
14.	Treści programowe 1. Review of existing cloud technologies and services offered by their providers. 2. Familiarization with selected cloud technologies and services offered in the cloud. 3. Designing and implementation of a cloud-based technology solution project (e.g. application, CD/CI solution, data science project) using selected cloud technologies and services offered in the cloud.

15. Zakładane efekty uczenia się

Knowledge

knows and understands the concept of cloud computing	K_W04, K_W15
knows and understands the differences between the basic ways of using of the cloud - infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS) and software-as-a-service (SaaS)	K_W04, K_W15
knows the basic services offered in the cloud	K_W15
knows where to look for knowledge about cloud services	K_W15

Skills

is able to design and implement a simple technology solution (e.g., application, CD/CI solution, data science project) using selected cloud technologies and services offered in the cloud	K_U16, K_U18
is able to independently learn cloud technologies	K_U19, K_U22
is able to present an idea of own project in an understandable and attractive way for the audience, and after its completion the project itself	K_U21
is able to adequately address comments and criticism on their proposed solutions	K_U21, K_U23
is able to constructively criticize another students' projects	K_U21, K_U23
is able to speak English at the B2 level	K_U20

16. Literatura obowiązkowa i zalecana

Teaching materials offered by cloud service providers

- Google Cloud Skills Boost https://www.cloudskillsboost.google/
- AWS Academy: Academy Cloud Foundations, Academy Learners Lab
- inne materiały dostępne w internecie: aws.amazon.com, cloud.google.com

17. Metody weryfikacji zakładanych efektów uczenia się

presentation of the project idea, presentation of the project, consultation on the progress of the project preparation, participation in the discussion on one's own project and the projects of other students

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Design and implementation of a cloud-based technology solution project (e.g., application, CD/CI solution, data science project) using cloud technologies and services offered in the cloud.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

presentation of own project and discussion on other students' projects	10 godz.
presentation of own project idea and discussion on other students' project ideas	4 godz.

Praca własna studenta

preparation of project proposal and its presentation	2 godz.
participation in introductory classes	1 godz.
study of selected technologies using the provided teaching materials	10 godz.
preparation of the project	16 godz.
preparation of the project presentation	2 godz.
additional consultations as needed	2 godz.

Sumarycznie

Łączna liczba godzin	47 godz.
Liczba punktów ECTS	2

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs administrowania systemem Linux** **Linux system administration**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKASL**
6.	Rodzaj przedmiotu/modułu **Kurs inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Brak
13.	Cele kształcenia dla przedmiotu Celem kursu jest rozwinięcie u uczestników praktycznych umiejętności zarządzania systemem Linux na komputerze osobistym lub serwerze. Zostaną przedstawione podstawowe informacje na temat budowy i sposobu działania systemów uniksopodobnych, a szczególnie różnych dystrybucji Linuksa. Zajęcia są adresowane do wszystkich studentów, którzy chcieliby sprawnie korzystać na co dzień z systemów linuksowych.

14. Treści programowe

I. WSTĘP: ŻYCIE Z LINUKSEM

1. Hardware i software. Budowa komputera i struktura oprogramowania podstawowego. Jądro i przestrzeń użytkownika. Proces rozruchu komputera. Instalacja systemu i rozpoczęcie pracy z Linuksem.
2. Terminal tekstowy. Powłoka systemowa. Podstawowe polecenia. Uruchamianie programów, standardowe strumienie wejściowe i wyjściowe, potoki. Uruchamianie i zatrzymywanie procesów. Zarządzanie plikami, prawa dostępu, kopiowanie, archiwizowanie. Edytory tekstu. Przetwarzanie plików tekstowych.
3. Użytkownik root. Hasła. Su i sudo.

II. ZAGADNIENIA SZCZEGÓŁOWE

1. Hardware i jego diagnostyka, lshw, lsusb, lspci, dmidecode, i2c-tools i in. Urządzenia. Sysfs i udev. Udevd i udevadm.
2. Dyski. Protokoły SCSI i ATA. USB Storage. Hdparm i smartmontools.
3. Partycje. MBR i GPT. Narzędzia do partycjonowania dysków, fdisk, parted.
4. Urządzenia blokowe, device mapper, losetup, dm-crypt, LVM.
5. Systemy plików. Budowa systemu plików, inode. Księgowanie. Systemy ext2, ext3 i ext4. Tworzenie i zarządzanie systemami plików. Diagnostyka systemów plików, fsck. Montowanie, mount, /etc/fstab itp. Dowiązania, pliki specjalne. FAT i NTFS. Filesystem in userspace. Systemy plików btrfs, ufs, zfs, ffs i in. Szczególne wymagania dysków SSD, discard i fstrim. Linux Directory Structure, katalogi /etc, /usr, /var, /tmp, /boot i ich przeznaczenie.
6. Przestrzeń wymiany w osobnej partycji i w pliku. Przestrzeń wymiany w pamięci RAM, zram, zswap.
7. Rozruch komputera. Firmware, Legacy BIOS i UEFI. Konfiguracja, efibootmgr. Inne (Coreboot, Libreboot, U-Boot). Bootloadery drugiego poziomu. GNU Grub 2 i jego konfiguracja. Informacja o Legacy GRUB. Inne popularne bootloadery, Syslinux, LILO, systemd-boot (Gummiboot). Konfiguracja wielosystemowa z BIOS i UEFI, współżycie z systemem MS Windows, rEFInd. PXE i rozruch poprzez sieć komputerową. BOOTP i TFTP. Plymuth i estetyka rozruchu.
8. Start jądra. Tworzenie i konfigurowanie initramfs. Uruchamianie awaryjne i tryb pojedynczego użytkownika. Moduły jądra, modprobe, lsmod, rmmod, /etc/modules.
9. Konfiguracja systemu. Katalog /etc i jego struktura.
10. Uruchamianie przestrzeni użytkownika. System V Init. Proces init, inittab, init.d i rc?.d, runlevels, telinit, update-rc.d, insserv, run-parts. Informacja o Upstart.
11. Systemd i jego konfiguracja. Filozofia działania systemd. Organizacja i składnia plików konfiguracyjnych. Demony systemd. Jednostki systemd. Cele i metody ich osiągania. Serwisy i inne jednostki.
12. Zarządzanie użytkownikami. Logowanie do systemu, getty i login. Konsola zdalna i minicom. Hasła, passwd i shadow. PAM. Własność procesów, suid.
13. Dziennik systemowy, rsyslog, logger, /var/log, journalctl i dmesg. Zdalny zapis dziennika. Protokół SMTP i narzędzia do korzystania z niego.
14. Synchronizacja czasu, NTP, ntpd, ntpdate, open-ntp, chrony, systemd-timesyncd.
15. Planowanie wykonania procesów, procesy wsadowe. Cron, crontab, anacron, systemd-timer. Polecenie at.
16. Zarządzanie procesami i ich diagnostyka, ps, lsof, strace, ltrace. Wysyłanie sygnałów do procesów, kill, killall, nohup. Wątki. Pomiar czasu procesora, obciążenia, zużycia pamięci i operacji wejścia/wyjścia (top, times, vmstat, iostat, iotop, pidstat itp). Priorytety procesów, nice. Komunikacja międzyprocesowa. Gniazda. Dbus. Komunikacja między użytkownikami, wall.
17. Biblioteki współdzielone, /lib, /usr/lib, ldd, ldconfig, /etc/ld.so.conf, LDLIBRARYPATH.
18. Repozytoria pakietów i dystrybucje Linuksa. Debian, Ubuntu i pakiety deb. Konfigurowanie apt i aptitude. RHEL, Fedora, Centos i rpm. Arch Linux i pacman. Dystrybucje specjalizowane, Kali Linux, Tails i in. Krzyżowanie dystrybucji. Linux from scratch.
19. Efektywna praca w trybie tekstowym. Powłoki bash (sh, dash, ash), c-shell (csh, tcsh), zsh. Biblioteka readline. Terminal, terminfo, termcap. Screen i tmux. Konfiguracja iterpretera powłoki. Tekst zachęty, Powershell. Lokalizacja i internacjonalizacja, locale, tzdata, kodowanie znaków.
20. Skrypty powłoki. Narzędzia sed, awk. Polecenia użytkowe wc, tr, sort, head, tail, tee, xargs, grep, find, locate i wiele innych.
21. Separacja procesów, namespaces, cgroups. Piaskownice, firejail i in. Apparmor. Selinux.
22. Wirtualizacja. LXC i docker. Qemu i KVM. Xen. Informacja o programie Virtualbox.
23. Sieci komputerowe i podstawowa konfiguracja sieci w Linuksie. Podstawowe polecenia konfiguracyjne. Konfiguracja statyczna i dynamiczna. DHCP. Gniazda. Bezpieczeństwo i iptables. Usługi sieciowe, sshd, inetd i xinetd. Gniazda systemd. RPC. Usługi nazw, bind9.
24. Usługi drukowania, lpr i Cups.
25. Informacja o trybie graficznym (gdyby kiedyś był potrzebny). Serwer wyświetlania X Windows, xinit, startx. Graficzny zarządca logowania (xdm, gdm, sddm, lightdm i in). Zarządca okien (kwin, Metacity, xfwm, Openbox i wiele in). Typy zarządców okien (tiling, stacking, compositing, dynamic). Środowisko graficzne (KDE, Gnome, XFCE i wiele innych). XTerminale (xterm, urxvt, konsole, terminal i in). Niedoskonałości serwera X Windows. Wayland. Zdalny terminal graficzny

15.	Zakładane efekty uczenia się

Wiedza

Zna podstawowe funkcje systemu operacyjnego i mechanizmy jego działania	K_W15
Zna podstawowe narzędzia administracyjne w Linuksie	K_W15

Umiejętności

Potrafi efektywnie wykonywać podstawowe czynności administracyjne w Linuksie	K_U19
Potrafi diagnozować problemy powstające podczas pracy systemu operacyjnego i sprawnie je rozwiązywać	K_U18

16. Literatura obowiązkowa i zalecana

1. Brian Ward, Jak działa Linux. Podręcznik administratora, wydanie II, Helion 2015.
2. Evi Nemeth, Garth Snyder, Trent R. Hein, Ben Whaley, Dan Mackin, Unix i Linux. Przewodnik administratora systemów, wydanie V, Helion 2018.

17. Metody weryfikacji zakładanych efektów uczenia się

- prezentacja rozwiązań zadań przy komputerze

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Ocena zajęć zależy od liczby punktów za prezentację rozwiązań zadań zdobytych podczas zajęć na pracowni.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do pracowni	30 godz.

Sumarycznie

Łączna liczba godzin	120 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs języka C++** **C++ course**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DK6CPP**
6.	Rodzaj przedmiotu/modułu **Kurs podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • kurs języka ANSI C z elementami C++ **Niezbędne kompetencje:** • umiejętność programowania strukturalnego i proceduralnego w języku ANSI C; • znajomość podstawowych struktur danych; • umiejętność czytania anglojęzycznej dokumentacji.
13.	Cele kształcenia dla przedmiotu Celem tego kursu jest nauczenie programowania obiektowego w bardzo popularnym współcześnie języku programowania C++. Na kursie jest prezentowany najnowszy standard tego języka oraz obszerne fragmenty biblioteki standardowej STL. Wykład jest ilustrowany programami napisanymi zgodnie z najlepszymi wzorcami, natomiast towarzyszące zajęcia laboratoryjne mają nauczyć dobrych praktyk w programowaniu i projektowaniu obiektowym.

14.	Treści programowe

1. Podstawowe konstrukcje językowe w C++.
2. Abstrakcja i hermetyzacja.
3. Inicjalizacja, kopiowanie, przenoszenie.
4. Składowe statyczne.
5. Przeciążanie operatorów.
6. Dziedziczenie, wielodziedziczenie.
7. Polimorfizm, klasy abstrakcyjne.
8. Przestrzenie nazw.
9. Wyjątki, asercje.
10. Szablony funkcji i szablony klas.
11. Konwersje.
12. Strumienie, operacje na plikach.
13. Podstawowe kolekcje standardowe.
14. Podstawowe algorytmy.
15. Operacje na łańcuchach znakowych.

15. Zakładane efekty uczenia się

Wiedza

Student zna standardowe typy danych w języku C++ oraz podstawowe konstrukcje programistyczne (instrukcje sterujące, funkcje) wykorzystywane w programowaniu imperatywnym	K_W04, K_W15
Student zna proste i złożone typy danych używane w języku C++	K_W15
Student zna zasady programowania obiektowego w C++	K_W04, K_W15
Student zna praktyczne aspekty złożoności pamięciowej i czasowej programów pisanych w języku wysokiego poziomu	K_W15

Umiejętności

Student potrafi pisać, uruchamiać i testować programy napisane w C++, używając wybranego środowiska programistycznego	K_U16
Student umie samodzielnie rozwiązywać różne zadania programistyczne korzystając z języka C++	K_U17
Student potrafi wybrać i użyć w swoich programach podstawowych i złożonych struktur danych	K_U19
Student umie podzielić program w C++ na moduły	K_U19

Kompetencje społeczne

Student rozumie znaczenie samodzielnej pracy przy zdobywaniu nowych umiejętności	K_K06
Student umie zadawać pytania dotyczące rozwiązywania jakiegoś zadania	K_K01
Student umie opowiedzieć o swoim rozwiązaniu koncentrując się na najbardziej istotnych zagadnieniach	K_K04

16. Literatura obowiązkowa i zalecana

Literatura podstawowa:

- B.Stroustrup: Język C++. Kompendium wiedzy. Wydanie 4. Wydawnictwo Helion, Gliwice 2014.
- J.Grębosz: Opus magnum C++11. Programowanie w języku C++. Tom 1, 2, 3. Wydawnictwo Helion, Gliwice 2018.

Literatura uzupełniająca:

- S.Rao: C++. Dla każdego. Wydanie 7. Wydawnictwo Helion, Gliwice 2014.
- S.Prata: Język C++. Szkoła programowania. Wydanie 6. Wydawnictwo Helion, Gliwice 2012.

17. Metody weryfikacji zakładanych efektów uczenia się

Pisanie i prezentacja krótkich programów komputerowych i małych aplikacji (miniprojekty).

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia przedmiotu należy zdobyć minimalną wymaganą (podaną w regulaminie przedmiotu) liczbę punktów, które uzyskuje się za wykonane w ramach pracowni programy.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

przygotowanie do pracowni	45 godz.
czytanie literatury i dokumentacji	15 godz.

Sumarycznie

Łączna liczba godzin	120 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs programowania gier w silniku Unity** **Course: Game Development in Unity3D**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKUnityEng**
6.	Rodzaj przedmiotu/modułu **Kurs zaawansowany**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu • podstawy programowania w C (zalecane) • podstawowa znajomość dowolnego programu do modelowania 3D (zalecane) • znajomość języka angielskiego w stopniu umożliwiającym zrozumienie materiałów i tutoriali
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest teoretyczne, a także praktyczne zaznajomienie studentów z procesem projektowania, programowania i testowania gier w silniku Unity3D.
14.	Treści programowe • Projektowanie gier • Podstawy języka C • Pętla gry • Rendering pipeline • Post processing • Podstawy grafiki komputerowej (ścieżki renderingu, rasteryzacja, cieniowanie, śledzenie promieni, macierze)

15.	Zakładane efekty uczenia się

Wiedza

rozumie proces powstawania i testowania gry	K_W15
zna różne metody monetyzacji gier	K_W16
zna podstawy programowania w języku C#	K_W15
umie dostatkować (juce) grę	K_W15
rozumie pojęcie gamifikacji	K_W16

Umiejętności

umie zaprojektować, stworzyć oraz przetestować grę używając silnika Unity3D	K_U16, K_U17
umie zbudować grę na wiele platform (minimalnie: Windows lub macOS i Android lub iOS)	K_U16, K_U17
umie obsługiwać silnik Unity3D	K_U19

16. Literatura obowiązkowa i zalecana

- https://unity.com/learn
- https://www.youtube.com/user/Brackeys

17. Metody weryfikacji zakładanych efektów uczenia się

napisanie i prezentacja programu komputerowego w postaci gry, prezentacja finalnego projektu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za zadania pracowniane (gry) oraz finalny projekt. Punkty za wszystkie wyżej wymienione aktywności liczą się łącznie.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

dodatkowa praca własna	10 godz.
przygotowanie raportu/prezentacji	5 godz.
praca nad finalnym projektem	30 godz.

Sumarycznie

Łączna liczba godzin	105 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs programowania gier w silniku Unreal 5** **Course: Game Development in Unreal Engine 5**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKUnrealEng**
6.	Rodzaj przedmiotu/modułu **Kurs zaawansowany**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Niezbędne kompetencje:** • Podstawy programowania w C++
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest teoretyczne, a także praktyczne zaznajomienie studentów z procesem projektowania, programowania i testowania gier w silniku Unreal 5.
14.	Treści programowe • Projektowanie gier • Podstawy języka C • Pętla gry • Rendering pipeline • Post processing • Podstawy grafiki komputerowej (ścieżki renderingu, rasteryzacja, cieniowanie, śledzenie promieni, macierze)

15.	Zakładane efekty uczenia się

Wiedza

rozumie proces powstawania i testowania gry	K_W15
zna różne metody monetyzacji gier	K_W16
zna podstawy programowania w języku C++	K_W15
umie dostatkować (juce) grę	K_W15
rozumie pojęcie gamifikacji	K_W16

Umiejętności

umie zaprojektować, stworzyć oraz przetestować grę używając silnika Unreal 5	K_U16, K_U17
umie zbudować grę na wiele platform (minimalnie: Windows lub macOS i Android lub iOS)	K_U16, K_U17
umie obsługiwać silnik Unreal 5	K_U19

16. Literatura obowiązkowa i zalecana

- https://www.unrealengine.com/en-US/learn

17. Metody weryfikacji zakładanych efektów uczenia się

- implementacja i prezentacja programu komputerowego w postaci gry
- projekt, realizacja oraz prezentacja finalnego projektu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za zadania pracowniane (gry) oraz finalny projekt. Punkty za wszystkie wyżej wymienione aktywności liczą się łącznie.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

dodatkowa praca własna	10 godz.
przygotowanie raportu/prezentacji	5 godz.
praca nad finalnym projektem	30 godz.

