UNIWERSYTET W SIEDLCACH

INSTYTUT INFORMATYKI

Kierunek INFORMATYKA

INFORMATOR

SYLABUS

Obowiązuje w roku akademickim 2025/26

studia I stopnia

(inżynierskie, profil praktyczny)

czas trwania: 7 semestrów

Siedlce 2025

Semestr 3

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Język angielski II**	
Nazwa w języku angielskim:	English II	
Język wykładowy:	angielski (wspomagany językiem polskim)	
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka	
Jednostka realizująca:	Centrum Języków Obcych	
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy	
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia	
Rok studiów:	Drugi	
Semestr:	Trzeci	
Liczba punktów ECTS:	**4**	
Imię i nazwisko koordynatora przedmiotu:	dr inż. Maria Markowska	
Imię i nazwisko prowadzących zajęcia:	mgr Joanna Madej-Borychowska	
Założenia i cele przedmiotu:	Zakłada się, że student posiada wiedzę i umiejętności posługiwania się językiem angielskim na poziomie wynikającym z ukończenia modułu „Język angielski I". Celem przedmiotu jest uzyskanie wiedzy i umiejętności posługiwania się językiem angielskim na poziomie B2 ESOKJ.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	słownictwo i struktury gramatyczne niezbędne do skutecznej komunikacji językowej na poziomie B2 z zakresu tematyki wynikającej z treści modułu kształcenia..	
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	zrozumieć znaczenie głównych wątków przekazu zawartego w złożonych tekstach na tematy konkretne i abstrakcyjne, łącznie ze zrozumieniem dyskusji na tematy z zakresu swojej specjalności.	K_U03, K_U04
U_02	formułować przejrzyste wypowiedzi ustne i pisemne dotyczące tematów ogólnych i specjalistycznych.	K_U03, K_U04
U_03	zdobywać informacje oraz udzielać ich.	K_U03, K_U04
U_04	brać udział w dyskusji, argumentować, wyrażać aprobatę i sprzeciw, negocjować.	K_U03, K_U04

U_05	kontrolować swoje wypowiedzi pod względem poprawności gramatycznej i leksykalnej.	K_U03
U_06	pracować samodzielnie z tekstem specjalistycznym.	K_U03
U_07	współpracować i pracować w grupie, przyjmując w niej różne role.	K_U05
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	krytycznej oceny posiadanej wiedzy oraz ma świadomość potrzeby znajomości języka obcego w życiu prywatnym i przyszłej pracy zawodowej.	K_K01

Forma i typy zajęć:	Studia stacjonarne: ćwiczenia audytoryjne – 60 godz. Studia niestacjonarne: ćwiczenia audytoryjne – 32 godz.

Wymagania wstępne i dodatkowe:

Umiejętność posługiwania się językiem angielskim na poziomie „Język angielski I".

Treści modułu kształcenia:

1. Internet: bezpieczeństwo, strony internetowe.
2. Roboty.
3. Grafika i multimedia.
4. Języki programowania.
5. Zawody w informatyce.
6. Sieci komputerowe.
7. Rzeczywistość wirtualna.
8. Bezpieczeństwo pracy.

Literatura podstawowa:

Infotech – English for Computer Users, S. R. Esteras, wyd. Cambridge University Press, Cambridge 2008;
English 4 IT: praktyczny kurs języka angielskiego dla specjalistów IT i nie tylko, B. Błaszczyk, Gliwice, cop. 2017.

Literatura dodatkowa:

1. Wielki słownik angielsko-polski / polsko-angielski, red. nauk. B. Lewandowska-Tomaszczyk, 2014, PWN-OUP;
2. Słownik informatyki stosowanej, angielsko-polski, polsko-angielski, M. Trojański, 2007, Warszawa, wyd. C.H. Beck 2007.

Planowane formy/działania/metody dydaktyczne:

Podejście eklektyczne, umożliwiające indywidualizację nauczania, czyli dostosowanie technik, form pracy, typów zadań i treści do danej grupy studentów. Stosowane formy pracy to, między innymi: praca w parach (np.: odgrywanie ról, wymiana informacji), praca w grupach (projekty, konkursy, rozwiązywanie problemów, zebranie słownictwa itp.), praca indywidualna studentów, czy też nauczanie tradycyjne − frontalne (prezentacja materiału leksykalnego, zasad gramatycznych, treści ilustracji itp.). Ćwiczenia wspomagane są technikami multimedialnymi.

Sposoby weryfikacji efektów uczenia się osiągniętych przez studenta:

Symbol efektu	**Metody weryfikacji efektów uczenia się**

W_01	Kolokwium pisemne składające się z kilku z podanych form: - krótkie ustrukturyzowane pytania, np.: *What is …?;* pytania jednokrotnego wyboru, np.: *Read the questions and select one answer: a, b, c, or d.*; lub wielokrotnego wyboru, np.: *Read the questions and select at least one answer: a, b, c, or d.*; - testy wyboru Tak/Nie, np.: *Decide whether the following statements are true or false. Circle YES or NO.* lub dopasowywania odpowiedzi, np.: *Match items 1–5 with options A–E to form correct collocations.*; - ćwiczeń sprawdzających umiejętności mówienia w formie zapisanej wypowiedzi ustnej monologowej i dialogowej, np.: *Complete the dialogue by filling in each gap with one word*.; oraz zastosowania wiedzy teoretycznej i praktycznej (leksykalnej, gramatycznej i pragmatycznej) w przetwarzaniu i tworzeniu wypowiedzi samodzielnych i w interakcji z rozmówcą, np.: *Write what you would say in each of the following situations.*
U_01-U_06	Kolokwium pisemne składające się z kilku z podanych form: - krótkie ustrukturyzowane pytania, np.: *What is …?;* pytania jednokrotnego wyboru, np.: *Select one answer: a, b, c, or d.*; lub wielokrotnego wyboru, np.: *Select at least one answer: a, b, c, or d.*; - testy wyboru Tak/Nie, np.: *Decide whether the following statements are true or false. Circle YES or NO.* lub dopasowywania odpowiedzi, np.: *Match items 1–5 with options A–E to form correct collocations.*; - ćwiczeń sprawdzających umiejętności mówienia w formie zapisanej wypowiedzi ustnej monologowej i dialogowej, np.: *Complete the dialogue by filling in each gap with one word*.; oraz zastosowania wiedzy teoretycznej i praktycznej (leksykalnej, gramatycznej i pragmatycznej) w przetwarzaniu i tworzeniu wypowiedzi samodzielnych i w interakcji z rozmówcą, np.: *Write what you would say in each of the following situations.*
U_07	Obserwacja studenta w trakcie wykonywanych ćwiczeń.
K_01	Obserwacja studenta w trakcie wykonywanych ćwiczeń.

Forma i warunki zaliczenia:

Zaliczenie semestru na ocenę na podstawie co najmniej dwóch kolokwiów sprawdzających stopień opanowania wiedzy i umiejętności - 90% oceny końcowej oraz obserwacja pracy studenta podczas ćwiczeń rozwijających umiejętności komunikacyjne - 10% oceny końcowej.

Kryteria oceniania:

- 0 – 50 pkt.: niedostateczna (F),
- 51 – 60 pkt.: dostateczna (E),
- 61 – 70 pkt.: dostateczna plus (D),
- 71 – 80 pkt.: dobra (C),
- 81 – 90 pkt.: dobra plus (B),
- 91 – 100 pkt.: bardzo dobra (A).

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w ćwiczeniach audytoryjnych	60 godz.
Samodzielne przygotowanie się do ćwiczeń audytoryjnych	30 godz.

Udział w konsultacjach	3 godz.
Samodzielne przygotowanie się do kolokwiów	7 godz.
Sumaryczne obciążenie pracą studenta	100 godz.
Punkty ECTS za przedmiot	4
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w ćwiczeniach audytoryjnych	32 godz.
Samodzielne przygotowanie się do ćwiczeń audytoryjnych	45 godz.
Udział w konsultacjach	3 godz.
Samodzielne przygotowanie się do kolokwiów	20 godz.
Sumaryczne obciążenie pracą studenta	100 godz.
Punkty ECTS za przedmiot	4

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Problemy społeczne i zawodowe informatyki**	
Nazwa w języku angielskim:	Social and Vocational Issues in Computer Science	
Język wykładowy:	polski	
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka	
Jednostka realizująca:	Instytut Informatyki	
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy	
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia	
Rok studiów:	Drugi	
Semestr:	Trzeci	
Liczba punktów ECTS:	**1**	
Imię i nazwisko koordynatora przedmiotu:	dr Jarosław Skaruz	
Imię i nazwisko prowadzących zajęcia:	dr Jarosław Skaruz	
Założenia i cele przedmiotu:	Założono, że studenci znają podstawy programowania w dowolnym języku programowania oraz uzyskali pierwsze doświadczenia podczas odbytej praktyki. Celem przedmiotu jest zaznajomienie studentów z podstawowymi problemami społecznymi i zawodowymi związanymi z wykonywaniem zawodu informatyka.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	wybrane zagadnienia z zakresu prawa związanego z wykonywaniem zawodu informatyka.	**K_W03**
W_02	zagadnienia z zakresu ergonomii stanowiska pracy.	**K_W03**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	do odpowiedzialnego pełnienia roli zawodowej informatyka, w tym do przestrzegania prawa szczególnie związanego z rola informatyka	**K_K04**

Forma i typy zajęć:	Wykłady (15 godz) – studia stacjonarne Wykłady (9 godz) – studia niestacjonarne

Wymagania wstępne i dodatkowe:

- Znajomość podstaw programowania.
- Podstawowe kompetencje zawodowe i społeczne uzyskane podczas praktyki.

Treści modułu kształcenia:

Wykład

1. Rozwój informatyki. Społeczeństwo informacyjne (1h)
2. Przygotowanie stanowiska pracy, ergonomia pracy przy komputerze (1h)
3. Ochrona danych osobowych (2h)
4. Ryzyka przedsięwzięć informatycznych (2h)
5. Rynek pracy (1h)
6. Metody zarządzania czasem (1h)
7. Prawa autorskie (2h)
8. Umowy outsourcingowe B2B (2h)
9. Umowy o wykonanie oprogramowania (2h)
10. Kolokwium (1h).