Sumarycznie

Łączna liczba godzin	105 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs rozszerzony języka Python** **Advanced Python Programming**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DK11RKJP**
6.	Rodzaj przedmiotu/modułu **K1 - kurs podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu • Znajomość podstawowych struktur danych: listy, kolejki, drzewa binarne. • Podstawowa umiejętność programowania w Pythonie.
13.	Cele kształcenia dla przedmiotu Celem zajęć jest • rozwijanie wiedzy i umiejętności korzystania z zaawansowanych cech języka (listy składane, generatory, metaprogramowanie) • nabycie umiejętności posługiwania się standardowymi bibliotekami: programowanie wielowątkowe, usługi sieciowe, interfejs użytkownika, dostęp i przechowywanie danych • poznanie narzędzi wspierających rozwój projektów (testowanie jednostkowe, profilowanie, metryki) • przegląd popularnych zastosowań Pythona wraz ze stosowanymi narzędziami: aplikacje webowe, przetwarzanie danych, uczenie maszynowe
14.	Treści programowe • listy składane, generatory • obiekty w nowym stylu, metaklasy • wątki, procesy • interfejsy użytkownika • testowanie programów, standard PEP 8 • programowanie sieciowe • trwałe słowniki, Python DB API • django, matplotlib, pandas, PyTorch/TensorFlow

15.	Zakładane efekty uczenia się

Wiedza

zna składnię i semantykę języka Python	K_W04, K_W15
zna najpopularniejsze moduły dostępne w Pythonie	K_W15

Umiejętności

potrafi implementować zadania wykorzystując zaawansowane właściwości Pythona, takie jak wbudowane struktury danych, listy składane czy obiekty w nowym stylu	K_U16, K_U17
potrafi implementować w Pythonie różne funkcjonalności aplikacji (trwałe przechowywanie danych, implementacja interfejsu użytkownika)	K_U16, K_U17
potrafi zadbać o jakość kodu i zna rekomendacje dot. nazewnictwa i formatowania kodu	K_U19
potrafi zaprogramować aplikację internetową z wykorzystaniem odpowiedniej platformy (np. django)	K_U19
umie opracować i zaprezentować architekturę rozwiązania informatycznego	K_U17, K_U19,K_U21

16. Literatura obowiązkowa i zalecana

- Tarek Ziade - Expert Python Programming
- Dusty Phillips - Python 3 Object Oriented Programming
- Rick Copeland- Essential SQLAlchemy
- A. Holovaty, J. Kaplan-Moss - The Django Book. Release 2.0
- An introduction to Python programming with NumPy, SciPy and Matplotlib/Pylab - Antoine Lefebvre

17. Metody weryfikacji zakładanych efektów uczenia się

- implementacja zadań
- realizacja bardziej złożonego projektu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia pracowni należy zdobyć wskazaną w regulaminie zajęć liczbę punktów za implementację programów oraz za zrealizowanie projektu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

Implementacja zadań: h	45 godz.
realizacja i prezentacja projektu: h	15 godz.

Sumarycznie

Łączna liczba godzin	120 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs: Podstawowy warsztat informatyka** **Computing for Computer Scientists**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKPWI**
6.	Rodzaj przedmiotu/modułu **Kurs podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 10 godzin, pracownia — 10 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Umiejętność obsługi komputera
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest przybliżenie studentom pierwszego roku podstawowych narzędzi do współtworzenia, kompilowania, uruchamiania, publikowania oraz dokumentowania swoich oraz cudzych programów.
14.	Treści programowe 1. Praca w systemie Linux: pliki, procesy, użytkownicy, potoki i strumienie, kompilowanie i uruchamianie programów, bash. 2. Polecenie ssh oraz podstawy kryptografii (klucze prywatne i publiczne) 3. Systemy kontroli wersji na przykładzie systemu git. 4. Podstawy pracy grupowej. 5. LaTeX (poprawne składanie tekstów takich jak rozwiązanie zadania, cv i list motywacyjny, wyrażenia matematyczne).

15. Zakładane efekty uczenia się

Wiedza

zna i rozumie sposoby wykorzystywania narzędzi informatycznych wymaganych w pracy informatyka	K_W15
posiada wiedzę o podstawowych narzędziach kryptograficznych, zna i rozumie podstawowe zagadnienia związane z bezpieczeństwem komputerowym z perspektywy użytkownika	K_W15
rozumie zalety korzystania z systemów kontroli wersji: zna i rozumie filozofię systemu git	K_W15

Umiejętności

potrafi używać systemu Linux	K_U19
jest w stanie samodzielnie nauczyć się wykorzystywania nieznanych mu narzędzi informatycznych na podstawie dokumentacji oraz samodzielnego wyszukiwania informacji w internecie	K_U22
posiada umiejętność krytycznego podejścia do informacji w wyszukanych przez siebie źródłach	K_U22
używa podstawowych narzędzi kryptograficznych	K_U19
potrafi używać narzędzi pozwalające na aktywny udział w projekcie programistycznych (git i serwisy typu github)	K_U19
potrafi składać tekst w wykorzystaniem systemu LaTeX	K_U16, K_U19

16. Literatura obowiązkowa i zalecana

W związku z deklarowanym celem przedmiotu - nauczenie studentów umiejętności samodzielnego znajdowania odpowiednich źródeł informacji, a także krytycznego podejścia do tych źródeł - nie jest podana literatura obowiązkowa.
Zalecane źródła: dokumentacja narzędzi informatycznych, podręczniki użytkownika, specjalistyczne fora dyskusyjne itp.

- Official Ubuntu Documentation https://help.ubuntu.com/
- GNU Bash Manual https://www.gnu.org/software/bash/manual/
- git User Manual https://git-scm.com/doc
- OpenSSH Manual https://www.openssh.com/manual.html
- LaTeX Documentation https://www.latex-project.org/help/documentation/
- The beamer class User Guide https://github.com/josephwright/beamer

17. Metody weryfikacji zakładanych efektów uczenia się

- prezentacja rozwiązania zadania
- test

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia pracowni należy zdobyć wymaganą, podaną w regulaminie przedmiotu liczbę punktów za rozwiązanie zadań na pracownię oraz za napisanie testu.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	10 godz.
pracownia	10 godz.

Praca własna studenta

przygotowywanie się do pracowni (rozwiązywania zadań, wyszukiwanie informacji w dokumentacji oraz w innych źródłach, czytanie materiałów uzupełniających)	30 godz.
przygotowanie projektu	25 godz.

Sumarycznie

Łączna liczba godzin	75 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs: Tworzenie aplikacji frontendowych** **Front-end apps development**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKFWebDev**
6.	Rodzaj przedmiotu/modułu **Kurs inżynierski**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Wstęp do programowania w języku C lub Python **Niezbędne kompetencje:** • podstawowa umiejętność programowania • podstawowa znajomość dowolnego systemu kontroli wersji
13.	Cele kształcenia dla przedmiotu Wykład ma na celu przybliżyć słuchaczom rolę frontend developera i związane z nią odpowiedzialności, a także przekazać wiedzę umożliwiającą wytworzenie pierwszych frontendowych aplikacji.

14. Treści programowe

1. Podstawy działania Internetu
2. HTML
   - Podstawy HTML
   - Dobre praktyki i konwencje
   - Formularze i walidacja
3. CSS
   - Podstawy CSS
   - Tworzenie Layoutu: Float, Grid, Flexbox, Positioning, Box Model
   - Responsywny design
   - Preprocesory CSS
   - BEM
4. JavaScript
   - Podstawy JavaScriptu
   - Manipulacja DOM
   - ES6+
   - Zaawansowane pojęcia: Hoisting, Event Bubbling, Scope, Prototype, Shadow DOM
   - Event Loop
   - TypeScript
5. Komunikacja z Backendem
   - REST, SOAP?, GraphQL?
6. Menadżer Pakietów
7. Narzędzia
   - Bundlery: Webpack, EsBuild
   - NPM
   - Lintery i formatery
8. Frameworki/Biblioteki
   - Przegląd dostępnych opcji
   - Podstawy React'a
9. Nowoczesny CSS
   - Styled Components
   - CSS Modules
   - Frameworki CSS: Material, Bootstrap, inne?
10. Testowanie Aplikacji
11. Optymalizacja

15. Zakładane efekty uczenia się

Wiedza

Posiada przeglądową wiedzę na temat wytwarzania aplikacji frontendowych	K_W15
Zna różne metody wytwarzania nowoczesnych aplikacji fontendowych	K_W15
Zna różne metody konfigurowania nowoczesnego środowiska developerskiego	K_W15
Zna podstawowe konstrukcje programistyczne oraz standardowe typy danych w języku JavaScript	K_W04, K_W15
Ma podstawową wiedzę na temat nowoczesnych Frameworków używanych w codziennej pracy Frontend Developera	K_W15
Zna podstawy tworzenia różnego rodzaju rozwiązań w Internecie	K_W15

Umiejętności

Potrafi podążać za zmianami technologicznymi	K_U22
Umie tworzyć rozwiązania frontendowe w oparciu o dostępne zasoby (narzędzia, biblioteki, frameworki itd.)	K_U16, K_U17
Umie wykorzystać istniejące zasoby frontendowe w budowie własnego rozwiązania	K_U19
Umie prezentować swoje idee w sposób dostosowany do wiedzy słuchaczy	K_U21

Kompetencje społeczne

Rozumie z czym wiąże się praca Frontend Developera	K_K04
Rozumie znaczenie roli Frontend Developera	K_K04

16. Literatura obowiązkowa i zalecana

- https://roadmap.sh/frontend

17.	Metody weryfikacji zakładanych efektów uczenia się Prezentacja rozwiązania zadań
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Zdobycie wymaganej liczby punktów z pracowni
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** wykład: 30 godz. pracownia: 30 godz. **Praca własna studenta** Rozwiązywanie zadań na pracownię: 40 godz. Studiowanie materiału z wykładu: 20 godz. **Sumarycznie** Łączna liczba godzin: 120 godz. Liczba punktów ECTS: 5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs: Wstęp do programowania w języku C** **Introduction to Programming in C**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DK13WDPC**
6.	Rodzaj przedmiotu/modułu **Kurs podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Przydatna będzie podstawowa umiejętność programowania.
13.	Cele kształcenia dla przedmiotu Celem tego kursu jest nauczenie i rozwinięcie podstawowych umiejętności programowania w jednym z najbardziej rozpowszechnionych języków imperatywnych, tzn. w języku C. Wykład jest ilustrowany wieloma przykładami, a towarzyszące mu zajęcia w pracowniach uczą (poprzez wiele drobnych zadań i końcowy projekt) praktyki dobrego programowania.
14.	Treści programowe • Podstawowe konstrukcje języka: instrukcje i deklaracje. • Standardowe typy danych, wyrażenia. • Standardowe wejście/wyjście, filtry. • Funkcje, moduły i struktura programu. • Metody agregacji danych: tablice, struktury i unie. • Wskaźniki. • Dynamiczne zarządzanie pamięcią, standardowe biblioteki C. • Sekwencyjne i swobodne przetwarzanie plików. • Programowanie okienkowego interfejsu w GTK+. • Klasy, obiekty i strumienie w C++.

15. Zakładane efekty uczenia się

Wiedza

Zna podstawowe konstrukcje programistyczne oraz standardowe typy danych w języku C (przypisanie, instrukcje sterujące, wywoływanie funkcji i przekazywanie parametrów, typy całkowite i zmiennopozycyjne). Wie jak napisać proste sparametryzowane funkcje w C.	K_W15
Rozumie pojęcie składni języka programowania.	K_W04
Zna podstawowe struktury danych i wykonywane na nich operacje (tablice, napisy, rekordy, pliki, wskaźniki, struktury wskaźnikowe, listy, stosy, kolejki i drzewa).	K_W08

Umiejętności

Potrafi pisać, uruchamiać i testować programy w C używając wybranego środowiska programistycznego.	K_U16
Potrafi wybrać i użyć w swoich programach podstawowe struktury danych.	K_U03
Umie podzielić program na moduły.	K_U17, K_U19
Potrafi pisać funkcje rekurencyjne.	K_U19
Umie czytać ze zrozumieniem programy zapisane w języku C.	K_U16

16. Literatura obowiązkowa i zalecana

- Kernighan B.W., Ritchie D.M.: Język ANSI C. Seria "Klasyka Informatyki". Warszawa, Wydawnictwa Naukowo-Techniczne 2003. Podręcznik podstawowy.
- King K.N.: Język C, Nowoczesne programowanie, Wydanie II, Helion 2011 (uwzględnia standard C99).
- Banahan M., Brady D., Doran M., The C Book, `http://publications.gbdirect.co.uk/c_book/` (dostępny za darmo jako pdf i na WWW).
- Schildt B., C: The Complete Reference: covers C++ and ANSI C, Osborne McGraw-Hill,
- ISO/IEC 2011 - Programming languages - C (aktualny standard ISO 2011), http://www.open-std.org/JTC1/SC22/wg14/www/docs/n1570.pdf (wstępna wersja opisu standardu).

17. Metody weryfikacji zakładanych efektów uczenia się

napisanie i prezentacja programu komputerowego; przygotowanie, zaprogramowanie i prezentacja projektu; kolokwium pisemne

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Zaliczenie wykładu:

- Kolokwium sprawdzające umiejętność programowania w C poprzez rozwiązanie 3 krótkich zadań programistycznych, w tym umiejętność napisania kompletnego programu i funkcji. Punkty uzyskane z kolokwium są doliczane do punktów z pracowni.

Zaliczenie pracowni:

- Listy zadań do samodzielnego programowania, wprowadzające kolejne elementy języka i oceniane przez prowadzących pracownię lub automatycznie przez system sprawdzający.
- Końcowy projekt sprawdzający umiejętność zaprojektowania i zaprogramowania większego programu. Uzyskane punkty są sumowane z punktami z kolokwium i wystawiana jest jedna ocena końcowa.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

udział w kolokwium	2 godz.
przygotowanie do kolokwium	3 godz.
przygotowanie do pracowni	30 godz.
praca nad projektem	20 godz.
studiowanie tematyki wykładów i literatury	10 godz.

Sumarycznie

Łączna liczba godzin	125 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs: Wstęp do programowania w języku Python** **Introduction to Programming in Python**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DK13WDPP**
6.	Rodzaj przedmiotu/modułu **Kurs podstawowy**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Przedmiot nie ma żadnych wymagań wstępnych (adresowany jest do studentów pierwszego semestru)
13.	Cele kształcenia dla przedmiotu Przedmiot ma dwa główne cele. Pierwszym jest nauczenie studentów programowania w języku imperatywnym. Drugim celem jest przedstawienei specyficznych cech Pythona. Zakładamy, że dla części uczestników będzie to pierwszy język programowania (aczkolwiek wykład może być pożyteczny również dla osób umiejących programować i nie znających Pythona)

14.	Treści programowe 1. Podstawowe składniki programów: stałe, operatory, wyrażenia, zmienne, instrukcja podstawienia, sekwencje instrukcji, instrukcja warunkowa, pętla while, pętla for i in range(N) 2. Typy proste: int, float, typ logiczny, napisy, podstawowe operacje na wartościach tych typów, leniwe wartościowanie wyrażeń logicznych 3. Definiowanie funkcji, przekazywanie parametrów, możliwe semantyki instrukcji przypisania (przepisywanie referencji i kopiowanie), zmienne lokalne i globalne 4. Listy w Pythonie: podstawowe operacje, wycinki, iteracja po listach, pisanie funkcji (niezmieniających wartości argumentów) oraz procedur (funkcji nie zwracających wartości i mających efekty uboczne), różnica w pracy z obiektami zmiennymi i niezmiennymi na przykładzie list i napisów, tablice dwuwymiarowe implementowane jako listy list, implementacja prostych algorytmów sortowania 5. Biblioteka turtle jako przykład prostej biblioteki umożliwiającej tworzenie rysunków, model RGB kolorów 6. Zbiory oraz operacje na zbiorach, wykorzystanie bibliotecznego sortowania, porównanie efektywności czasowej różnych rozwiązań, ze szczególnym naciskiem na kwadratową złożoność pojedynczej pętli i na sposoby unikania tego problemu 7. elementy programowania funkcyjnego: listy składane, map, filter, funkcje jako wartości pierwszego rzędu, wyrażenia funkcyjne (lambda), rekurencja w definiowaniu funkcji, funkcje all i any, funkcje z różną liczbą argumentów 8. Podstawowe operacje na słownikach, praktyczne programy korzystające z dużych słowników (prosty program tłumaczący z polskiego na angielski), tworzenie słowników z wartościami domyślnymi (defaultdict z modułu collections), słownik jako rzadka reprezentacja macierzy (krotki jako klucze) 9. Czytanie i pisanie danych do plików, iteracja po pliku 10. Programy jako dane: wykorzystywanie funkcji eval i exec, wyjątki i ich przechwytywanie, tworzenie prostych interpreterów różnych języków bazujących na ww. funkcjach i przekształcaniu napisów 11. Drzewa i rekurencyjne struktury danych, grafy jako słowniki, proste algorytmy grafowe (przeszukiwanie wgłąb, wszerz, wyszukiwanie składowych) 12. yield i tworzenie funkcji iteratorów, elementy programowania obiektowego (obiekty, konstruktory, klasy, przeciążanie operatorów w celu tworzenia wariantów typów standardowych (liczby wymierne, listy z dowiązaniami o interfejsie analogicznym do standardowych list)

15. Zakładane efekty uczenia się

Wiedza

student zna podstawowe proste i złożone typy danych używane w językach programowania	K_W04, K_W07
student zna podstawowe instrukcje tworzące programy imperatywne	K_W04, K_W07, K_W15
student zna podstawy języka Python, podstawy programowania obiektywnego i funkcyjnego	K_W04, K_W15
student zna praktyczne aspekty złożoności pamięciowej i czasowej programów pisanych w języku wysokiego poziomu	K_W15

Umiejętności

student umie samodzielnie rozwiązywać z języku Python różne zadania programistyczne	K_U16
student umie analizować i modyfikować kod w języku Python napisany przez inną osobę	K_U16
student potrafi łączyć imperatywny styl programowania z elementami programowania funkcyjnego	K_U03
student umie pisać programy tworzące obrazy, przetwarzające tekst, dane liczbowe, operujące na plikach	K_U19
student umie pisać programy wykorzystujące zaawansowane struktury danych	K_U16, K_U17

Kompetencje społeczne

student rozumie znaczenie samodzielnej pracy przy zdobywaniu nowych umiejętności	K_K08
student umie zadawać pytania przy napotkaniu trudności przy rozwiązywaniu jakiegoś zadania	K_K01
student umie opowiedzieć o swoim rozwiązaniu jakiegoś problemu skupiając się na najbardziej istotnych dla słuchacza elementach	K_K04

16. Literatura obowiązkowa i zalecana

- Materiały na stronie wykładu (wszystkie zadanie da się rozwiązać, korzystając z tego źródła wiedzy)

Literatura uzupełniająca

- Materiały ze strony python.org
- Dowolny podręcznik Pythona dla początkujących, na przykład: Michael Dawson, "Python dla każdego. Podstawy programowania."