Literatura podstawowa:

1. Castells M., Społeczeństwo sieci, WN PWN, Warszawa 2007.
2. Cieciura M.: Wybrane problemy społeczne i zawodowe informatyki, VIZJA, wyd. III, Warszawa 2012

Literatura dodatkowa:

1. Cieciura M.: Podstawy technologii informacyjnych z przykładami zastosowań. Opolgraf S.A. Warszawa 2006

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	będzie sprawdzany na kolokwium na ostatnim wykładzie. Przykładowe pytania na kolokwium: 1. Opisz najważniejsze zasady związane z ochron danych osobowych. 2. Opisz najważniejsze zapisy umów outsourcingowych. 3. Opisz najważniejsze zapisy w umowach o wytworzenie oprogramowania. 4. Opisz najważniejsze zasady związane z majątkowymi prawami autorskimi.
W_02	będzie sprawdzany na kolokwium na ostatnim wykładzie. Przykładowe pytania na kolokwium: 1. Opisz idealne środowisko pracy informatyka.
K_01	Efekt sprawdzany jest podczas kolokwium.

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem na ocenę. Z kolokwium student może otrzymać maksymalnie 100 punktów. Moduł zostanie zaliczony pod warunkiem uzyskania co najmniej połowy punktów.

Ocena końcowa z przedmiotu, w zależności od sumy uzyskanych punktów (maksymalnie 100 pkt) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 50 pkt: niedostateczna (F),
- 51 – 60 pkt: dostateczna (E),
- 61 – 70 pkt: dostateczna plus (D),
- 71 – 80 pkt: dobra (C),
- 91 – 100 pkt: bardzo dobra (A).

Bilans punktów ECTS:	
Studia stacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godzin
Udział w konsultacjach godz. z przedmiotu	1 godzina
Przygotowanie się do kolokwium	9 godzin
Sumaryczne obciążenie pracą studenta	25 godzin
Punkty ECTS za przedmiot	1
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	9 godzin
Udział w konsultacjach godz. z przedmiotu	1 godziny
Przygotowanie się do kolokwium	15 godzin
Sumaryczne obciążenie pracą studenta	25 godzin
Punkty ECTS za przedmiot	1

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Laboratorium matematyki**	
Nazwa w języku angielskim:	Matematics laboratory	
Język wykładowy:	polski	
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka	
Jednostka realizująca:	Instytut Matematyki	
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy	
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia):	pierwszego stopnia	
Rok studiów:	drugi	
Semestr:	trzeci	
Liczba punktów ECTS:	2	
Imię i nazwisko koordynatora przedmiotu:	dr Dorota Kozak-Superson	
Imię i nazwisko prowadzących zajęcia:	dr Agnieszka Prusińska, dr Agnieszka Siłuszyk, dr Dorota Kozak-Superson	
Założenia i cele przedmiotu:	Zakłada się, że student posiada wiedzę z matematyki w zakresie szkoły średniej i pierwszego roku studiów na kierunku informatyka. Celem laboratorium jest zapoznanie studentów z możliwościami obliczeniowymi programów: Excel w obszarze algebry, Wolfram Alpha w zakresie analizy matematycznej oraz Statistica w obszarze statystyki.	

Symbol efektu	**Efekty uczenia się:** **UMIEJĘTNOŚCI** **Student potrafi:**	**Symbol efektu kierunkowego**
U_01	wykorzystywać programy komputerowe (arkusz kalkulacyjny MS Excel, Wolfram Alpha, Statistica) do realizacji zadań z różnych działów matematyki	**K_U02**
U_02	rozpoznać problem i o ile to możliwe ułożyć algorytm jego rozwiązania, korzystając z możliwości różnych programów komputerowych	**K_U01**

Forma i typy zajęć:	studia stacjonarne: laboratorium (30 godz.) studia niestacjonarne: laboratorium (18 godz.)

Wymagania wstępne i dodatkowe:
Znajomość matematyki w zakresie szkoły średniej i pierwszego roku studiów na kierunku informatyka.

Treści modułu kształcenia:

1. Wprowadzenie do Wolfram Alpha. Obliczenia z programem Wolfram Alpha.
2. Wyrażenia algebraiczne. Wielomiany. Rozwiązywanie równań i układów równań.
3. Dziedzina funkcji. Wykresy funkcji. Granica. Asymptoty.
4. Rachunek różniczkowy.
5. Rachunek całkowy.
6. Wykorzystanie arkusza kalkulacyjnego do zadań optymalizacyjnych. Programowanie liniowe - metoda simplex.
7. Algebra liniowa z komputerem. Macierze i wyznaczniki. Działania na macierzach. Wyznacznik macierzy. Macierz odwrotna. Układy równań.
8. Statystyka z programem Statistica. Wprowadzenie do programu. Statystyka opisowa. Metody wizualizacji danych oraz sposób interpretacji otrzymanych wyników.

Literatura podstawowa:

1. M. Sobczyk, Statystyka, Wydawnictwo Naukowe PWN, Warszawa 2004
2. M. Hadaś-Dyduch, A. Wolny-Dominiak, Matematyka z komputerem w przykładach, Wydawnictwo Uniwersytetu Ekonomicznego w Katowicach, Katowice 2023.
3. Red. D.Kopańska-Bródka, Algebra liniowa z komputerem, Wydawnictwo Uniwersytetu Ekonomicznego w Katowicach, Katowice 2011.