17. Metody weryfikacji zakładanych efektów uczenia się

Prezentacja programu napisanego w domu, samodzielne pisanie programów na zajęciach (twz. wprawki programistyczne), kolokwium

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- należy zdobyć wymaganą, określoną w regulaminie przedmiotu łączną liczbę punktów za zadania domowe, wprawki programistyczne (czyli programy pisane na zajęciach) oraz kolokwium
- dodatkowo konieczne jest zdobycie określonej w regulaminie minimalnej liczby punktów za kolokwium

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

Przygotowanie do kolokwium	15 godz.
Studiowanie materiałów dodatkowych	15 godz.
Samodzielna analiza programów z wykłądu	10 godz.
Dodatkowa praca nad zadaniami w domu	50 godz.

Sumarycznie

Łączna liczba godzin	150 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Kurs: zaawansowane techniki w C++ i STL** **Advanced techniques in C++ and STL**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DKZCPP**
6.	Rodzaj przedmiotu/modułu **Kurs zaawansowany**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • kurs języka C++ • wstęp do informatyki / algorytmy i struktury danych **Niezbędne kompetencje:** • umiejętność programowania obiektowego w języku C++; • znajomość klasycznych struktur danych wykorzystywanych w algorytmice; • umiejętność czytania anglojęzycznej dokumentacji.
13.	Cele kształcenia dla przedmiotu Celem kursu jest opanowanie przez studentów zaawansowanych technik programowania w nowoczesnej wersji języka C++ oraz maksymalnego wykorzystania możliwości zmodernizowanej biblioteki standardowej STL. Wykład jest ilustrowany programami napisanymi zgodnie z najlepszymi wzorcami, natomiast towarzyszące zajęcia laboratoryjne mają nauczyć dobrych praktyk w programowaniu i projektowaniu obiektowym z wykorzystaniem szablonów.

14.	Treści programowe

1. obsługa błędów i wyjątków
2. łańcuchy i konwersje liczbowe
3. wyrażenia regularne
4. inteligentne wskaźniki
5. zegary i czasomierze
6. obiekty funkcyjne i lambdy
7. kontenery i adaptatory kontenerów
8. iteratory i adaptatory iteratorów
9. iteratory strumieniowe i plikowe
10. algorytmy niemodyfikujące i modyfikujące
11. liczby zespolone i algorytmy numeryczne
12. liczby i rozkłady pseudolosowe
13. współbieżność i synchronizacja wątków
14. metaprogramowanie
15. lokalizacja

15. Zakładane efekty uczenia się

Wiedza

Student zna zasady programowania obiektowego w C++	K_W04, K_W15
Student zna typy szablonowe w języku C++	K_W07
Student zna praktyczne aspekty złożoności pamięciowej i czasowej programów pisanych w języku wysokiego poziomu	K_W15

Umiejętności

Student potrafi pisać, uruchamiać i testować programy napisane w C++, używając wybranego środowiska programistycznego	K_U16
Student umie samodzielnie rozwiązywać trudne zadania programistyczne korzystając z języka C++ i biblioteki STL	K_U17, K_U19
Student potrafi wybrać i użyć w swoich programach struktury danych i algorytmy zdefiniowane w bibliotece standardowej	K_U19
Student umie podzielić program w C++ na moduły	K_U17

Kompetencje społeczne

Student rozumie znaczenie samodzielnej pracy przy zdobywaniu nowych umiejętności	K_K08
Student umie zadawać pytania dotyczące rozwiązywania jakiegoś zadania	K_K01
Student umie opowiedzieć o swoim rozwiązaniu koncentrując się na najbardziej istotnych zagadnieniach	K_K04

16. Literatura obowiązkowa i zalecana

Literatura podstawowa:

- B.Stroustrup: Język C++. Kompendium wiedzy. Wydanie 4. Wydawnictwo Helion, Gliwice 2014.
- N.M.Josuttis: C++. Biblioteka standardowa. Podręcznik programisty. Wydanie 2. Wydawnictwo Helion, Gliwice 2014.

Literatura uzupełniająca:

- J.Grębosz: Opus magnum C++11. Programowanie w języku C++. Tom 1, 2, 3. Wydawnictwo Helion, Gliwice 2018.
- J.Galowicz: C++17 STL. Receptury. Wydawnictwo Helion, Gliwice 2018.

17. Metody weryfikacji zakładanych efektów uczenia się

Pisanie i prezentacja krótkich programów komputerowych, powiązanych tematycznie z materiałem prezentowanym podczas wykładu.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia przedmiotu należy zdobyć minimalną wymaganą (podaną w regulaminie przedmiotu) liczbę punktów, które uzyskuje się za wykonane w ramach pracowni programy.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
pracownia	30 godz.

Praca własna studenta

przygotowanie do pracowni	45 godz.
czytanie literatury i dokumentacji	30 godz.

Sumarycznie

Łączna liczba godzin	135 godz.
Liczba punktów ECTS	5

5 Projekty programistyczne

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Projekt: Kompetytywna Sztuczna Inteligencja** **Project: Competitive Artificial Intelligence**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DPZCompaAI**
6.	Rodzaj przedmiotu/modułu **Projekt**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Znajomość tematyki omawianej na przedmiocie Sztuczna Inteligencja (wymagane) oraz Seminarium: Kompetytywna Sztuczna Inteligencja (zalecane), Artificial Intelligence for Games (zalecane).
13.	Cele kształcenia dla przedmiotu Podstawowe cele przedmiotu tu: • teoretyczne i praktyczne zaznajomienie studentów z procesem projektowania, programowania i testowania botów • zaznajomienie ze specyfiką gier programistycznych / zawodów AI • rozwinięcie umiejętności pisania algorytmów AI A także nauka współpracy w grupie, wyciągania wniosków z nawzajemnych dyskusji, krytycznego myślenia.
14.	Treści programowe 1. Zapoznanie się ze specyfiką wybranych zawodów programistycznych AI 2. Organizacja w grupie, wybór tematyki projektu 3. Klasyfikacja gry (pełna informacja, losowość, analiza wielkości drzewa gry) i wiążących się z tym następstw - w szczególności określenie adekwatnych podejść 4. Implementacja wstępnej wersji projektu, testy porównawcze różnych algorytmów AI, dyskusje międzygrupowe 5. Implementacja pełnej wersji bota, optymalizacja wybranych algorytmów 6. Udział w zawodach

15. Zakładane efekty uczenia się

Wiedza

zna działanie co najmniej jednego z frameworków wykorzystywanego w zawodach sztucznej inteligencji	K_W15
rozumie proces powstawania i testowania bota	K_W15
rozumie specyfikę gier programistycznych, zna klasyfikację gier ze względu na cechy charakterystyczne	K_W15
zna algorytmy AI i podejścia specyficzne dla rozwiązywania gier	K_W14

Umiejętności

umie zaprojektować, stworzyć oraz przetestować bota grającego w zadaną zasadami konkursu grę	K_U16, K_U17, K_U18, K_U19
umie ocenić treść problemu pod kątem potencjalnie najbardziej skutecznych rozwiązań	K_U19
umie zaprogramować algorytmy AI przystosowane do różnorodnych zagadnień	K_U16, K_U17, K_U18, K_U19
umie zaprezentować swój projekt, przedstawić uzasadnienia dla podjętych w czasie jego realizacji decyzji oraz w sposób krytyczny analizować pomysły swoje innych	K_U21
potrafi współpracować w grupie i modyfikować wstępne założenia na podstawie otrzymanych informacji zwrotnych	K_U23
potrafi analizować podejścia innych zespołów oraz przekazywać wartościową informację zwrotną	K_U23

16. Literatura obowiązkowa i zalecana

Wybrane prace naukowe z dziedziny algorytmów AI, blogi uczestników zawodów, opisy zwycięskich programów, dokumentacja techniczna konkursów.

17. Metody weryfikacji zakładanych efektów uczenia się

Napisanie i prezentacja programu bota skutecznie grającego w zadaną grę, regularny udział w dyskusjach międzygrupowych, prezentacja finalnego projektu, udział w zawodach jeśli to możliwe (nie wymagane ale zalecane).

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- Uczestnictwo w spotkaniach międzygrupowych, regularne prezentacje postępów prac
- Aktywne branie udziału w prezentacjach programów innych grup, konsultowanie pomysłów
- Stworzenie i prezentacja gotowego projektu
- Napisanie raportu z rozwoju projektu

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.

Praca własna studenta

Wybór tematu i zapoznanie się z zasadami wybranych zawodów	10 godz.
Przygotowanie raportu końcowego	5 godz.
Prezentacja prac i konsultacje z innymi grupami	20 godz.
Przygotowanie, implementacja i testy	65 godz.

Sumarycznie

Łączna liczba godzin	130 godz.
Liczba punktów ECTS	4

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Projekt: Machine Learning for Temporal Data Mining** **Project: Machine Learning for Temporal Data Mining**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DPZMLforTDM**
6.	Rodzaj przedmiotu/modułu **Projekt**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu wiedza z eksploracji danych i uczenia maszynowego (w zakresie potrzebnym do realizacji projektu) oraz umiejętności techniczne w tym zakresie
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest nabycie/doskonalenie umiejętności pracy zespołowej oraz rozszerzenie wiedzy z algorytmów uczenia maszynowego do eksploracji i modelowania danych temporalnych.
14.	Treści programowe Wybrane algorytmy uczenia maszynowego danych do eksploracji i modelowania danych temporalnych, m.in.: • sposoby reprezentacji danych temporalnych i ekstrakcji cech, • algorytmy grupowania i klasyfikacji danych temporalnych, • modelowanie danych temporalnych ze stanami ukrytymi, • zastosowania w systemach wspomagania decyzji.

15.	Zakładane efekty uczenia się

Wiedza

rozumie wybrane zaawansowane podejścia uczenia maszynowego dla danych temporalnych	K_W14
rozumie problemy związane z przetwarzaniem danych temporalnych	K_W15

Umiejętności

umie formułować zagadnienia praktyczne jako zadania uczenia maszynowego dla danych temporalnych	K_U19
umie stosować i modyfikować algorytmy uczenia maszynowego dla danych temporalnych	K_U17, K_U19
umie stosować metody uczenia maszynowego i głębokiego do danych temporalnych	K_U17, K_U19
umie prezentować swoje idee w sposób dostosowany do wiedzy słuchaczy	K_U21

16.	Literatura obowiązkowa i zalecana

wybrane artykuły naukowe z czasopism i konferencji z dziedziny eksploracji danych, inteligencji obliczeniowej i uczenia maszynowego

17.	Metody weryfikacji zakładanych efektów uczenia się

prezentacja, dyskusja

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Zaliczenie wymaga wykonywania przydzielanych indywidualnie zadań projektowych, rozliczania rezultatów i współpracy z grupą.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.

Praca własna studenta

przygotowywanie się do zajęć (w tym czytanie materiałów dodatkowych)	10 godz.
przygotowanie prezentacji rezultatów swoich prac	10 godz.
realizacja zadań projektowych	50 godz.

Sumarycznie

Łączna liczba godzin	100 godz.
Liczba punktów ECTS	4

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Projekt: Rozwój Systemu Zapisów** **Project: Development of the Enrollment System**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DP16PrZap**
6.	Rodzaj przedmiotu/modułu **Projekt**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **nieokreślony**
11.	Forma zajęć i liczba godzin pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Podstawowy warsztat informatyka **Niezbędne kompetencje** • Podstawowa znajomość relacyjnych baz danych (SQL) • Świadomość technologii webowych (HTML, CSS, Javascript/TypeScript) • Podstawy Pythona
13.	Cele kształcenia dla przedmiotu Rozróżniamy dwa podstawowe cele przedmiotu. Z perspektywy prowadzącego celem jest rozwijanie Systemu Zapisów w naszym instytucie i wykorzystanie do tego celu zdolności oraz zapału studentów. Studenci jednocześnie (drugi cel) uczą się uczestnictwa w dużym projekcie programistycznym, do którego przychodzą, gdy projekt jest już dość rozwinięty. Muszą wykonać w nim pewne zadania, zmagając się jednocześnie z niedoskonałościami tego projektu. Jest to scenariusz bardziej odpowiadający realiom pracy zawodowej, niż na większości kursów, gdzie studenci tworzą projekt „od zera”.
14.	Treści programowe • Korzystanie ze standardowych narzędzi programistycznych: np. systemu kontroli wersji, CI (testów automatycznych). • Typowy proces tworzenia oprogramowania — zadania, pomysły, projektowanie, kodowanie, weryfikowanie (code review, testy), dołączanie. • Standardowe problemy i rozwiązania w oprogramowaniu webowym — bezpieczeństwo, radzenie sobie z równoległością, wątki, wydajność. • Modne technologie webowe: Django, Vue.

15.	Zakładane efekty uczenia się

Wiedza

Zna wybrane technologie używane w aplikacjach webowych (serwerowych).	K_W15
Rozumie zasady działania relacyjnych baz danych i umie efektywnie z nich korzystać.	K_W15

Umiejętności

Potrafi rozwijać, uruchamiać i testować oprogramowanie w zadanej technologii.	K_U16
Projektuje rozwiązanie dla postawionego zadania i prezentuje to rozwiązanie.	K_U17
Potrafi ocenić oraz uargumentować zasadność stosowania różnych narzędzi i technik.	
Potrafi zidentyfikować istotny dla siebie fragment kodu w dużym systemie informatycznym i przeanalizować jego działanie.	K_U18
Umie używać popularnych narzędzi do pracy zespołu programistycznego (takich jak Slack, Github).	K_U19
Umie zrobić rzeczowe „code-review” i nie przyjmuje postawy defensywnej, gdy ktoś zrobi „code-review” jemu.	K_U23
Umie korzystać z dokumentacji oraz znajdować w Internecie rozwiązania przy napotkanych przeszkodach technicznych.	K_U22

16. Literatura obowiązkowa i zalecana

- Dokumentacja Django (https://docs.djangoproject.com/en/).
- Dokumentacja Vue (https://vuejs.org/v2/guide/).
  Oczywiście dokumentacji nie trzeba czytać w całości.

17. Metody weryfikacji zakładanych efektów uczenia się

Realizacja zadań w Systemie Zapisów.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Projekt programistyczny, cotygodniowa ewaluacja pracy na spotkaniach.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.

Praca własna studenta

Praca nad projektem	70 godz.

Sumarycznie

Łączna liczba godzin	100 godz.
Liczba punktów ECTS	4

6 Seminaria

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Proseminarium: Bezpieczeństwo i ochrona informacji** **Seminar: Information Safety and Security**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-BiOI**
6.	Rodzaj przedmiotu/modułu **Proseminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium — 28 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu zainteresowanie tematyką bezpieczeństwa i ochrony informacji
13.	Cele kształcenia dla przedmiotu Celem seminarium jest przedstawienie wiedzy z zakresu bezpieczeństwa i ochrony informacji oraz przygotowanie słuchaczy do zgodnego z RODO wytwarzania, testowania i eksploatacji oprogramowania.
14.	Treści programowe • aspekty prawne bezpieczeństwa i ochrony informacji • bezpieczeństwo aplikacji webowych i mobilnych • certyfikaty, podpisy elektroniczne • bezpieczeństwo sieci i infrastruktury • bezpieczeństwo rozwiązań chmurowych i IoT • analiza i testowanie bezpieczeństwa • zapewnienie bezpieczeństwa i ochrony informacji w procesie wytwarzania oprogramowania • zapewnienie ciągłości działania • znaczenie wprowadzenia i utrzymywania polityki bezpieczeństwa • przegląd najnowszych ataków

15. Zakładane efekty uczenia się

Wiedza

Zna najważniejsze zagadnienia dotyczące zagrożenia bezpieczeństwa działania oprogramowania: ataków i awarii	K_W15, K_W16
Ma podstawową wiedzę dotyczącą prawnych aspektów pracy informatyka, ze szczególnym uwzględnieniem obowiązującego prawa autorskiego i zagadnień ochrony danych osobowych	K_W15, K_W16

Umiejętności

Potrafi wykonać prostą analizę podatności systemu systemu informatycznego na ataki i awarie	K_U18
Potrafi prezentować opracowane zagadnienia, formułować opinie, a także podejmować dyskusję i analizować problemy dotyczące zagrożeń systemów informatycznych związanych z atakami i awaryjnością	K_U15, K_U21
Potrafi dokonać krytycznej analizy zgodności wytwarzania, testowania i eksploatacji oprogramowania z RODO	K_U18

Kompetencje społeczne

Jest świadom ograniczeń własnej wiedzy i umiejętności w zakresie zagrożeń (ataków i awaryjności), zabezpieczania oraz przepisów prawa dotyczących systemów informatycznych. Rozumie konieczność doskonalenia swoich zawodowych kompetencji w tym zakresie	K_K01
Wykazuje gotowość do wypełniania społecznych zobowiązań wynikających z charakteru pracy typowej dla absolwentów kierunku informatyka, w szczególności w zakresie ciągłej analizy zagrożeń i zapewnienia bezpiecznego działania systemów informatycznych	K_K03

16. Literatura obowiązkowa i zalecana

1. Stallings W., Brown. : Bezpieczeństwo systemów informatycznych. Zasady i praktyka
2. Liderman K.: Bezpieczeństwo informacyjne
3. Depo J., Piwowarski J.: Bezpieczeństwo informacyjne
4. Stevens W.R.: Biblia TCP/IP
5. Weidman G. : Bezpieczny system w praktyce. Wyższa szkoła hackingu i testy penetracyjne
6. Wilhelm T.: Profesjonalne testy penetracyjne. Zbuduj własne środowisko do testów

17. Metody weryfikacji zakładanych efektów uczenia się

Opracowanie i przedstawienie prezentacji na zadany temat.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Ocena wygłoszonych referatów (wykładów audiowizualnych) pod kątem poprawności merytorycznej i komunikatywności przekazu. Konieczne jest również aktywne uczestnictwo w dyskusji na zajęciach.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	28 godz.

Praca własna studenta

studiowanie materiałów i opracowanie koncepcji prezentacji	20 godz.
przygotowanie prezentacji	10 godz.

Sumarycznie

Łączna liczba godzin	58 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Proseminarium: Linux kawałek po kawałku** **Linux piece by piece**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DSLinuxKPK**
6.	Rodzaj przedmiotu/modułu **Proseminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Kurs administrowania systemem Linux (nie wymagany, ale zalecany). **Niezbędne kompetencje:** • Znajomość podstaw administrowania systemem Linux. • Umiejętność korzystania z repozytoriów oprogramowania.
13.	Cele kształcenia dla przedmiotu Dogłębne poznanie architektury systemu linuksowego, jego komponentów i zależności pomiędzy nimi. Rozwinięcie umiejętności skompilowania ze źródeł, skompletowania i skonfigurowania działającego systemu linuksowego.

14.	Treści programowe Aby stworzyć kompletny system operacyjny wykorzystujący jądro Linuksa potrzeba skompletować od autorów (upstream) kody źródłowe wielu programów, dostosować je (patch) oraz skompilować i skonfigurować kompletną przestrzeń użytkownika. W ten sposób powstają tzw. dystrybucje Linuksa. Aby korzystać z Linuksa, zwykle instalujemy jedną z takich dystrybucji, dzięki temu jesteśmy zwolnieni z konieczności kompilowania i konfigurowania systemu, a nawet nie musimy sobie zdawać sprawy z jego zawartości. Świetnym sposobem na poznanie architektury systemu linuksowego jest własnoręczne zbudowanie go krok po kroku od zera. Pomaga w tym projekt Gerarda Beekmansa „Linux from scratch" rozpoczęty w 1999 i nadal aktywnie rozwijany (obecna wersja: 11.1). Jest to podręcznik wzbogacony sporą liczbą pomocniczych materiałów (skrypty, patche itp.). Wykonanie kompletnego projektu LFS, poza głębokim poznaniem wewnętrznej struktury systemu linuksowego, pozawala na zdobycie umiejętności niezbędnych do samodzielnego budowania specjalnych wersji takich systemów przeznaczonych do specjalnych zastosowań. Każdy z uczestników zajęć otrzyma do opracowania fragment systemu LFS, który powinien dobrze zrozumieć i przygotować w domu, a następnie przedstawić wszystkim uczestnikom zajęć podczas dwugodzinnej prezentacji tak, by mogli oni odtworzyć omówiony fragment we własnych instalacjach. Na koniec semestru wszyscy uczestnicy zajęć powinni mieć gotowe, działające instalacje.
15.	Zakładane efekty uczenia się **Wiedza** Wie, z jakich komponentów składa się system linuksowy (bootloader, jądro, init system, demony itd.), jakie są ich zadania i jak je można dostosowywać do własnych potrzeb — K_W15 Zna alternatywne rozwiązania różnych komponentów systemu linuksowego i potrafi wybrać najlepiej dostosowane do swoich potrzeb (np. OpenRC zamiast SystemD itp.) — K_W15 **Umiejętności** Potrafi skonfigurować każdy z komponentów systemu linuksowego — K_U19 Potrafi pobrać z repozytoriów źródłowych, skompilować i skonfigurować duże pakiety oprogramowania. W szczególności potrafi skompilować własną wersję jądra systemu Linux — K_U19 Potrafi współpracować z innymi nad wspólnym projektem — K_U23 **Kompetencje społeczne** Umie przedstawić pozostałym członkom grupy efekty własnej pracy, w szczególności zebrane informacje i własne doświadczenia — K_K05
16.	Literatura obowiązkowa i zalecana 1. Linux From Scratch, created by Gerard Beekmans, managing editor: Bruce Dubbs, editors: Douglas R. Reno, DJ Lucas.
17.	Metody weryfikacji zakładanych efektów uczenia się Opracowanie i przedstawienie prezentacji na zadany temat.
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Aby zaliczyć przedmiot należy: • w trakcie całego semestru kompilować i konfigurować krok po kroku swój własny system; • uczestniczyć w zajęciach omawiających kolejne komponenty systemu i wykorzystywać zdobytą na nich wiedzę do tworzenia własnego systemu; • przygotować i przeprowadzić jedne z takich zajęć (prezentacja, pokaz na żywo); • na ostatnich zajęciach przedstawić swój gotowy, działający system.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

przygotowanie prezentacji	35 godz.

Sumarycznie

Łączna liczba godzin	65 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Proseminarium: Wizualizacja danych** **Proseminar: Data Visualisation**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-PsWizDan**
6.	Rodzaj przedmiotu/modułu **Proseminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Brak
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie studentów z ponadczasowymi zasadami dotyczącymi prezentowania danych, a także z aktualnymi sposobami tworzenia wizualizacji.
14.	Treści programowe Przedmiot omawia podstawowe uniwersalne prawa dotyczące tworzenia wizualizacji, w tym dobór kolorów i formatu, a następnie przechodzi do wizualizowania różnego rodzaju danych: • wizualizacja słów (wordle), tekstów (w tym redukcja przestrzeni wielowymiarowych), • wizualizacja rzeczy i zdarzeń na mapach, w tym tworzenie map ciepła (heatmap), • wizualizacja danych społecznościowych i innych rodzajów grafów, • grafy przepływu, • i inne, w zależności od aktualnej mody. Oprócz tego na przedmiocie będziemy omawiać współczesne narzędzia do wizualizacji i ich możliwości.

15. Zakładane efekty uczenia się

Wiedza

Zna zasady tworzenia czytelnych wizualizacji danych	K_W15
Zna podstawowe narzędzia do tworzenia wizualizacji	K_W15

Umiejętności

Potrafi samodzielnie zgłębić zadany temat i przygotować na ten temat prezentacje	K_U15, K_U20
Potrafi wygłosić prezentację multimedialną oraz odpowiadać na zadawane w niej pytania, w tym potrafi zadbać o techniczną stronę prezentacji (stworzenie slajdów, odpowiednie ich odtworzenie, zadbanie o odpowiednie warunki do prezentowania narzędzi informatycznych w czasie swojej prezentacji)	K_U22
Potrafi odróżnić dobrą wizualizację od złej	K_U21
Potrafi odpowiednio dobrać typ wizualizacji do danych	K_U19
Potrafi zaprogramować kilka rodzajów wizualizacji danych	K_U16

Kompetencje społeczne

Ma umiejętność prezentowania w przystępny sposób swojej wiedzy innym	K_K05

16. Literatura obowiązkowa i zalecana

Steele, J., Iliinsky, N. (2010). Beautiful visualization: Looking at data through the eyes of experts. " O'Reilly Media, Inc.".

17. Metody weryfikacji zakładanych efektów uczenia się

prezentacja materiału przez studenta, stworzenie wizualizacji przez studenta

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Student powinien przeprowadzić prezentację i stworzyć trzy wizualizacje używając technik omówionych na przedmiocie. Ocena takiej osoby będzie średnią ważoną tych dwóch ocen, z wagami:

- prezentacja 75
- wizualizacje 25

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

dodatkowe konsultacje w ramach potrzeb	4 godz.
projekt końcowy	10 godz.
przygotowywanie się do prezentacji (w tym czytanie materiałów dodatkowych)	35 godz.

Sumarycznie

Łączna liczba godzin	79 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Rozstrzygalność i złożoność logik motywowanych informatyką** **Decidability and Complexity of Logics Motivated by Computer Science**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DSDecLogic**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium —- 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** • Logika w informatyce **Niezbędne kompetencje:** • mile widziana wiedza z przedmiotu Języki Formalne i Złożoność Obliczeniowa dotycząca rozstrzygalności/nierozstrzygalności oraz klas złożoności
13.	Cele kształcenia dla przedmiotu Seminarium ma na celu zapoznanie studentów z klasycznymi i nowymi wynikami dotyczącymi rozstrzygalności teorii logicznych motywowanych informatyką oraz rozwijanie umiejętności samodzielnego czytania prac naukowych dotyczących logiki w informatyce, ich analizowania i prezentowania.
14.	Treści programowe Techniki służące do badania rozstrzygalności i złożoności obliczeniowej problemów spełnialności i odpowiadania na zapytania do baz danych/baz wiedzy (także w klasie modeli skończonych).

15. Zakładane efekty uczenia się

Wiedza

Zna definicje fragmentów logiki pierwszego rzędu (logik) motywowanych informatyką i logik deskrypcyjnych. Wie o możliwych zastosowaniach takich logik w różnych obszarach informatyki (bazy danych, reprezentacja wiedzy, automatyczna weryfikacja)	K_W01, K_W14
zna klasyczne i najnowsze techniki używane do badania rozstrzygalności/złożoności problemów decyzyjnych w logice, rozumie zależność pomiędzy siłą wyrazu formalizmu a jego złożonością	K_W09, K_W14
Rozumie znaczenie rozstrzygalnych fragmentów logiki pierwszego rzędu/logik deskrypcyjnych dla różnych obszarów informatyki.	K_W13

Umiejętności

potrafi czytać i analizować artykuły naukowe dotyczące logiki w informatyce	K_U15, K_U20, K_U22
Potrafi przygotować i wygłosić referat	K_U15, K_U20, K_U22

Kompetencje społeczne

Umie dostosować formę i treść prezentacji do wiedzy i umiejętności słuchaczy	K_K05
Aktywnie prezentuje krytyczną postawę wobec stwierdzeń, uwag i wniosków.	K_K06

16. Literatura obowiązkowa i zalecana

- Szereg prac naukowych (zarówno nowych i jak i klasycznych) dotyczących tematyki przedmiotu.

17. Metody weryfikacji zakładanych efektów uczenia się

Opracowanie i wygłoszenie referatu na zadany temat.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Student będzie miał za zadanie przedstawić jedną lub dwie prezentacje na zadany temat. Oceniana będzie poprawność merytoryczna i jasność przekazu. Oczekuję również, że student będzie słuchał prezentacji innych uczestników seminarium oraz uczestniczył w dyskusji na ich temat.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

Przygotowanie prezentacji	20 godz.
zapoznawanie się z zadanym tematem	30 godz.

Sumarycznie

Łączna liczba godzin	80 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium aproksymacyjno-optymalizacyjno-kombinatoryczne** **Seminar on approximation algorithms and combinatorial optimization**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DS9AOK**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty:** * Algorytmy i struktury danych * Matematyka dyskretna **Niezbędne kompetencje:** * Umiejętność konstrukcji i analizy algorytmów. * Mile widziana znajomość jednego z działów zaawansowanej algorytmiki lub optymalizacji kombinatorycznej, np. algorytmów probabilistycznych, aproksymacyjnych, grafowych.
13.	Cele kształcenia dla przedmiotu Seminarium ma na celu zapoznanie studentów z bardziej złożonymi lub najnowszymi wynikami dotyczącymi algorytmiki i optymalizacji kombinatorycznej oraz rozwijanie umiejętności samodzielnego opracowywania, analizowania i prezentowania zaawansowanych zagadnień algorytmicznych i optymalizacyjnych.
14.	Treści programowe Analiza istotnych problemów optymalizacji kombinatorycznej.

15. Zakładane efekty uczenia się

Wiedza

na wybrane wyniki dotyczące badań algorytmicznych i problemów optymalizacji kombinatorycznej.	K_W07, K_W14

Umiejętności

Potrafi wyszukiwać, zgłębiać i analizować aktualne artykuły naukowe dotyczące algorytmiki i optymalizacji kombinatorycznej.	K_U15, K_U20, K_U22
Potrafi samodzielnie przygotować prezentację i wygłosić referat.	K_U15, K_U20, K_U22
Potrafi ocenić przydatność prezentowanych rozwiązań algorytmicznych, optymalizacyjnych i kombinatorycznych i ich znaczenie dla innych działów informatyki.	K_U21

Kompetencje społeczne

Umie dostosować formę i treść prezentacji do wiedzy i umiejętności słuchaczy.	K_K05
Rozumie potrzebę ciągłego rozwoju i śledzenia postępów w badaniach algorytmicznych.	K_K01
Aktywnie prezentuje krytyczną postawę wobec stwierdzeń, uwag i wniosków, zwłaszcza niepopartych logicznym uzasadnieniem.	K_K06

16. Literatura obowiązkowa i zalecana

- David P. Williamson, David B. Shmoys: The Design of Approximation Algorithms, Cambridge University Press, 2011.
- artykuły czasopismowe i konferencyjne

17. Metody weryfikacji zakładanych efektów uczenia się

Opracowanie i przedstawienie prezentacji na zadany temat.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia seminarium wymagane jest przygotowanie i przeprowadzenie prezentacji na wybrany temat, ocenianej pod kątem poprawności merytorycznej i komunikatywności przekazu. Konieczne jest również aktywne uczestnictwo w dyskusji na zajęciach.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

przygotowanie raportu/prezentacji	20 godz.
studiowanie tematyki wykładów i literatury	15 godz.

Sumarycznie

Łączna liczba godzin	65 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium: Analiza problemów algorytmicznych** **Seminar: Analysis of algorithmic problems**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-AnPrbAlg**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **nieokreślony**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zalecane zrealizowane przedmioty** • Logika dla informatyków • Wstęp do informatyki (Lub doświadczenie zdobyte podczas przygotowań/udziału w Olimpiadzie Informatycznej i podobnych konkursach.)
13.	Cele kształcenia dla przedmiotu Podstawowym celem przedmiotu jest omawianie zadań i technik pojawiających się na różnego rodzaju zawodach programistycznych (indywidualnych: Olimpiada Informatyczna, Potyczki Algorytmiczne, rundy Codeforces, drużynowych: AMPPZ, CERC, ICPC World Finals). Oprócz zapoznania się uczestników ze wzorcowymi rozwiązaniami celem jest także refleksja nad poprawnością/złożonością tych rozwiązań, na co (zwykle) nie ma czasu podczas samych zawodów.
14.	Treści programowe Podczas zajęć omawianie są (między innymi) zadania z zawodów: 1. Olimpiada Informatyczna. 2. Potyczki Algorytmiczne. 3. Rundy Codeforces. 4. Akademickie Mistrzostwa Polski w Programowaniu Zespołowym. 5. Central European Regional Contest. 6. ICPC World Finals. Oprócz samych zadań i ich rozwiązań uczestnicy przedstawiają ogólne techniki oraz triki przydatne w ich sprawnej implementacji.

15.	Zakładane efekty uczenia się

Wiedza

zna podstawowe metody projektowania i analizy algorytmów	K_W07, K_W14

Umiejętności

potrafi formułować precyzyjną specyfikację problemu oraz przedstawić formalny dowód poprawności i złożoności wybranego rozwiązania	K_U21
potrafi przedstawić wybrane rozwiązanie i brać udział w dyskusji na jego temat	K_U15, K_U20, K_U22

16.	Literatura obowiązkowa i zalecana

Brak.

17.	Metody weryfikacji zakładanych efektów uczenia się

Prezentacja rozwiązania zadania, prezentacja ogólnej techniki.

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia seminarium konieczny jest aktywny udział w zajęciach (sugerowanie/omawianie zadań) lub przygotowanie i przedstawienie krótkiego wystąpienia na wybrany przez siebie temat.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

studiowanie omawianych zadań i literatury	45 godz.