Literatura dodatkowa:

1. M. Gewert, Z. Skoczylas, Analiza matematyczna 1 i 2, GiS, Wrocław 2011
2. S. Przybyło, A. Szlachtowski, Algebra i geometria afiniczna w zadaniach, Warszawa 1994

Planowane formy/działania/metody dydaktyczne:

Ćwiczenia laboratoryjne z wykorzystaniem programów Wolfram Alpha, Statistica oraz Excel.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
U_01	Efekt realizowany podczas ćwiczeń oraz kolokwiów. Przykładowe zadanie: Dla podanych funkcji: wyznaczyć asymptoty; wyznaczyć ekstrema lokalne; znaleźć punkty przegięcia.
U_02	Efekt realizowany podczas ćwiczeń. Przykładowe zadanie: Suma długości trzech krawędzi prostopadłościanu wychodzących z jednego wierzchołka jest równa 54 cm. Długość jednej z tych krawędzi jest dwa razy większa od drugiej. Jakie są długości krawędzi tego prostopadłościanu, który ma największą objętość?

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem na ocenę na podstawie uczestnictwa na zajęciach (nie więcej niż dwie nieusprawiedliwione nieobecności) oraz wyników dwóch kolokwiów (po 25 pkt.).
Ocena z przedmiotu będzie wyliczana według zasady:

- 0 – 25 pkt: niedostateczna (F),
- 26 – 30 pkt: dostateczna (E),
- 31 – 35 pkt: dostateczna plus (D),
- 36 – 40 pkt: dobra (C),
- 41– 45 pkt: dobra plus (B),

- 46 – 50 pkt: bardzo dobra (A).

W przypadku nieuzyskania potrzebnej do zaliczenia liczby punktów studentowi przysługuje prawo do kolokwium poprawkowego.

Bilans punktów ECTS:

Studia stacjonarne	
Aktywność	Obciążenie studenta
Udział w ćwiczeniach laboratoryjnych	30 godz.
Udział w konsultacjach z przedmiotu	2 godz.
Samodzielne przygotowanie się do ćwiczeń	8 godz.
Przygotowanie się do kolokwiów	10 godz.
Sumaryczne obciążenie pracą studenta	50 godz.
Punkty ECTS za przedmiot	2 ECTS

Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w ćwiczeniach laboratoryjnych	18 godz.
Udział w konsultacjach z przedmiotu	2 godz.
Samodzielne przygotowanie się do ćwiczeń	20 godz.
Przygotowanie się do kolokwium	10 godz.
Sumaryczne obciążenie pracą studenta	50 godz.
Punkty ECTS za przedmiot	2 ECTS

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Algorytmy i złożoność**	
Nazwa w języku angielskim:	Algorithms and Complexity	
Język wykładowy:	polski	
Kierunek studiów, dla którego przedmiot jest oferowany:	informatyka	
Jednostka realizująca:	**Instytut Informatyki**	
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy	
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia	
Rok studiów:	drugi	
Semestr:	trzeci	
Liczba punktów ECTS:	4	
Imię i nazwisko koordynatora przedmiotu:	dr Artur Niewiadomski	
Imię i nazwisko prowadzących zajęcia:	dr Artur Niewiadomski	
Założenia i cele przedmiotu:	Zakłada się, że student zna podstawy programowania w języku C++ i Java. Celem zajęć jest przekazanie studentom wiedzy na temat struktur danych oraz różnych aspektów korzystania z algorytmów, w tym ich projektowania i złożoności obliczeniowej oraz zdobycie praktycznych umiejętności ich wykorzystania w implementowanych systemach.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	wybrane klasyczne algorytmy, ich własności oraz rodzaje, notacje wykorzystywane do szacowania złożoności oraz najważniejsze klasy złożoności obliczeniowej, zagadnienia związane z problemami NP-zupełnymi	**K_W06**
W_02	zagadnienia dotyczące dynamicznych struktur danych, reprezentacji grafów, algorytmów przeszukiwania grafów, operacji słownikowych na drzewach i ich złożoności obliczeniowej	**K_W06**
W_03	zagadnienia związane z wybranymi algorytmami sortowania i algorytmami operującymi na tekście oraz ich złożoności obliczeniowej	**K_W06**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	implementować i analizować klasyczne algorytmy związane z przetwarzaniem list, grafów i drzew, a także wybrane algorytmy sortowania i algorytmy tekstowe	**K_U22**
U_02	implementować i analizować algorytmy uwzględniające programowanie strukturalne i obiektowe	**K_U22**
U_03	dobierać i wykorzystywać klasyczne algorytmy oraz struktury danych adekwatnie do postawionego zadania	**K_U22**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	krytycznej oceny posiadanej wiedzy oraz jest gotów do uznawania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych z zakresu informatyki	**K_K01**

Forma i typy zajęć:	Studia stacjonarne: wykłady (30 godz.), ćwiczenia laboratoryjne (30 godz.) Studia niestacjonarne: wykłady (15 godz.), ćwiczenia laboratoryjne (15 godz.)

Wymagania wstępne i dodatkowe:

Warunkiem uczestnictwa w zajęciach jest wcześniejsze uzyskanie zaliczenia z następujących przedmiotów:

- Matematyka I, Matematyka II, Matematyka III
- Podstawy programowania, Programowanie obiektowe

lub znajomość literatury obowiązującej w tych przedmiotach. Student musi mieć opanowane podstawy kombinatoryki. Ponadto wymagana jest znajomość podstaw strukturalnego lub obiektowego języka programowania, a w tym umiejętność definiowania typów, posługiwanie się instrukcjami iteracyjnymi, podprogramami i rekurencją.