Sumarycznie

Łączna liczba godzin	75 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium: Blockchain i jego zastosowania** **Seminar on Blockchain and its applications**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DSBLKCHN**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Ogólne obycie w świecie bezpieczeństwa IT.
13.	Cele kształcenia dla przedmiotu Celem seminarium jest zgłębienie technologii Blokchain i obszarów jej zastosowania w znacznie szerszym kontekście niż tylko w kryptowalutach.
14.	Treści programowe Tematyka seminarium jest związana z technologiią Blokchain i obszarami jej zastosowania rozumianymi w znacznie szerszym kontekście niż tylko w kryptowalutach.
15.	Zakładane efekty uczenia się **Wiedza** Zna dobrze wybrane zagadnienia technologii Blockchain — K_W15 **Umiejętności** Umie zaprojektować rozwiązanie dla wybranego aspektu — K_U19 Umie przedstawić wybrane zagadnienie w przystępny i zrozumiały sposób — K_U15, K_U20, K_U22 Umie znaleźć i opracować odpowiednie materiały dla wybranego tematu — K_U15, K_U20 Umie dobrać odpowiednie przykłady dla prezentowanego materiału — K_U15, K_U20, K_U22
16.	Literatura obowiązkowa i zalecana Artykuły dotyczące prezentowanych tematów.
17.	Metody weryfikacji zakładanych efektów uczenia się Opracowanie i przedstawienie prezentacji na wybrany temat, dyskusja, rozmowa z prowadzącym

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Zaliczenie na podstawie prezentacji z wybranego zagadnienia.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** seminarium — 30 godz. **Praca własna studenta** przygotowanie raportu/prezentacji — 15 godz. studiowanie tematyki wykładów i literatury — 30 godz. **Sumarycznie** Łączna liczba godzin — 75 godz. Liczba punktów ECTS — 3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium: Grafika komputerowa** **Seminar: Computer Graphics**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DS6GK**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Niezbędne kompetencje:** • znajomośc podstawowych zagadnień grafiki komputerowej
13.	Cele kształcenia dla przedmiotu Celem seminarium jest zapoznanie z rozwojem współczesnej grafiki komputerowej i nabycie umiejętności śledzenia postępów na bazie wiodących konferencji w dziedzinie. Przygotowanie prezentacji ma nauczyć czytania prac naukowych i zwięzłego prezentowania ich treści. Udział w dyskusji ma na celu doskonalenie umiejętności krytycznego myślenia i dzielenia się wiedzą.
14.	Treści programowe Najnowsze prace z szeroko rozumianych dziedzin grafiki komputerowej np.: • foto-realistyczne metody syntezy obrazów • architektura i programowanie kart graficznych (algorytmy i API) • metody modelowania scen 3D • własności powierzchni i funkcje BRDF • skanowanie 3D • wizja komputerowa, w tym metody przetwarzania obrazów.

15. Zakładane efekty uczenia się

Wiedza

zna postępy w wybranych poddziedzinach grafiki komputerowej	K_W15

Umiejętności

umie znaleść i porównać opublikowane na konferencjach rozwiązania zadanego problemu	K_U15, K_U20, K_U22
potrafi samodzielnie studiować nowe metody prezentowane na konferencjach	K_U15, K_U20, K_U22
umie prezentować prace naukowe i brać udział w dyskusji	K_U21
umie pracować i dyskutować w zespole także z osobami o mniejszej wiedzy na dany temat	K_U23

Kompetencje społeczne

rozumie znaczenie grafiki komputrowej dla wspólczesnego społeczeństwa	K_K04

16. Literatura obowiązkowa i zalecana

Prace z wiodących konferencji w dziedzinie grafiki komputerowej takich jak: SIGGRAPH, SIGGRAPH-Asia, EUROGRAPHICS
Odnośniki:

- http://www.siggraph.org
- http://www.eg.org
- http://kesen.realtimerendering.com

17. Metody weryfikacji zakładanych efektów uczenia się

- prezentacja
- udział w dyskusji

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia seminarium niezbędna jest obecność na zajęciach, udział w dyskusji i przygotowanie co najmniej jednej dłuższej prezentacji lub kilku na mniej obszerne tematy. Oceniane będzie zarówno pogłębiona znajomość tematu, jakość prezentacji, a także udział w dyskusjach.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

przygotowanie prezentacji	20 godz.
studiowanie prac przedstawianych i tematyki seminarium	25 godz.

Sumarycznie

Łączna liczba godzin	75 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium: Kompetytywna Sztuczna Inteligencja** **Seminar: Competitive Artificial Intelligence**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DSCompeAI**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Znajomość tematyki omawianej na przedmiocie Sztuczna Inteligencja (wymagane) oraz Artificial Intelligence for Games (zalecane).
13.	Cele kształcenia dla przedmiotu Znajomość istniejących zawodów programistycznych AI oraz stosowanych w nich algorytmów. Umiejętność wyboru odpowiedniego podejścia w zależności od struktury zadanego problemu.
14.	Treści programowe Zaawansowane algorytmy sztucznej inteligencji oraz ich rozmaite warianty, rozszerzenia i możliwości optymalizacji (MiniMax, MCTS, RHEA, BeamSearch/ChokudaiSearch, Transposition Tables, Opening Books, Pathfinding, ...). Zawody kompetetywnej sztucznej inteligencji, platformy oraz protokoły (CodinGame, IEEE COG competitions, AI Cup, Battlecode, Bomberlnd, Terminal, Lux AI, ...).

15.	Zakładane efekty uczenia się

Wiedza

Zna najnowsze badania w dzedzinie sztucznej inteligencji w grach, orientuje się w zaawansowanych algorytmach AI stosowanych w grach komputerowych oraz zna najważniejsze odbywające się w tematyce zawody	K_W15
Posiada przeglądową wiedzę na temat aktualnie prowadzonych badań w dziedzinie Game AI	K_W15

Umiejętności

Potrafi samodzielnie zrozumieć pracę naukową i zgłębić wybrany temat wyszukując informacje	K_U15, K_U20, K_U22
Potrafi przygotować i wygłosić prezentację dotyczącą wybranego tematu	K_U21

Kompetencje społeczne

Potrafi przeprowadzić krytyczną analizę wybranego tematu związanego z tematyką gier	K_K06
Potrafi brać udział w dyskusji, formułując swoje opinie dotyczące prezentowanego tematu	K_K06

16. Literatura obowiązkowa i zalecana

Wybrane prace naukowe z dziedziny, blogi uczestników zawodów, opisy zwycięskich programów, dokumentacja techniczna konkursów.

17. Metody weryfikacji zakładanych efektów uczenia się

- Opracowanie i przedstawienie prezentacji na zadany temat
- Dyskusja

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- Opracowanie wybranego tematu.
- Ocena wystąpienia seminaryjnego na temat zadanego materiału.
- Obecność na zajęciach i udział w dyskusjach.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

studiowanie wybranego temtu	25 godz.
przygotowanie prezentacji	15 godz.

Sumarycznie

Łączna liczba godzin	70 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium: Kryptografia i bezpieczeństwo komputerowe** **Seminar on Cryptography and Computer Security**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DSCrypSec**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu Podstawowa wiedza z zakresu kryptografii, preferowane zaliczenie przedmiotu o tej nazwie.
13.	Cele kształcenia dla przedmiotu Celem seminarium jest zgłębienie różnych aspektów kryptografii i okolic oraz jej zastosowanie z bezpieczeństwie komputerowym.
14.	Treści programowe 1. Cryptanalysis 2. Proxy re-encryption schemes 3. Remote Integrity Check 4. Batch cryptography 5. Homomorphic encryption 6. Blockchain and Smart Contracts 7. Steganography 8. Zero knowledge proofs 9. Kleptography 10. Review of latest articles from ESORICS, CRYPTO, EUROCRYPT and other top security conferences
15.	Zakładane efekty uczenia się **Wiedza** Zna dobrze wybrane zagadnienia z dziedziny bezpieczeństwa komputerowego — K_W15 **Umiejętności** Umie przedstawić wybrane zagadnienie w przystępny i zrozumiały sposób — K_U21 Umie znaleźć i opracować odpowiednie materiały dla wybranego tematu — K_U15, K_U20, K_U22 Umie dobrać odpowiednie przykłady dla prezentowanego materiału — K_U21, K_U22

16.	Literatura obowiązkowa i zalecana • Bashir, I. (2018). Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained. Packt Publishing Ltd. • Katz, J., Lindell, Y. (2014). Introduction to modern cryptography. CRC press. • Katzenbeisser, S., Petitcolas, F. A. (1999). Information Hiding Techniques for Steganography and Digital Watermarking, 2000 ARTECH HOUSE. Inc., Norwood, MA. • Krenn, S. (2012). Bringing zero-knowledge proofs of knowledge to practice. Logos Verlag • Norman, A. T. (2017). Blockchain Technology Explained: The Ultimate Beginners Guide About Blockchain Wallet, Mining, Bitcoin, Ethereum, Litecoin, Zcash, Monero, Ripple, Dash, IOTA And Smart Contracts. CreateSpace Independent Publishing Platform. • Raggo, M. T., Hosmer, C. (2012). Data hiding: exposing concealed data in multimedia, operating systems, mobile devices and network protocols. Newnes. • Schneier, B. (2007). Applied cryptography: protocols, algorithms, and source code in C. John Wiley Sons.
17.	Metody weryfikacji zakładanych efektów uczenia się Opracowanie i przedstawienie prezentacji na zadany temat
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu Opracowanie i przedstawienie prezentacji na zadany temat
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** seminarium — 30 godz. **Praca własna studenta** przygotowanie raportu/prezentacji — 15 godz. studiowanie tematyki wykładów i literatury — 30 godz. **Sumarycznie** Łączna liczba godzin — 75 godz. Liczba punktów ECTS — 3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Seminarium: Testowanie oprogramowania** **Seminar: Software Testing**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DS7TO**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Niezbędne kompetencje:** • Znajomość języków programowania: C++, Javy lub Pythona. • Zaliczenie przedmiotu Testowanie oprogramowania, ew. Testowanie gier lub aktualne zaliczanie tego przedmiotu Programowanie obiektowe.
13.	Cele kształcenia dla przedmiotu Celem seminarium jest rozszerzenie wiedzy z zakresu testowania oprogramowania.
14.	Treści programowe Zapoznanie studentów z nowoczesnymi, praktycznymi metodami i narzędziami testowania dotyczącymi następujących obszarów: • testy funkcjonalne, • testy niefunkcjonalne, • testowanie oprogramowania zorientowanego obiektowo, • testowanie oprogramowania internetowego, • testowanie oprogramowania mobilnego, • testowanie użyteczności, • testowanie gier, • testowanie integracyjne, • testowanie systemowe, • testowanie bezpieczeństwa, • testowanie wydajności, • strategie planowania testów, • efektywne zarządzanie testami, • zgodne z RODO testowanie oprogramowania.

15. Zakładane efekty uczenia się

Wiedza

Zna wybrane metody, techniki i narzędzia testowania oprogramowania oraz podstawy ich funkcjonowania	K_W15
Ma wiedzę dotyczącą społecznych i etycznych aspektów pracy informatyka, ze szczególnym uwzględnieniem zapewnienia poprawnego działania oprogramowania	K_W15

Umiejętności

Potrafi formułować opinie na temat procesu testowania oprogramowania, a także prowadzić dyskusję przedstawiając i oceniając różne opinie i stanowiska z tym związane	K_U21
Potrafi przygotować prezentacje dotyczące zaawansowanych zagadnień testowania oprogramowania i przedstawiać je osobom niebędącym specjalistami	K_U15, K_U21
Potrafi samodzielnie zdobywać wiedzę i rozwijać umiejętności zawodowe w obszarze metod, techniki i procesu testowania oprogramowania	K_U22

Kompetencje społeczne

Jest świadom możliwości popełniania błędów podczas programowania. Wykazuje rozważny krytycyzm wobec opinii na temat poprawnego działania oprogramowania	K_K01
Wykazuje gotowość do pełnienia, w sposób odpowiedzialny i respektujący zasady etyki zawodowej, roli testera oprogramowania	K_K03
Wykazuje samodzielność myślenia i działania przy rozwiązywaniu problemów i wykonywaniu zadań typowych dla testowania	K_K06

16. Literatura obowiązkowa i zalecana

1. Roman A.: Testowanie i jakość oprogramowania, PWN 2015
2. Roman A., Zmitrowicz K. (red): Testowanie w praktyce. Studium przypadków, PWN 2016
3. Zmitrowcz K.: Jakość projektów informatycznych. Rozwój i testowanie oprogramowania, Helion 2016
4. Smiglin R.: Zawód tester, Helion 2015
5. Pawlak R.: Testowanie oprogramowania, podręcznik dla początkujących, Helion 2014
6. Copeland D. A Practitioner's Guide to Software Test Design, Artech House 2004
7. Crispin L., Gregory C.: Agile Testing: A Practical Guide for Testers and Agile Teams, 2009

17. Metody weryfikacji zakładanych efektów uczenia się

Opracowanie i przedstawienie prezentacji na zadany temat

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Ocena wygłoszonych referatów (wykładów audiowizualnych) pod kątem poprawności merytorycznej i komunikatywności przekazu. Konieczne jest również aktywne uczestnictwo w dyskusji na zajęciach.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

studiowanie materiałów i opracowanie koncepcji prezentacji	20 godz.
przygotowanie prezentacji	15 godz.

Sumarycznie

Łączna liczba godzin	65 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Spectral Graph Theory** **Spectral Graph Theory**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **angielski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DSSpeGraph**
6.	Rodzaj przedmiotu/modułu **Seminarium**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin seminarium — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Corses followed:** • Theory of Linear and Integer Programming • or, Introduction to Linear Optimization **Recommended prior knowledge in:** • Linear Algebra • Basic Graph Theory
13.	Cele kształcenia dla przedmiotu • To expose students to the results in spectral graph theory • To train reading advanced mathematical theory • To exercise presenting selected advanced results
14.	Treści programowe • eigenvalues • laplacians • ramdom walks • effective resistance • random trees

15.	Zakładane efekty uczenia się

Knowledge

Has basic knowledge in Eigenvalues Optimization	K_W02, K_W14
Understands Graph Laplacians	K_W14
Is familiar with Random Walks in Graphs	K_W13, K_W14
Knows the concept of Effective Resistance	K_W14
Understands basic methods for sampling Random Spanning Trees	K_W14

Skills

Can read and learn advanced mathematical content	K_U15, K_U20, K_U22
Can present selected constructs and results to other students	K_U21

Social Skills

Can udjust the level of presentation to the knowledge of the audience	K_K05

16. Literatura obowiązkowa i zalecana

`http://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf`

17. Metody weryfikacji zakładanych efektów uczenia się

Seminar presentation

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Seminar presentation

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

seminarium	30 godz.

Praca własna studenta

additional consultations — if necessary	0 godz.
literature study	30 godz.
preparing individual presentation of a selected topic	20 godz.

Sumarycznie

Łączna liczba godzin	80 godz.
Liczba punktów ECTS	3

7 Pozostałe zajęcia

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Etyka dla myślących** **Ethics for the thinking**
2.	Dyscyplina **filozofia**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-EtkdMsl**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu • Zapoznanie studenta z najważniejszymi koncepcjami etyki normatywnej; • Wdrożenie studenta do posługiwania się pojęciami etycznymi potrzebnymi do zbudowania świadomego światopoglądu moralnego; • Wyrobienie umiejętności krytycznej lektury tekstu filozoficznego; • Wyrobienie umiejętności racjonalnej argumentacji na rzecz przyjętej hipotezy; • Doskonalenie umiejętności pisemnego wypowiadania się na tematy o wysokim stopniu abstrakcyjności.

14.	Treści programowe W trakcie wykładów zostaną omówione najważniejsze szkoły i koncepcje etyki filozoficznej ze szczególnym podkreśleniem wybranego pojęcia moralnego. Ćwiczenia będą polegały na krytycznym dyskutowaniu koncepcji omawianych na wykładach. 1. Wprowadzenie do etyki jako dziedziny filozoficznej. Etyka a moralność, prawo i obyczaj. Etyka opisowa, normatywna i metaetyka. Pojęcie moralności. 2. Początki refleksji moralnej: Homer i Sokrates. Pojęcie honoru. 3. Platon – sprawiedliwy człowiek, sprawiedliwe państwo. Pojęcie sprawiedliwości. 4. Arystoteles – etyka złotego środka. Pojęcie cnoty. 5. Filozofowie w poszukiwaniu szczęścia: stoicyzm, epikureizm, cynizm, hedonizm. Pojęcie szczęścia. 6. Wiara i rozum – św. Augustyn i św. Tomasz z Akwinu. Pojęcie sumienia. 7. Uczucia moralne – Hume. Pojęcie tolerancji. 8. Etyka umowy społecznej. Pojęcie godności. 9. Imperatyw kategoryczny Kanta. Pojęcie obowiązku. 10. Zasada utylitaryzmu. Pojęcie wolności. 11. Egzystencjalizm. Nietzsche i Scheler. Pojęcie sensu życia. 12. Szkoła lwowsko-warszawska. Kotarbiński i Czeżowski. Pojęcie dylematu moralnego. 13. Elzenberg i etyka wartości. Pojęcie wartości. 14. Sens i znaczenie ocen i norm moralnych. Główne kierunki metaetyki. Pojęcie normy i oceny moralnej. 15. Klasyfikacja stanowisk etycznych. Podsumowanie.
15.	Zakładane efekty uczenia się **Wiedza** Student ma podstawową wiedzę z zakresu etyki filozoficznej Student zna podstawową terminologię etyczną Student zna i rozumie rolę etyki w kształtowaniu świadomych postaw moralnych. **Umiejętności** Student potrafi przeczytać, przedyskutować i skategoryzować dany tekst etyczny Student potrafi poprawnie rozpoznać i zastosować wybraną terminologię etyczną Student potrafi racjonalnie wypowiedzieć się słownie oraz pisemnie na temat wybranych zagadnień moralnych. **Kompetencje społeczne** Student jest zdolny do samodzielnego budowania systemu własnych poglądów i formułowania krytycznych ocen Student dzięki posiadanej wiedzy i umiejętnościom etycznym rozumie potrzebę ciągłego rozwoju osobistego Student ma świadomość znaczenia europejskiego dziedzictwa etycznego i potrafi je docenić.