Treści modułu kształcenia:

Wykład:

1. **Wprowadzenie do algorytmów**: definicja, cechy algorytmu, sposoby zapisu, metody konstruowania algorytmów, przykład: algorytm NWD.
2. **Złożoność obliczeniowa algorytmów:** klasy złożoności czasowej, rzędy wielkości, porównanie i ocena złożoności algorytmów, złożoność pamięciowa.
3. **Algorytmy obliczeniowe:** algorytmy kombinatoryczne (obliczanie permutacji, silni, wariacji, kombinacji), algorytmy operacji na macierzach, algorytmy przekształcania liczb, algorytm znajdowania najmniejszego lub największego elementu.
4. **Dynamiczne struktury danych:** listy, stosy, kolejki - definicje, implementacje, operacje i ich złożoność, zastosowania.
5. **Rodzaje i reprezentacje grafów:** grafy nieskierowane i skierowane, grafy ważone, reprezentacje macierzowe i listowe.
6. **Algorytmy grafowe I:** DFS, BFS – złożoność, zastosowania, detekcja cykli, obliczanie spójnych składowych.
7. **Algorytmy grafowe II:** algorytmy znajdowania najkrótszej ścieżki w grafie ważonym (Dijkstra, Bellman-Ford), obliczanie silnie spójnych składowych (Kosaraju, Tarjan), złożoność.
8. **Drzewa I:** BST, AVL, RST, budowanie, balansowanie, zastosowania, operacje na drzewach (insert, delete, search) i ich złożoność.
9. **Drzewa II:** TRIE, PATRICIA, 2-3 drzewa, drzewa turniejowe, zastosowania, operacje na drzewach i ich złożoność.
10. **Problemy NP-trudne i trudniejsze**: klasy P, NP, NP-zupełność, lista problemów NP-trudnych, dowodzenie NP-trudności, problemy nierozstrzygalne, heurystyki dla problemów NP-zupełnych.
11. **Algorytmy sortowania I:** sortowanie proste – przez wybieranie, wstawianie, bąbelkowe, Shell, porównanie efektywności i złożoności.
12. **Algorytmy sortowania II:** sortowanie szybkie (quicksort), kopcowe, pozycyjne, porównanie efektywności i złożoności.
13. **Sortowanie zewnętrzne (plikowe):** sortowanie na plikach, łączenie naturalne i proste, sortowanie wielofazowe.
14. **Algorytmy wyszukiwania wzorca w tekście:** dopasowywanie wzorca i wyszukiwanie 'naiwne', drzewo sufiksowe i graf podsłów.
15. **Algorytmy ze strukturą drzewiastą (z nawrotami) i rekurencyjne**: algorytm ustawiania 8 hetmanów, algorytm znalezienia drogi skoczka szachowego, wieże Hanoi, omówienie zakresu i zasad egzaminu.

Laboratorium:

1. Implementacja prostych algorytmów (m.in. NWD, znajdowanie liczb pierwszych), typy danych i konstrukcje w języku programowania.
2. Tablice i pliki. Implementacja zbioru i operacji na zbiorach.

3. Algorytmy obliczeniowe.
4. Stosy. Proste algorytmy wykorzystujące stosy.
5. Listy jednokierunkowe i dwukierunkowe.
6. Reprezentacje grafów.
7. Algorytmy grafowe I.
8. Algorytmy grafowe II.
9. Drzewa I.
10. Drzewa II.
11. Algorytmy sortowania I.
12. Algorytmy sortowania II.
13. Algorytmy sortowania III.
14. Algorytmy tekstowe.
15. Rekursja z nawrotami.

Literatura podstawowa:

1. Banachowski L., Diks K., Rytter W., Algorytmy i struktury danych, Helion, Warszawa, 2021
2. Aho A.V., Hopcroft J.E., Ullman J.D., Projektowanie i analiza algorytmów komputerowych, PWN, Warszawa 2003
3. Wirth N., Algorytmy + struktury danych = programy, WNT Warszawa, 1999

Literatura dodatkowa:

1. Niewiadomski A., Świtalski P., Sidoruk T., Penczek W., Applying Modern SAT-solvers to Solving Hard Problems. Fundam. Inform. 165(3-4): 321–344 (2019), https://doi.org/10.3233/FI-2019-1788
2. Papadimitriou Ch. H., Złożoność obliczeniowa, Helion 2012.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi. Zajęcia laboratoryjne – zajęcia praktyczne z wykorzystaniem języka Java lub C# i wybranego środowiska programistycznego (np. InteliJ, Visual Studio).