16. Literatura obowiązkowa i zalecana

Literatura podstawowa:

- M. Środa, Etyka dla myślących. Podręcznik dla szkół ponadgimnazjalnych, Warszawa 2016.
- Platon, Obrona Sokratesa (różne wydania).
- Platon, Pańswo, ks. 4 i 7.
- Arystoteles, Etyka nikomachejska, ks. 2-3.
- M. Aureliusz, Rozmyślania, ks. 2-5
- św. Augustyn, Wyznania ks. 7,5, Tomasz z Akwinu, Summa, kw. 80, 83 i 90.
- D. Hume, Badania dotyczące zasad moralności, cz. 1-5.
- T. Hobbes, Lewiatan, rozdz. 10-21, J. Rawls, Teoria sprawiedliwości, par. 1-4, 11-17, 62-63.
- I. Kant, Uzasadnienie metafizyki moralności, rozdz. 1-2.
- J. S. Mill, Utylitaryzm.
- F. Nietzsche, Z genealogii moralności, cz. I.
- H. Elzenberg, Wartość i powinność, w: O wartościach, normach i problemach moralnych, (red.) M. Środa, Warszawa 1994.
- T. Czeżowski, Aksjologiczne i deontyczne normy moralne, w: O wartościach, normach i problemach moralnych, (red.) M. Środa, Warszawa 1994.
- A. Pap, Teorie metaetyczne, w: Metaetyka, (red.) I. Lazari-Pawłowska, Warszawa 1975.

Literatura dodatkowa:

- A. MacIntyre, Krótka historia etyki, Warszawa 2013.
- P. Vardy, P Grosch, Etyka, Poglądy i problemy, Poznań 2010.

17. Metody weryfikacji zakładanych efektów uczenia się

- egzamin ustny,
- przygotowywanie wypowiedzi pisemnych.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- Wykład: egzamin ustny (trzy pytania na podstawie przygotowanych uprzednio przez studenta konspektów lektur źródłowych). Obecność.
- Ćwiczenia: obecność oraz przedstawianie pięciu pisemnych wypowiedzi na zadane do zajęć pytania.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
ćwiczenia	30 godz.

Praca własna studenta

przygotowanie do zajęć (pisanie wypowiedzi pisemnych)	30 godz.
przygotowanie do egzaminu (czytanie wskazanej literatury)	30 godz.

Sumarycznie

Łączna liczba godzin	120 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Etyka dla myślących (w)** **Ethics for the thinking**
2.	Dyscyplina **filozofia**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-EtkdMsl**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu • Zapoznanie studenta z najważniejszymi koncepcjami etyki normatywnej; • Wdrożenie studenta do posługiwania się pojęciami etycznymi potrzebnymi do zbudowania świadomego światopoglądu moralnego; • Wyrobienie umiejętności krytycznej lektury tekstu filozoficznego; • Wyrobienie umiejętności racjonalnej argumentacji na rzecz przyjętej hipotezy; • Doskonalenie umiejętności pisemnego wypowiadania się na tematy o wysokim stopniu abstrakcyjności.

14.	Treści programowe W trakcie wykładów zostaną omówione najważniejsze szkoły i koncepcje etyki filozoficznej ze szczególnym podkreśleniem wybranego pojęcia moralnego. Ćwiczenia będą polegały na krytycznym dyskutowaniu koncepcji omawianych na wykładach. 1. Wprowadzenie do etyki jako dziedziny filozoficznej. Etyka a moralność, prawo i obyczaj. Etyka opisowa, normatywna i metaetyka. Pojęcie moralności. 2. Początki refleksji moralnej: Homer i Sokrates. Pojęcie honoru. 3. Platon – sprawiedliwy człowiek, sprawiedliwe państwo. Pojęcie sprawiedliwości. 4. Arystoteles – etyka złotego środka. Pojęcie cnoty. 5. Filozofowie w poszukiwaniu szczęścia: stoicyzm, epikureizm, cynizm, hedonizm. Pojęcie szczęścia. 6. Wiara i rozum – św. Augustyn i św. Tomasz z Akwinu. Pojęcie sumienia. 7. Uczucia moralne – Hume. Pojęcie tolerancji. 8. Etyka umowy społecznej. Pojęcie godności. 9. Imperatyw kategoryczny Kanta. Pojęcie obowiązku. 10. Zasada utylitaryzmu. Pojęcie wolności. 11. Egzystencjalizm. Nietzsche i Scheler. Pojęcie sensu życia. 12. Szkoła lwowsko-warszawska. Kotarbiński i Czeżowski. Pojęcie dylematu moralnego. 13. Elzenberg i etyka wartości. Pojęcie wartości. 14. Sens i znaczenie ocen i norm moralnych. Główne kierunki metaetyki. Pojęcie normy i oceny moralnej. 15. Klasyfikacja stanowisk etycznych. Podsumowanie.
15.	Zakładane efekty uczenia się **Wiedza** Student ma podstawową wiedzę z zakresu etyki filozoficznej Student zna podstawową terminologię etyczną Student zna i rozumie rolę etyki w kształtowaniu świadomych postaw moralnych. **Umiejętności** Student potrafi przeczytać, przedyskutować i skategoryzować dany tekst etyczny Student potrafi poprawnie rozpoznać i zastosować wybraną terminologię etyczną Student potrafi racjonalnie wypowiedzieć się słownie oraz pisemnie na temat wybranych zagadnień moralnych. **Kompetencje społeczne** Student jest zdolny do samodzielnego budowania systemu własnych poglądów i formułowania krytycznych ocen Student dzięki posiadanej wiedzy i umiejętnościom etycznym rozumie potrzebę ciągłego rozwoju osobistego Student ma świadomość znaczenia europejskiego dziedzictwa etycznego i potrafi je docenić.

16.	Literatura obowiązkowa i zalecana **Literatura podstawowa:** • M. Środa, Etyka dla myślących. Podręcznik dla szkół ponadgimnazjalnych, Warszawa 2016. • Platon, Obrona Sokratesa (różne wydania). • Platon, Pańswo, ks. 4 i 7. • Arystoteles, Etyka nikomachejska, ks. 2-3. • M. Aureliusz, Rozmyślania, ks. 2-5 • św. Augustyn, Wyznania ks. 7,5, Tomasz z Akwinu, Summa, kw. 80, 83 i 90. • D. Hume, Badania dotyczące zasad moralności, cz. 1-5. • T. Hobbes, Lewiatan, rozdz. 10-21, J. Rawls, Teoria sprawiedliwości, par. 1-4, 11-17, 62-63. • I. Kant, Uzasadnienie metafizyki moralności, rozdz. 1-2. • J. S. Mill, Utylitaryzm. • F. Nietzsche, Z genealogii moralności, cz. I. • H. Elzenberg, Wartość i powinność, w: O wartościach, normach i problemach moralnych, (red.) M. Środa, Warszawa 1994. • T. Czeżowski, Aksjologiczne i deontyczne normy moralne, w: O wartościach, normach i problemach moralnych, (red.) M. Środa, Warszawa 1994. • A. Pap, Teorie metaetyczne, w: Metaetyka, (red.) I. Lazari-Pawłowska, Warszawa 1975. **Literatura dodatkowa:** • A. MacIntyre, Krótka historia etyki, Warszawa 2013. • P. Vardy, P Grosch, Etyka, Poglądy i problemy, Poznań 2010.
17.	Metody weryfikacji zakładanych efektów uczenia się • egzamin ustny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu • Wykład: egzamin ustny (trzy pytania na podstawie przygotowanych uprzednio przez studenta konspektów lektur źródłowych). Obecność.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** wykład — 30 godz. **Praca własna studenta** przygotowanie do egzaminu (czytanie wskazanej literatury) — 40 godz. **Sumarycznie** Łączna liczba godzin — 70 godz. Liczba punktów ECTS — 3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Historia filozofii** **History of Philosophy**
2.	Dyscyplina **filozofia**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-HistFiloz**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest kształtowanie umiejętności samodzielnego myślenia na kanwie wiedzy zdobytej podczas wykładów i umiejętności nabytych na ćwiczeniach, kształtowanie wrażliwości na różnego rodzaju argumentację obecną w całości antroposfery, rozwijanie umiejętności krytycznego myślenia, praktyczne przygotowanie do wstępnej, samodzielnej lektury tekstów z zakresu historii filozofii.

14.	Treści programowe

1. Charakterystyka filozofii i jej subdyscyplin, geneza nazwy.
2. Presokratycy, sofiści, Sokrates – poszukiwanie przedmiotu filozofii.
3. Platon i Arystoteles – systemy świata starożytnego.
4. Stoicy i epikurejczycy – charakterystyka myśli hellenistycznej.
5. Wczesne średniowiecze i św. Augustyn.
6. Scholastyka i św. Tomasz.
7. Renesans i początek nowożytności: Machiavelli, Pico della Mirandola, Hobbes.
8. Kartezjusz – „ojciec filozofii nowożytnej".
9. Leibniz i Spinoza: wokół substancji; lub Locke, Berkeley, Hume: wokół percepcji.
10. Kant - „przewrót kopernikański" i Schopenhauer - „świat jako wola i wyobrażenie"
11. Hegel: historia i dialektyka oraz krytycy jego filozofii zapośredniczeni w niej: Kierkegaard, Marks, Feuerbach.
12. Pozytywizm i filozofia nauki – Comte, Mach, Koło Wiedeńskie, Wittgenstein, Kuhn, Popper.
13. Różne oblicza współczesności: Fenomenologia i pragmatyzm; filozofia chrześcijańska i marksizm; poststrukturalizm i personalizm; postmodernizm i dekonstrukcjonizm.
14. Wokół egzystencjalizm: Szestow, Heidegger, Jaspers, Sartre'a; lub filozofia życia: Nietzsche, Dilthey, Bergson.
15. Cum grano salis: wokół „partii matematycznej" w filozofii: o pitagorejczykach, Platonie, Kartezjuszu, Spinozie, Kancie, Whiteheadzie, Russellu... pod kątem ich przekonania o możliwość wykorzystania rozwiązań matematycznych do rozstrzygnięć ontologicznych i metafizycznych, jak również etycznych i politycznych.

15. Zakładane efekty uczenia się

Wiedza

Posiada podstawową wiedzę o specyfice i roli filozofii oraz podstawowych zmianach w obrębie poruszanej przez nią problematyki od starożytności do współczesności Charakteryzuje różne - omawiane na ćwiczeniach i wykładzie - koncepcje filozoficzne; wymienia najbardziej charakterystyczne dla nich założenia	

Umiejętności

Potrafi analizować tekst, a także interpretować problemy w nim zawarte w szerszym tle odniesień do kultury współczesnej i własnej wrażliwości Stosuje poprawnie podstawowe pojęcia z zakresu filozofii	

Kompetencje społeczne

Przejawia otwartość i wrażliwość na różnego rodzaju argumentacje i dostrzega ograniczenia różnego typu argumentacji, zwłaszcza tych które uznaje za optymalne	

16.	Literatura obowiązkowa i zalecana

Literatura obowiązkowa do ćwiczeń:

1. Obraz filozofii i filozofa: „Obrona Sokratesa” Platona
2. Wokół filozofii starożytnego świata: Wybrane fragmenty z „Państwa” Platona (rozdział 7) i „Metafizyki” Arystetelesa z ksiąg II i IV
3. Wokół filozofii średniowiecznego świata: Wybrane fragmenty z Augustyna „Solilokwiów” (rozdział 2,6) i Tomasza „O królowaniu” (rozdział 13-15)
4. Dylematy nowożytności: Wybrany - przez studentów - jeden blok: 1.) „Rozprawy o metodzie” Descartes'a i „Etyki” Spinozy lub 2.) Pascala „Myśli” lub 3.) Wolter „Kandyd, czyli optymizm” lub 4.) Locke „List o tolerancji”
5. Systemy nowożytnego świata: Wybrany - przez studentów - jeden blok: 1.) fragmenty z „Monadologii” Leibniza lub 2.) fragmenty z „Prolegomeny” Kanta lub 3.) fragmenty z „Nauki logiki” Hegla
6. Różne oblicza współczesności: Wybrany - przez studentów - jeden blok 1.) Kuhn „Struktura rewolucji naukowych” (roz. V „Priorytet paradygmatów” lub 2.) Fromm „Ucieczka od wolności” (roz. IV „Dwa aspekty wolności człowieka współczesnego”) lub 3.) J. P. Sartre „Byt i Nicość” (Wprowadzenie: W poszukiwaniu bytu) lub 4.) Th. W. Adorno „Teoria estetyczna” („Wczesne wprowadzenie”)
7. O filozofii, nauce, matematyce i życiu na wesoło: H. Steinhaus „Słownik racjonalny” Literatura uzupełniająca (do wykładu) wybrany podręcznik:
8. W. Tatarkiewicz „Historia filozofii” t. 1-3
9. S. Swieżawski „Dzieje europejskiej filozofii klasycznej”
10. B. Russell „Dzieje filozofii zachodu”

17.	Metody weryfikacji zakładanych efektów uczenia się

egzamin ustny, dyskusja, praca pisemna

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Ćwiczenia: obecność, aktywność (ocenianie ciągłe), na ocenę bdb. - miniesej 2-3 strony.
Wykład: egzamin ustny weryfikujący opanowanie wiedzy i umiejętności.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
ćwiczenia	30 godz.

Praca własna studenta

przygotowanie do ćwiczeń	30 godz.
studiowanie tematyki wykładów i literatury	30 godz.
przygotowanie do egzaminu	10 godz.
przygotowanie raportu/prezentacji	10 godz.

Sumarycznie

Łączna liczba godzin	140 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Historia filozofii (w)** **History of Philosophy**
2.	Dyscyplina **filozofia**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-HistFiloz**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest kształtowanie umiejętności samodzielnego myślenia na kanwie wiedzy zdobytej podczas wykładów i umiejętności nabytych na ćwiczeniach, kształtowanie wrażliwości na różnego rodzaju argumentację obecną w całości antroposfery, rozwijanie umiejętności krytycznego myślenia, praktyczne przygotowanie do wstępnej, samodzielnej lektury tekstów z zakresu historii filozofii.

14.	Treści programowe

1. Charakterystyka filozofii i jej subdyscyplin, geneza nazwy.
2. Presokratycy, sofiści, Sokrates – poszukiwanie przedmiotu filozofii.
3. Platon i Arystoteles – systemy świata starożytnego.
4. Stoicy i epikurejczycy – charakterystyka myśli hellenistycznej.
5. Wczesne średniowiecze i św. Augustyn.
6. Scholastyka i św. Tomasz.
7. Renesans i początek nowożytności: Machiavelli, Pico della Mirandola, Hobbes.
8. Kartezjusz – „ojciec filozofii nowożytnej".
9. Leibniz i Spinoza: wokół substancji; lub Locke, Berkeley, Hume: wokół percepcji.
10. Kant - „przewrót kopernikański" i Schopenhauer - „świat jako wola i wyobrażenie"
11. Hegel: historia i dialektyka oraz krytycy jego filozofii zapośredniczeni w niej: Kierkegaard, Marks, Feuerbach.
12. Pozytywizm i filozofia nauki – Comte, Mach, Koło Wiedeńskie, Wittgenstein, Kuhn, Popper.
13. Różne oblicza współczesności: Fenomenologia i pragmatyzm; filozofia chrześcijańska i marksizm; poststrukturalizm i personalizm; postmodernizm i dekonstrukcjonizm.
14. Wokół egzystencjalizm: Szestow, Heidegger, Jaspers, Sartre'a; lub filozofia życia: Nietzsche, Dilthey, Bergson.
15. Cum grano salis: wokół „partii matematycznej" w filozofii: o pitagorejczykach, Platonie, Kartezjuszu, Spinozie, Kancie, Whiteheadzie, Russellu... pod kątem ich przekonania o możliwość wykorzystania rozwiązań matematycznych do rozstrzygnięć ontologicznych i metafizycznych, jak również etycznych i politycznych.

15. Zakładane efekty uczenia się

Wiedza

Posiada podstawową wiedzę o specyfice i roli filozofii oraz podstawowych zmianach w obrębie poruszanej przez nią problematyki od starożytności do współczesności Charakteryzuje różne - omawiane na ćwiczeniach i wykładzie - koncepcje filozoficzne; wymienia najbardziej charakterystyczne dla nich założenia	

Umiejętności

Potrafi analizować tekst, a także interpretować problemy w nim zawarte w szerszym tle odniesień do kultury współczesnej i własnej wrażliwości Stosuje poprawnie podstawowe pojęcia z zakresu filozofii	

Kompetencje społeczne

Przejawia otwartość i wrażliwość na różnego rodzaju argumentacje i dostrzega ograniczenia różnego typu argumentacji, zwłaszcza tych które uznaje za optymalne	

16.	Literatura obowiązkowa i zalecana Literatura obowiązkowa do ćwiczeń: 1. Obraz filozofii i filozofa: „Obrona Sokratesa” Platona 2. Wokół filozofii starożytnego świata: Wybrane fragmenty z „Państwa” Platona (rozdział 7) i „Metafizyki” Arystetelesa z ksiąg II i IV 3. Wokół filozofii średniowiecznego świata: Wybrane fragmenty z Augustyna „Solilokwiów” (rozdział 2,6) i Tomasza „O królowaniu” (rozdział 13-15) 4. Dylematy nowożytności: Wybrany - przez studentów - jeden blok: 1.) „Rozprawy o metodzie” Descartes'a i „Etyki” Spinozy lub 2.) Pascala „Myśli” lub 3.) Wolter „Kandyd, czyli optymizm” lub 4.) Locke „List o tolerancji” 5. Systemy nowożytnego świata: Wybrany - przez studentów - jeden blok: 1.) fragmenty z „Monadologii” Leibniza lub 2.) fragmenty z „Prolegomeny” Kanta lub 3.) fragmenty z „Nauki logiki” Hegla 6. Różne oblicza współczesności: Wybrany - przez studentów - jeden blok 1.) Kuhn „Struktura rewolucji naukowych” (roz. V „Priorytet paradygmatów” lub 2.) Fromm „Ucieczka od wolności” (roz. IV „Dwa aspekty wolności człowieka współczesnego”) lub 3.) J. P. Sartre „Byt i Nicość” (Wprowadzenie: W poszukiwaniu bytu) lub 4.) Th. W. Adorno „Teoria estetyczna” („Wczesne wprowadzenie”) 7. O filozofii, nauce, matematyce i życiu na wesoło: H. Steinhaus „Słownik racjonalny” Literatura uzupełniająca (do wykładu) wybrany podręcznik: 8. W. Tatarkiewicz „Historia filozofii” t. 1-3 9. S. Swieżawski „Dzieje europejskiej filozofii klasycznej” 10. B. Russell „Dzieje filozofii zachodu”
17.	Metody weryfikacji zakładanych efektów uczenia się egzamin ustny
18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu egzamin ustny weryfikujący opanowanie wiedzy i umiejętności.
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** wykład — 30 godz. **Praca własna studenta** studiowanie tematyki wykładów i literatury — 30 godz. przygotowanie do egzaminu — 10 godz. przygotowanie raportu/prezentacji — 10 godz. **Sumarycznie** Łączna liczba godzin — 80 godz. Liczba punktów ECTS — 3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Idee post-humanizmu: projekt sztucznej inteligencji i problem możliwych umysłów** **Ideas of Post-Humanism: the Artificial Intelligence Project and the Problem of Possible Minds**
2.	Dyscyplina **filozofia**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DHSPostHum**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zaznajomienie słuchaczy podstawowymi ideami określającymi nurt post-humanizmu, który pojawił się wraz programem naturalizacji humanistyki. W tym podstaw rozumienia kultury, relacji społecznych umysłu i świadomości w kontekście różnych nurtów nauk kognitywnych.

14.	Treści programowe

1. Metafora komputera w badaniach nad umysłem i świadomością oraz jej konsekwencje dla rozumienia kultury i człowieka.
2. Post-humanizm czy humanizm rozszerzony – komu lub czemu możemy przypisać cechę umysłu.
3. Kultura jako rzeczywistość wirtualna: problem znaczenia, sensu i komunikacji w teorii informacji.
4. Humanistyka jako artefakt kultury w świecie technologii informacyjnych.
5. Mózg Boltzmanna i inne eksperymenty myślowe czyli rzeczywistość z punktu widzenia możliwego umysłu.
6. Umysł świadomy czyli jaki? Stopniowalność świadomości i idea panpsychizmu.
7. Czym jest świadomość – symulatorem czy symulacją: pomieszanie porządków logicznych i przyczynowo – skutkowych.
8. Zakres pojęcia świadomości: perspektywa pierwszo-osobowa i perspektywa trzecio-osobowa. Świat prywatnych fenomenów a rzeczywistość obiektywna i społeczna.
9. Świadomość refleksyjna i jej rola w procesach przetwarzania informacji o świecie. Problem podmiotu w naukach kognitywnych.
10. Idea odwróconej inżynierii: jeśli potrafimy skonstruować świadomy umysł to wiemy, czym on jest.
11. Świadomość i tożsamość: czy jesteśmy tym, kim myślimy, że jesteśmy?
12. Wielorakość tożsamości graczy i aktorów: świat społeczny jako symulacja ról.
13. Idea persony w klasycznej humanistyce i jako rola do odegrania.
14. Umysł i świadomość poza ludzkim mózgiem: problem A.I. i G.A.I. Systemy przetwarzania informacji i problem kontroli nad zachowaniami społecznymi. Dystopia końca ludzkości.

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe idee określające nurt post-humanizmu zna podstawy rozumienia kultury, relacji społecznych umysłu i świadomości w kontekście różnych nurtów nauk kognitywnych	

16. Literatura obowiązkowa i zalecana

Literatura obowiązkowa:

1. Bostrom N. (2016). Superinteligencja. Scenariusze, strategie, zagrożenia, Warszawa: Helion.
2. Chalmers D. (2010). Świadomy umysł, Warszawa: PWN.
3. Clark A., Chalmers D.J. (2008). Umysł rozszerzony, w: Analityczna metafizyka umysłu. Najnowsze kontrowersje, M. Miłkowski, R. Poczobut, Warszawa: IFiS PAN.
4. Dennett C.D. (2005). Słodkie sny. Filozoficzne przeszkody na drodze do nauki o świadomości, Poznań: Prószyński i S-ka.
5. Gazzaniga M.S. (2011). Istota człowieczeństwa. Co sprawia że jesteśmy wyjątkowymi, Sopot: Smak słowa, cz. II, W świecie społecznym.
6. Gleick J., Informacja. Bit, wszechświat, rewolucja, Kraków 2012.
7. Marcus G. Prowizorka w mózgu, Sopot: Smak słowa, rodz. 2, 4, 5, Pamięć, Dokonywanie wyborów, Język
8. Searle J.R. (2010). Umysł. Krótkie wprowadzenie, Poznań: Rebis, rozdz. 6, 9, 11, Intencjonalność, Nieświadomość i wyjaśnianie zachowania, Ja.

Literatura zalecana

- Damasio A.R. (2000). Tajemnica i świadomość. Jak ciało i emocje współtworzą świadomość, Poznań: Rebis.
- Dennett D.C., Od bakterii do Bacha. O ewolucji umysłów, tłum. K. Bielecka, M. Miłkowski, Copernicus Center Press, Warszawa 2017.
- Dennett D.C., Dźwignie wyobraźni i inne narzędzia do myślenia, tłum. Ł. Kurek, Copernicus Center Press, Warszawa 2015.
- Gamble Clive, J. Gowlett, R. Dunbar, Potęga mózgu. Jak ewolucja życia społecznego kształtowała ludzki umysł, tłum. R. Kosarzycki, Copernicus Center Press, Kraków 2017.
- Godfrey-Smith Peter, Inne umysły. Ośmiornice i prapoczątki świadomości, Copernicus Center Press, Kraków 2019
- Majchrowicz B., J. Doboszewski, T. Placek, Droga donikąd? Co pozostało z eksperymentów Libeta nad wolną wolą, w: „Filozofia nauki", Rok XXIV, 2016, nr 2(94).

17. Metody weryfikacji zakładanych efektów uczenia się

udział w dyskusji, praca semestralna, egzamin

18.	Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu obecność na ćwiczeniach, praca semestralna, egzamin
19.	Nakład pracy studenta **Zajęcia z udziałem nauczyciela** wykład: 30 godz. ćwiczenia: 30 godz. **Praca własna studenta** studiowanie tematyki wykładów i literatury: 40 godz. przygotowanie do egzaminu: 20 godz. **Sumarycznie** Łączna liczba godzin: 120 godz. Liczba punktów ECTS: 5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Manipulacje i nadużycia na rynku finansowym** **Manipulation and abuses on the financial market**
2.	Dyscyplina **ekonomia i finanse**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DHSManINad**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin wykład — 30 godzin, ćwiczenia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu 1. Zapoznanie studenta z różnymi formami i postaciami szeroko rozumianych nadużyć i manipulacji na rynku finansowym w Polsce. 2. Wyposażenie studenta w wiedzę na temat przestępstw na rynku finansowym. 3. Zapoznanie studenta z różnego rodzaju nadużyciami instytucji finansowych w relacjach umownych z klientami (z niebezpiecznymi , strukturyzowanymi i skomplikowanymi produktami finansowymi o dużym ryzyku oraz nieuczciwymi praktykami rynkowymi wprowadzającymi w błąd, praktykami naruszających zbiorowe interesy konsumentów, missellingiem). 4. Wyposażenie studenta w wiedzę na temat zagrożeń wynikających z zastosowania nowych technologii oraz umiejętność ich diagnozowania i unikania ich negatywnych skutków.

14.	Treści programowe 1. Formy i przejawy szeroko rozumianych nadużyć i manipulacji na rynku finansowym w Polsce 2. Stosowanie niedozwolonych klauzul umownych w konsumenckich umowach o usługi finansowe jako zakazana praktyka konsumencka (decyzje Prezesa UOKIK w tych sprawach i nakładane kary) 3. Praktyki naruszające zbiorowe interesy konsumentów na rynku usług finansowych (decyzje Prezesa UOKIK w tych sprawach i nakładane kary) 4. Nieuczciwe praktyki rynkowe na rynku usług finansowych. Manipulowanie informacją o odpłatności za usługi finansowe 5. Czyny nieucziwej konkurencji na rynku finansowym 6. Nadużycia w relacjach umownych instytucji finansowych z konsumentami (problem niebezpiecznych, hybrydowych, strukturyzowanych produktów finansowych zagrożonych dużym ryzkiem, polisolokaty, "kredyty frankowe", rabatowanie kosztów kredytu dla konsumentów) 7. Misselling na rynku usług finansowych 8. Nadużycia związane w przeprowadzaniem transakcji płatniczych (omyłkowy przelew na złe konto-procedura zwrotu pieniędzy, ataki phishingowe na klientów banku, wykonanie transakcji nieautoryzowanej, nienależyte wykonanie transakcji 9. Nowe technologie na rynku finansowym i problem cyberbezpieczeństwa. Kradzież tożsamości i i inne fraudy. 10. Zjawisko prania brudnych pieniędzy i mechanizmy jego przeciwdziałanie 11. Nadużycia na rynku kapitałowym (manipulacje instrumentami finansowymi i insider tranding), naduzycia na rynku obligacji 12. Prowadzenie działalności na rynku finansowym bez zezwolenia lub wpisu do rejestru. 13. Piramidy finansowe 14. Przestępstwa na rynku finansowym w ujęciu ogólnym wynikające z przepisów karnych ustaw sektorowych 15. Przeciwdziałanie nadużyciom na rynku finansowym i mechanizmy usuwania ich skutków (mechanizmy karne, cywilnoprawne, administracyjnoprawne). Mechanizmy rozpatrywania reklamacji i sporów (dochodzenie indywidualnych roszczeń konsumenckich wyikajacych z nadużyć) 16. Ochrona konsumentów przed nadużyciami na rynku finansowym i rola organów ochrony.
15.	Zakładane efekty uczenia się **Wiedza** zna różne formy i postacie nadużyć na rynku finansowym potrafi identyfikować zachowania nielegalne instytucji finansowych oraz pomiotów podejmujących działalność na tym rynku bez wymaganego zezwolenia zna, definiuje różne formy naruszeń instytucji finansowych wobec klientów w postaci nieuczciwych praktyk rynkowych, praktyk naruszających zbiorowe interesy konsumentów, missellingu, stosowania klauzul abuzywnych **Umiejętności** potrafi zdiagnozować zagrożenia związane z manipulowaniem informacją na rynku finansowym i określać charakter tych naruszeń i ich skutki prawne posługuje się aparatem pojęciowym w zakresie nadużyć i manipulacji na rynku finansowym **Kompetencje społeczne** potrafi identyfikować niebezpieczne, skomplikowane produkty finansowe zna różne formy przestępstw i innych nadużyć na rynku finansowym

16. Literatura obowiązkowa i zalecana

Literatura Podstawowa

1. Raport Doradczego Komitetu Naukowego przy Rzeczniku Finansowym: NIEPRAWIDŁOWOŚCI NA RYNKU FINANSOWYM A OCHRONA KONSUMENTA, Warszawa, wrzesień 2019 r., Doradczy Komitet Naukowy Rzecznika Finansowego, `https://rf.gov.pl/pdf/DKN_Raport_nieprawidlowosci_wrzesien2019.pdf.2019`
2. Manipulacje i oszustwa na rynku finansowym. Perspektywa konsumenta. Wykrywanie, przeciwdziałanie, zapobieganie , Monkiewicz J., Jurkowska-Zeidler A., Rutkowska- Tomaszewska E., Wiktorow A. , Wydawnictwo Naukowe PWN . Warszawa 2020
3. Rynek finansowy. Zapobieganie przyczynom przestępczości (PDF), Blicharz G., Oręziak B., Wielec M. , WYDAWNICTWO INSTYTUTU WYMIARU SPRAWIEDLIWOŚCI. Warszawa 2021

Literatura Uzupełniająca

1. Wybrane przestępstwa na rynku finansowym , Jaroszewicz P., Wojciechowska-Mytych K., Komisja Nadzoru Finansowego. Warszawa 2015
2. Manipulacja instrumentami finansowymi i insider trading. Analiza prawo-ekonomiczna, Martysz Cz. , Wolters Kluwer Polska . Warszawa 2015
3. Obowiązki banków wObec klientów przewidziane w ustawie o obrocie instrumentami finansowymi, , Majewski K., KNF. Warszawa 2019
4. OTWARTA BANKOWOŚĆ W ŚWIETLE WYMOGÓW DYREKTYWY PSD2 – wyzwania i perspektywy rozwoju dla polskiego sektora FinTech, (PDF), Leżoń K., KNF . Warszawa 2019
5. POŚREDNICY KREDYTOWI W ZAKRESIE KREDYTU KONSUMENCKIEGO ORAZ INSTYTUCJE POŻYCZKOWE – FORMALNE I PRAKTYCZNE ASPEKTY REJESTRACJI .Poradnik dla przedsiębiorcy i konsumenta (PDF), Brand. A., Przybysz M., Sakowska A., Świderska J., KNF . Warszawa 2020
6. Whistleblowing w bankach, Cichy Ł., KNF . Warszawa 2017
7. Bezpieczeństwo konsumentów : na rynkach usług finansowych i społecznych , Kasprzak R., Lubowiecki-Vikuk A. (red.), Oficyna Wydawnicza SGH. Warszawa 2020
8. Manipulacja i wykorzystanie przewagi informacyjnej na rynku finansowym , Więckowska M., Borowski K., Oficyna Wydawnicza SGH - Szkoła Główna Handlowa. Warszawa 2020
9. Misselling usług finansowych : problemy i uwarunkowania prawno-ekonomiczne , Nizioł K. , Wydawnictwo Naukowe Uniwersytetu Szczecińskiego. Szczecin 2019
10. Piramidy finansowe : teoria, regulacje, praktyka , Masiukiewicz M. , Wydawnictwo Naukowe PWN SA.. Warszawa 2015

17. Metody weryfikacji zakładanych efektów uczenia się

Egzamin, obecność na zajęciach, kolokwium zaliczeniowe

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

- Wykład — egzamin
- Ćwiczenia — praca końcowa semestralna, kolokwium opisowe

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.
ćwiczenia	30 godz.

Praca własna studenta

Przygotowanie do kolokwium	20 godz.
Przygotowanie do egzaminu	20 godz.
Czytanie zadanie zadanej literatury	30 godz.

Sumarycznie

Łączna liczba godzin	130 godz.
Liczba punktów ECTS	5

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Metody implementacji algorytmów** **Efficient Implementation of Algorithms**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DPMIA**
6.	Rodzaj przedmiotu/modułu **Inne**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **zimowy**
11.	Forma zajęć i liczba godzin ćwiczenia — 15 godzin, pracownia — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zalecana znajomość przedmiotów** • Wstęp do informatyki • Algorytmy i struktury danych **Niezbędne kompetencje** • Znajomość podstawowych algorytmów • Znajomość dowolnego wysokopoziomowego języka programowania (najlepiej C++)
13.	Cele kształcenia dla przedmiotu Celem przedmiotu są między innymi: • nauczenie studentów zarówno podstawowych, jak i bardziej zaawansowanych algorytmów, • przećwiczenie efektywnej implementacji algorytmów, • poznanie struktur danych przydatnych w pisaniu programów, • nauczenie prezentacji własnych rozwiązań, • zdobycie umiejętności analizowania problemów algorytmicznych, • zdobycie umiejętności wykorzystania znanych algorytmów przy rozwiązywaniu różnorodnych problemów, • przygotowanie do startu w zawodach algorytmicznych, tj. AMPPZ, CERC itd.

14.	Treści programowe Rozwiązywanie problemów z zakresu algorytmów: • zachłannych, • wymagających złożonej implementacji, • wyszukiwania binarnego, • dynamicznych, • struktury zbiorów rozłącznych, • struktury drzew przedziałowych, • kombinatorycznych, • teorii liczb, • gier, • grafowych, • skojarzeń, • maksymalnego przepływu, • tekstowych, • geometrycznych.