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Egzamin pisemny, przykładowe zadanie: Określ klasy złożoności podanych algorytmów.
W_02	Egzamin pisemny, przykładowe zadanie: Dla danego grafu podaj kolejność odwiedzanych wierzchołków przez algorytm DFS i BFS.
W_03	Egzamin pisemny, przykładowe zadanie: Porównaj algorytm sortowania przez wstawianie i sortowanie kopcowe.
U_01	Sprawdzane sukcesywnie i oceniane po każdych laboratoriach. Przykładowe zadanie laboratoryjne: Zaimplementuj algorytm BFS operujący na listowej reprezentacji grafu.
U_02	Sprawdzane sukcesywnie i oceniane po każdych laboratoriach. Przykładowe zadanie laboratoryjne: Korzystając z podanej hierarchii klas zaimplementuj algorytm sortowania przez wstawianie i sortowanie szybkie.
U_03	Sprawdzane sukcesywnie i oceniane po każdych laboratoriach. Przykładowe zadanie laboratoryjne: Zaimplementuj znajdowanie najkrótszej ścieżki w grafie ważonym.
K_01	Obserwacja postawy studenta, jego umiejętności analizy i syntezy poszczególnych informacji podczas zajęć i konsultacji.

Forma i warunki zaliczenia:

Moduł kończy się egzaminem. Do egzaminu mogą przystąpić osoby, które uzyskały zaliczenie laboratorium. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim. Na tej formie zajęć student może maksymalnie uzyskać 50 pkt. Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania powyżej 25 punktów.
Poprawa laboratorium: najpóźniej w ciągu 2 tygodni od danych zajęć. Poprawa nie jest możliwa po zakończeniu semestru.
Egzamin jest egzaminem pisemnym. Można na nim uzyskać do 50 pkt. Egzamin będzie zaliczony w przypadku uzyskania powyżej 25 pkt.

Ocena końcowa z przedmiotu, w zależności od sumy uzyskanych punktów (maksymalnie 100 pkt) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 50 pkt: niedostateczna (F),
- 51 – 60 pkt: dostateczna (E),
- 61 – 70 pkt: dostateczna plus (D),
- 71 – 80 pkt: dobra (C),
- 81 – 90 pkt: dobra plus (B),
- 91 – 100 pkt: bardzo dobra (A).

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	30 godzin
Udział w ćwiczeniach laboratoryjnych	30 godzin
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	20 godzin
Udział w konsultacjach godz. z przedmiotu	2 godziny
Obecność na egzaminie	2 godziny
Przygotowanie się do egzaminu	16 godzin
Sumaryczne obciążenie pracą studenta	100 godz.
Punkty ECTS za przedmiot	**4**

Studia niestacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	15 godzin
Udział w ćwiczeniach laboratoryjnych	15 godzin
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	30 godzin
Udział w konsultacjach godz. z przedmiotu	2 godziny
Obecność na egzaminie	2 godziny
Przygotowanie się do egzaminu	36 godzin
Sumaryczne obciążenie pracą studenta	100 godz.

Punkty ECTS za przedmiot	**4**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Sieci komputerowe**	
Nazwa w języku angielskim:	Computer Networks	
Język wykładowy:	polski	
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka	
Jednostka realizująca:	Instytut Informatyki	
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy	
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia	
Rok studiów:	drugi	
Semestr:	trzeci	
Liczba punktów ECTS:	**4**	
Imię i nazwisko koordynatora przedmiotu:	dr hab. Stanisław Ambroszkiewicz Prof. uczelni	
Imię i nazwisko prowadzących zajęcia:	dr hab. Stanisław Ambroszkiewicz Prof. uczelni dr Grzegorz Terlikowski	
Założenia i cele przedmiotu:	Studenci powinni poznać zarówno klasyczna architekturę Internetu, jak i obecne trendy związane z ewolucją globalnych sieci komputerowych związanych z wirtualizacją, SDN i Chmurami obliczeniowymi. Celem modułu jest zapoznanie ze klasycznymi (Internet) oraz współczesnymi protokołami i technologiami używanymi w sieciach komputerowych z uwzględnieniem SDN i NFV, oraz nabycie umiejętności pozwalającym na projektowanie i zarządzanie sieciami LAN.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	podstawowe pojęcia z dziedziny sieci komputerowych.	**K_W07**
W_02	zasady działania warstw sieci komputerowych w Internetowym modelu warstwowym	**K_W07**
W_03	protokoły komunikacyjne wykorzystywane w sieciach komputerowych,	**K_W07**
W_04	adresowanie, protokoły i standardy wykorzystywane powszechnie w Internecie	**K_W07, K_W09**
W_05	podstawy programowania sieciowego w oparciu o język Java,	**K_W07**
W_06	podstawy projektowania i zarządzania sieciami LAN.	**K_W07**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	implementować proste aplikacje sieciowe na gniazdach TCP	**K_U22**
U_02	zaprojektować, zrealizować i skonfigurować prostą sieć (routery, serwery, hosty), także z wykorzystaniem DHCP	**K_U21**
U_03	diagnozować i usuwać usterki w sieciach komputerowych	**K_U11, K_U21**
U_04	posługiwać się symulatorem sieci.	**K_U17, K_U07**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	

K_01	krytycznej oceny posiadanej wiedzy oraz jest gotów do uznawania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych z zakresu informatyki	**K_K01**

Forma i typy zajęć:	studia stacjonarne: wykłady (30 godz.), ćwiczenia laboratoryjne (30 godz.), studia niestacjonarne: wykłady (18 godz.), ćwiczenia laboratoryjne (21 godz).

Wymagania wstępne i dodatkowe:

1. Umiejętność programowania w języku obiektowym (Java).
2. Znajomość architektury systemu komputerowego.

Treści modułu kształcenia:

1. **Podstawowe pojęcia i definicje związane z technologiami sieciowymi.** Pojęcie protokołu, warstwy protokołów w sieciach komputerowych. Model warstwowy OSI i Internetowy TCP-UDP/IP - porównanie.
2. **Rys historyczny Internetu**, rola aplikacji takich jak email a zwłaszcza WWW.
3. **Warstwa aplikacji – wprowadzenie.** Model programowanie sieciowego klient-serve, gniazda (sockety) TCP, oraz UDP. P2P – Gnutella oraz Torrnent. Aplikacje sieciowe w chmurze obliczeniowej oraz architektura serverless.
4. **Przegląd podstawowych aplikacji i protokołów sieciowych:** ftp, email, WWW, oraz DNS.
5. **Warstwa aplikacji - adresacja i nazewnictwo. Adresacja IP i omówienie DNS.**
6. **Warstwa aplikacji - zastosowania internetowe.** Omówienie protokołu HTTP, SMTP i POP3, telnet, FTP, adresacji URL i HTML. Video Streaming oraz Content Distribution Networks
7. **Warstwa transportu – wprowadzenie.** Zasady pewnego przesyłania danych w sieciach komputerowych. Algorytmy Go-Back-N i Selective Repeat.
8. **Warstwa transportu.** Protokół TCP oraz struktura segmentu TCP, protokół UDP.
9. **Warstwa sieci – wprowadzenie.** Zasady routingu, algorytm routingu Link State.
10. **Warstwa sieci.** Algorytm routingu Distance Vector. Routing hierarchiczny i BGP
11. **Data Plane oraz Control Planes.** Generalized Forwarding oraz SDN.
12. **Warstwa łącza danych - dostęp do medium.** Kodowanie sygnałów w sieci. Rodzaje protokołów dostępu do medium (MAC), adresowanie fizyczne MAC. Techniki wykrywania błędów.
13. **Warstwa łącza danych – technologie.** Prekursor Ethernet-u - ALOHA net. Protokół CSMA/CD. Ethernet oraz protokół ARP. Token Ring oraz FDDI.Wi-Fi.
14. **Warstwa łącza danych - urządzenia oraz elementy projektowania sieci lokalnych.** Karty sieciowe, modemy, koncentratory, mostki, przełączniki
15. **Video streaming oraz CDN content distribution networks**
    - Multimedia: video
    - Content distribution networks (CDNs
    - Przykład: Netflix
16. **Ewolucja funkcjonalności warstwy transportu**
    - QUIC: Quick UDP Internet Connections
17. **Generalized Forwarding, SDN**
    - dopasuj+wykonaj
    - OpenFlow
    - Middleboxes
18. **Software defined networking (SDN) control plane**
    - Komponenty Kontrolera SDN
    - Protokół OpenFlow
19. **Bezpieczeństwo sieci komputerowych.** Zagrożenia bezpieczeństwa, techniki włamań, metody zapewniania bezpieczeństwa. Zarys technologii DES, RSA, podpis cyfrowy oraz PGP.

Treści zajęć laboratoryjnych

1 **Adresowanie IP.** Znaczenie klas adresów IP, podsieci, masek podsieci, konfiguracja adresu IP na interfejsach routera.
2 **Elementy implementacji aplikacji sieciowej typu klient/serwer:** Prosty klient TCP i serwer TCP.
3 **Elementy implementacji aplikacji sieciowej typu klient/serwer c.d:** Prosty klient UDP i serwer UDP.
4 **Podstawy konfigurowania sieci LAN.** Identyfikacja podstawowych urządzeń sieciowych, standardy okablowanie, wprowadzenie do Cisco Packet Tracer, tworzenie prostych sieci LAN.
5 **Elementy administrowania i zarządzania sieciami.** Wykorzystanie programów do weryfikacji konfiguracji sieci: ipconfig, ping, pathping, traceroute, arp, netstat, nbtstat.
6 **Podstawy konfigurowania routerów Cisco.** Weryfikacja i modyfikacja plików konfiguracyjnych routera. Instalacja, konfiguracja i umiejętność wykorzystania serwera TFTP do wysyłania i odbierania plików.
7 **Konfiguracja i weryfikacja działania protokołu RIP w sieciach LAN.** Znaczenie routingu. Cechy i wersje protokołu RIP.
8 **Konfiguracja i weryfikacja działania protokołu OSPF w sieciach LAN.** Cechy protokołu OSPF. Routing hierarchiczny i obszary autonomiczne. Pojęcie redystrybucji protokołów routingu.
9 **Konfigurowanie routingu statycznego.** Pojęcie trasy domyślnej. Ręczne wprowadzaniu tras do tablicy routingu.
10 **Zespołowa realizacja zadanej topologii sieci.** Adresowanie sieci dysponując ”nie zaadresowaną” topologią. Symulacja zaprojektowanej sieci.
11 **Konfigurowanie usług DNS, DHCP i NAT na routerach.** Znaczenie i funkcjonowanie usług NAT, DNS i DHCP w sieciach komputerowych.
12 **Konfigurowanie sieci VLAN na przełącznikach.** Tworzenie nowych sieci VLAN, przypisywanie portów przełącznika do konkretnych VLAN-ów, konfigurowanie połączeń typu trunk,
13 **Konfigurowanie standardowych list kontroli dostępu (ACL).** Tworzeniu reguł filtrujących ruch sieciowy na podstawie adresu źródłowego pakietów IP. Pojęcie interfejsy wyjściowego i wejściowego.
14 **Konfigurowanie rozszerzonych list kontroli dostępu (ACL).** Zezwalanie lub blokowanie pakietów na podstawie określonych kryteriów, takich jak adres źródłowy, adres docelowy, protokół czy numer portu.
15 **Obrona zadań indywidualnych.** Testowanie i weryfikacja poprawności realizacji projektu indywidualnego. Możliwe wprowadzenie niewielkich modyfikacji do projektu, w celu weryfikacji samodzielności jego wykonania.