15. Zakładane efekty uczenia się

Wiedza

zna podstawowe metody projektowania, analizowania i implementowania algorytmów, a także podstawowe struktury danych i operacje na nich	K_W07
ma wiedzę na temat języków programowania	K_W08
zna i rozumie podstawowe techniki dowodzenia poprawność rozwiązań	K_W01
ma podstawową wiedzę na temat matematyki, prawdopodobieństwa, matematyki dyskretnej, teorii liczb, teorii grafów i kombinatoryki	K_W02

Umiejętności

potrafi pisać, uruchamiać i testować programy	K_U16
potrafi projektować, implementować oraz analizować algorytmy pod kątem poprawności i złożoności obliczeniowej	K_U06
potrafi ocenić przydatność typowych metod i narzędzi informatycznych oraz wybrać i zastosować odpowiednią metodę i narzędzia do typowych zadań informatycznych	K_U19
potrafi prezentować opracowane zagadnienia, formułować opinie, a także podejmować dyskusję i analizować problemy informatyczne	K_U21
potrafi posługiwać się podstawowymi pojęciami i twierdzeniami z zakresu matematyki, rachunku prawdopodobieństwa i statystyki oraz stosować je do rozwiązywania problemów	K_U13
potrafi stosować poznane pojęcia i metody matematyki dyskretnej do modelowania i analizy problemów informatycznych	K_U06

Kompetencje społeczne

wykazuje samodzielność myślenia i działania przy rozwiązywaniu problemów i wykonywaniu zadań typowych dla zawodów związanych z informatyką	K_K06
aktywnie prezentuje krytyczną postawę wobec stwierdzeń, uwag i wniosków, zwłaszcza niepopartych logicznym uzasadnieniem	K_K06

16. Literatura obowiązkowa i zalecana

- Cormen Thomas H., Leiserson Charles E., Rivest Ronald L, Clifford Stein, Wprowadzenie do algorytmów
- blog na stronie https://codeforces.com/
- https://cp-algorithms.com/
- https://solve.edu.pl/ sparingi/resources

17. Metody weryfikacji zakładanych efektów uczenia się

Rozwiązanie zadań poruszonych na zajęciach.

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia ćwiczenio-pracowni należy rozwiązać odpowiednią liczbę zadań na zajęciach, oraz wykazać odpowiednią aktywność na omówieniu zadań.
Dodatkowa aktywność oraz prezentacje rozwiązań mogą wpłynąć na obniżenie powyższego limitu.

19.	Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	30 godz.
ćwiczenia	15 godz.

Praca własna studenta

dobre zrozumienie oraz zaimplementowanie rozwiązań przedstawionych na zajęciach	60 godz.
przejrzenie materiałów przekazanych przez prowadzących przed zajęciami	5 godz.

Sumarycznie

Łączna liczba godzin	115 godz.
Liczba punktów ECTS	4

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Ochrona własności intelektualnej** **Intellectual Property Rights**
2.	Dyscyplina **nauki prawne**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-KOWI**
6.	Rodzaj przedmiotu/modułu **Nieinformatyczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **nieokreślony**
11.	Forma zajęć i liczba godzin ćwiczenia — 15 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest kształcenie umiejętności praktycznej analizy przepisów dotyczących ochrony własności intelektualnej oraz świadomości konieczności ich przestrzegania.
14.	Treści programowe 1. Źródła praw na dobrach niematerialnych. 2. Zakres przedmiotowy i podmiotowy prawa autorskiego. 3. Autorskie prawa osobiste i majątkowe. 4. Okres ochrony utworu. 5. Domena publiczna. 6. Dozwolony użytek prywatny i publiczny. 7. Prawo cytatu i plagiat. 8. Odpowiedzialność za naruszenie praw autorskich. 9. Umowy o przekazaniu praw autorskich i umowy licencyjne. 10. Wolne licencje, Crative Commons i Ruch Wolnej Kultury. 11. Organizacje zbiorowego zarządzania prawami autorskimi i pokrewnymi. 12. Prawa własności przemysłowej. 13. Bazy danych. 14. Zwalczanie nieuczciwej konkurencji.

15.	Zakładane efekty uczenia się

Wiedza

Ma podstawową wiedzę na temat aktów prawnych obowiązujących w zakresie ochrony własności intelektualnej	K_W16

Umiejętności

potrafi analizować i interpretować tekst ustawy o prawie autorskim i prawach pokrewnych	K_U22
potrafi zaprezentować przygotowany materiał dotyczący wybranych zagadnień z zakresu ochrony własności intelektualnej	K_U22

Kompetencje społeczne

ma świadomość konieczności przestrzegania prawa z zakresu ochrony własności intelektualnej	K_K08

16. Literatura obowiązkowa i zalecana

1. Ustawa z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz.U. z 2017r. poz. 880)
2. Krótki kurs własności intelektualnej. Materiały dla uczelni http://prawokultury.pl/kurs/
3. Marek Łazewski, Mariusz Gołębiowski: Vademecum Innowacyjnego Przedsiębiorcy, tom III – Własność intelektualna, Warszawa 2006, ISDN 83-907625-6-0.
4. Ustawa z dnia 30 czerwca 2000 r. Prawo własności przemysłowej (Dz.U. z 2017r. poz. 776, z późn. zm.)
5. Traktat Światowej Organizacji Własności Intelektualnej o Prawie Autorskim, sporządzony w Genewie dnia 20 grudnia 1996 r. (Dz.U. z 2005r. Nr 3, poz.12)
6. Richard M. Stallman: „Własność intelektualna” to zwodniczy miraż, 2004

17. Metody weryfikacji zakładanych efektów uczenia się

przygotowanie pracy pisemnej dotyczącej zagadnienia z zakresu ochrony własności intelektualnej

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Warunkiem zaliczenia przedmiotu jest:

- obecność na zajęciach - dopuszczalna jest jedna nieobecność, którą należy zaliczyć w formie uzgodnionej z prowadzącym,
- aktywny udział w grupowej analizie tekstu,
- przedstawienie opracowanego zagadnienia z zakresu ochrony własności intelektualnej.

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

ćwiczenia	15 godz.

Praca własna studenta

studiowanie tematyki wykładów i literatury	10 godz.
przygotowanie raportu/prezentacji	10 godz.

Sumarycznie

Łączna liczba godzin	35 godz.
Liczba punktów ECTS	1

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Od Mordoru po Matrix. Fantastyczna filozofia polityki** **From Mordor to Matrix. The Fantastic Political Philosophy**
2.	Dyscyplina **nauki o polityce i administracji**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-DHSFantaFil**
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu • Zainteresowanie fantastyką literacką i filmową jako filozoficzną i polityczną (ideologiczną) formą wypowiedzi, • Zainteresowanie myślą polityczną i filozofią polityki.
13.	Cele kształcenia dla przedmiotu Zapoznanie studentów z problematyką będącą przedmiotem zainteresowania filozofii oraz antropologii polityki (natura polityki, wspólnoty politycznej i człowieka jako „zwierzęcia politycznego”; uniwersalne mechanizmy i aporie władzy, koncepcje strukturalnej organizacji społeczeństwa; pochodzenie, zmienność, ewolucja– progres, regres, cykliczność - i postulowana wartość moralna ustrojów oraz instytucji politycznych; światopoglądy, ideologie, modele polityczne i ich zderzenia z praxis; legitymizacja; dyskurs polityczny; konflikt społeczny; rewolucja i wojna; mechanizmy fanatyzmu, ekstremizmu i radykalizmu, utopia jako fikcja spekulatywna; dystopia jako krytyka polityczna, sprzężenie zwrotne między nieuświadomioną i uświadomioną fikcją a praktyką polityczną) poprzez analizę utworów, gatunków, konwencji i toposów historycznej oraz współczesnej fantastyki.

14.	Treści programowe 1. Polityczne znaczenie mitów „starego” i „nowego” typu (refleksja historiozoficzna, premodernizm kontra nowoczesność; dialektyka Oświecenia, natura, kultura i technologia a człowieczeństwo); 2. Wpływ fantastyki (fikcji spekulatywnej) na politykę, religię oraz naukę i vice versa; 3. Utopia (futurotopia, retrotopia) i dystopia (ich treść, funkcje społeczne i polityczne, potencjał intelektualny oraz moralny); 4. Wizje bliskiej i odległej przyszłości ludzkości w kontekście polityczno-społecznym; 5. Rzeczywistość wirtualna i sztuczne inteligencje w kontekście politycznym. 6. Rewolucja biotechnologiczna, transhumanizm, Osobliwość („kryzys narzędziowni” i konieczność rewolucji paradygmatycznej). 7. Polityczne i antropologiczne konteksty, toposy, funkcje i znaczenia charakterystyczne dla form i gatunków (mit, utopia, fantasy, science fiction, cyberpunk, postapokaliptyka) oraz dzieł literatury i filmu (m.in. „Metropolis”, „The Shape of Things to Come”, „Odyseja kosmiczna 2001”, „Władca Pierścieni”, „Gwiezdne wojny”, „Żołnierze kosmosu”, „1984”, „Matrix”) i autorów (m.in. Morus, Wells, Orwell, Campbell, Lem, Dukaj, Bradbury, Robinson, Liu, Stephenson).
15.	Zakładane efekty uczenia się **Wiedza** Zna i rozumie w stopniu elementarne kategorie filozoficzne występujące w analizowanych tekstach kultury oraz ich rolę w kształtowaniu refleksji politologicznej Zna i rozumie w stopniu najważniejsze nurty myśli politycznej przejawiające się w analizowanych klasycznych i współczesnych dziełach fikcji spekulatywnej **Umiejętności** Potrafi porządkować, klasyfikować, porównywać oraz analizować i interpretować zjawiska polityczne analizowane w wybranych klasycznych dziełach fikcji spekulatywnej, uwzględniając wzajemne oddziaływania zjawisk politycznych i społecznych Potrafi weryfikować proponowane w analizowanych dziełach fikcji spekulatywnej rozwiązania problemów politycznych. Potrafi poddawać krytyce proponowane rozwiązania i argumentować sądy na temat możliwych rozwiązań problemu **Kompetencje społeczne** Jest gotów do krytycznej oceny swojej wiedzy w zakresie fikcji spekulatywnej oraz filozofii polityki, ciągłego dokształcania się i rozwoju zawodowego oraz korzystania z opinii ekspertów w przypadku trudności z samodzielnym rozwiązaniem problemów

16. Literatura obowiązkowa i zalecana

Obowiązkowa

- Frederick Jameson, Archeologie przyszłości. Pragnienie zwane utopią i inne fantazje naukowe, Wydawnictwo Uniwersytetu Jagiellońskiego, Kraków 2012.
- James Gunn, Droga do science fiction, Wydawnictwo Alfa, 4 (1985-1988).
- Tomasz Majkowski, W cieniu białego drzewa. Powieść fantasy w XX wieku, Wydawnictwo Uniwersytetu Jagiellońskiego 2013.

Zalecana

- The Cambridge Companion to Science Fiction (red. Edward James, Farah Mendlesohn), Cambridge University Press 2003.
- The Oxford Handbook of Science Fiction (red. Rob Latham), Oxford University Press 2014.
- Adam Roberts, The History of Science Fiction, Palgrave Macmillan 2016.
- Reading Science Fiction (red. James Gunn, Marleen S. Barr, Matthew Candelaria),
- Palgrave Macmillan 2009. Carl Freedman, Critical Theory and Science Fiction, Wesleyan University Press 2000.
- John Rieder, Colonialism and the Emergence of Science Fiction, Wesleyan University Press 2008.
- The Philosophy of Science Fiction Film (red. Steven M. Sanders), The University Press of Kentucky 2008.
- Patrick Curry, Defending Middle-Earth. Tolkien: Myth and Modernity, HarperCollins 2012.
- Abaigail E. Ruane, Patrick James, The International Relations of Middle-Earth. Learning from The Lord of the Rings, The University of Michigan Press 2012.
- Dariusz Brzostek, Fikcja antropologiczna czy antropologia spekulatywna? O funkcjach poznawczych narracji fantastycznonaukowych, „Prace kulturoznawcze” 23, nr 2-3, Wrocław 2019.

17. Metody weryfikacji zakładanych efektów uczenia się

egzamin pisemny

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

egzamin pisemny

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.

Praca własna studenta

przygotowanie do egzaminu lub rozwiązywanie dodatkowych zadań	20 godz.
lektura tekstów	20 godz.

Sumarycznie

Łączna liczba godzin	70 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Polityka gospodarcza** **Economic Policy Policy**
2.	Dyscyplina **nauki o polityce i administracji**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu
6.	Rodzaj przedmiotu/modułu **Humanistyczno-społeczny**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin wykład — 30 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu brak
13.	Cele kształcenia dla przedmiotu Celem przedmiotu jest zapoznanie studentów z podstawami i głównymi pojęciami polityki gospodarczej w otwartej gospodarce rynkowej, w celu umożliwienia im lepszego zrozumienia głównych zagadnień i problemów związanych z przebiegiem procesów gospodarczych.
14.	Treści programowe • Dziedzictwo gospodarki centralnie planowanej • Transformacja gospodarcza • Rozwój a wzrost gospodarczy • Polityka fiskalna • Polityka pieniężna • Rynek pracy • Problem starzenia się społeczeństw • Cykle koniunkturalne • Kryzysy gospodarcze • Inflacja • Ekonomiczne uwarunkowania procesów integracyjnych • Unia walutowa

15.	Zakładane efekty uczenia się

Wiedza

zna podstawowe i główne pojęcia polityki gospodarczej w otwartej gospodarce rynkowej zna główne zagadnienie i problemy związane z kierowaniem procesami gospodarczymi	

Umiejętności

potrafi analizować zjawiska i procesy zachodzące w gospodarce krajowej i międzynarodowej z wykorzystaniem standardowych metod i narzędzi stosowanych w ekonomii potrafi ocenić role przedsiębiorczości jako „czwartego czynnika" ekonomicznego potrafi samodzielnie interpretować zjawiska gospodarcze w powiązaniu z poznanymi koncepcjami teoretycznymi potrafi ocenić wpływ uwarunkowań i ograniczeń natury politycznej, prawnej oraz społecznej na kształt realizowanej polityki ekonomicznej	

16. Literatura obowiązkowa i zalecana

- B. Winiarski, Polityka gospodarcza, Wydawnictwo Naukowe PWN, Warszawa 2009
- T. Włudyka, Marcin Smaga, Instytucje gospodarki rynkowej, Wolters Kluwer, Warszawa 2018

17. Metody weryfikacji zakładanych efektów uczenia się

Test obejmujący cały zakres wykładu

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Test jednokrotnego wyboru

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

wykład	30 godz.

Praca własna studenta

Samodzielna praca studenta	50 godz.

Sumarycznie

Łączna liczba godzin	80 godz.
Liczba punktów ECTS	3

1.	Nazwa przedmiotu/modułu w języku polskim oraz angielskim **Praktyka programowania sportowego** **Sport programming in practice**
2.	Dyscyplina **informatyka**
3.	Język wykładowy **polski**
4.	Jednostka prowadząca przedmiot **Instytut Informatyki**
5.	Kod przedmiotu/modułu **28-INF-S-PrProgSport**
6.	Rodzaj przedmiotu/modułu **Inne**
7.	Kierunek studiów (specjalność/specjalizacja) **ISIM**
8.	Poziom studiów **studia I stopnia**
9.	Rok studiów (jeśli obowiązuje) —
10.	Semestr **letni**
11.	Forma zajęć i liczba godzin pracownia — 75 godzin
12.	Wymagania wstępne w zakresie wiedzy, umiejętności i kompetencji społecznych dla przedmiotu/modułu **Zrealizowane przedmioty** • Metody implementacji algorytmów
13.	Cele kształcenia dla przedmiotu Podstawowym celem zajęć jest ułatwienie uczestnikom przygotowań do startów w zespołowych zawodach programistycznych (Akademickie Mistrzostwa Polski w Programowaniu Zespołowym, Central European Programming Contest, ICPC World Finals). Prowadzący (z pomocą uczestników) wybiera w każdym tygodniu pięciogodzinny konkurs o odpowiednim poziomie trudności. Następnie uczestnicy rozwiązują zadania z wybranego konkursu w tak zwanym trybie wirtualnym, czyli widząc jak ich aktualne wyniki mają się do wyników prawdziwych zawodów).
14.	Treści programowe Podczas zajęć uczestnicy rozwiązują zadania z zespołowych konkursów typu ICPC, w szczególności z wybranych regionalnych eliminacji do tych zawodów.

15.	Zakładane efekty uczenia się

Wiedza

zna podstawowe metody projektowania i analizy algorytmów	K_W07

Umiejętności

potrafi formułować precyzyjną specyfikację problemu	K_U01
potrafi sprawnie zaimplementować rozwiązanie zadania w odpowiednio wybranym języku programowania	K_U16
potrafi sprawnie projektować, implementować oraz analizować algorytmy	K_U06
potrafi efektywnie pracować w zespole	K_U23

Kompetencje społeczne

pracując z zespole dzieli zadania biorąc pod uwagę słabe i mocne strony każdego z zawodników w drużynie	K_K05

16. Literatura obowiązkowa i zalecana

Brak.

17. Metody weryfikacji zakładanych efektów uczenia się

napisanie rozwiązań zadań z wybranych konkursów

18. Warunki i forma zaliczenia poszczególnych komponentów przedmiotu/modułu

Do zaliczenia przedmiotu potrzebny jest aktywny udział w zajęciach. Przez aktywny udział rozumie się rozwiązania przynajmniej jednego zadania z konkursu wybranego na dany tydzień (podczas zajęć lub w terminie ustalonym i podanym na początku semestru razem z innymi zasadami).

19. Nakład pracy studenta

Zajęcia z udziałem nauczyciela

pracownia	75 godz.

Sumarycznie

Łączna liczba godzin	75 godz.
Liczba punktów ECTS	2