Literatura podstawowa:

1. James Kurose, Keith Ross - Computer Networking_ A Top-Down Approach, 7th Edition (2017) and 8th Edition (2022)
2. K. Krysiak. Sieci Komputerowe - Kompendium. Wydawnictwo Helion 2005
3. Cloud Native Architectures: Design high-availability and cost-effective applications for the cloud. 1st ed. Packt Publishing, 2018. Web. 14 Oct. 2022.

Literatura dodatkowa:

1. Jim Doherty. SDN and NFV Simplified: A Visual Guide to Understanding Software Defined Networks and Network Function Virtualization. Copyright © 2016 Pearson Education, Inc. ISBN-13: 978-0-13-430640-7, ISBN-10: 0-13-430640-6
2. Akademia Sieci Cisco. CCNA Exploration, Semestr 1-4. PWN, Warszawa 2011

3. Leinwand, B. Pinsky. Konfiguracja Routerów Cisco. Podstawy. Mikom, Warszawa 2002.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, Laboratoria z wykorzystaniem sprzętu sieciowego. Zamieszczanie na stronach internetowych zadań i materiałów do ćwiczeń

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
U_01	jest sprawdzany na laboratorium.
U_02	jest sprawdzany w czasie ocenianych zadań na laboratoriach.
U_03	jest sprawdzany w czasie ocenianych zadań na laboratoriach.
U_04	jest sprawdzany w czasie ocenianych zadań na laboratoriach.
W_01	Sprawdzany jest na egzaminie. Przykładowe pytania: ● *Omów topologie fizyczne i logiczne w sieciach komputerowych.* *Porównaj architektury klient-serwer i peer-to-peer.*
W_02	sprawdzany jest na egzaminie. Przykładowe pytania: ● *Wymień warstwy modelu OSI i omów rolę tych warstw.* *Podaj przykłady protokołów działających w poszczególnych warstwach. Uzasadnij dlaczego ten protokół działa w tej warstwie.*
W_03	sprawdzany jest na egzaminie. Przykładowe pytania: ● *Technologia Ethernet. Format ramki Ethernet.* ● *Protokół IP, format pakietu IP.* ● *Protokół TCP, nawiązywanie połączenia w TCP. Porty dobrze znanych usług.* *Protokół UDP, różnice pomiędzy TCP i UDP.*
W_04	Sprawdzany jest na egzaminie. Przykładowe pytania: ● *Omów adresację IPv4, Podziel sieć na podsieci, określ typ adresu (hosta, rozgłoszeniowy, sieci).* *Omów protokół http/smpt/pop3/ftp.*
W_05	sprawdzany jest na egzaminie. Przykładowe pytanie: *Zaprojektuj adresację małej sieci.*
W_06	sprawdzany jest na egzaminie. Przykładowe pytanie efektu: *Dobierz odpowiednią topologię fizyczną i logiczną dla małej organizacji lub firmy.*
K_01	sprawdzany jest na egzaminie. Przykładowe pytania: *1. Wpływ Chmur obliczeniowych na ewolucję Internetu.* *Nowe trendy w rozwoju globalnej sieci komputerowej*

Forma i warunki zaliczenia:

Moduł kończy się egzaminem. Do egzaminu mogą przystąpić osoby, które uzyskały zaliczenie laboratorium. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim oraz z samodzielnie wykonanego zadania indywidualnego według schematu:

- Regularne zajęcia – 26 pkt.,
- Obrona zadania indywidualnego – 14 pkt.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej połowy punktów z poszczególnych form aktywności studenta: regularne zajęcia – co najmniej 13 pkt., obrona indywidualnego zadania – co najmniej 7 pkt. Na tej formie zajęć student może maksymalnie uzyskać 40 pkt.

Egzamin jest egzaminem ustnym. Można na nim uzyskać do 60 pkt. Egzamin będzie zaliczony w przypadku uzyskania, co najmniej 30 pkt. Ocena końcowa z przedmiotu, w zależności od sumy uzyskanych punktów (maksymalnie 100 pkt) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 50 pkt: niedostateczna (F),
- 51 – 60 pkt: dostateczna (E),
- 61 – 70 pkt: dostateczna plus (D),
- 71 – 80 pkt: dobra (C),
- 81 – 90 pkt: dobra plus (B),
- 91 – 100 pkt: bardzo dobra (A).

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	30 godz.
Udział w ćwiczeniach laboratoryjnych	30 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	22 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Obecność na egzaminie	1 godzina
Przygotowanie się do egzaminu	14 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	**4 ECTS**

Studia niestacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	18 godz.
Udział w ćwiczeniach laboratoryjnych	21 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	36 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Obecność na egzaminie	1 godzina
Przygotowanie się do egzaminu i	21 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**

Punkty ECTS za przedmiot	**4 ECTS**

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Bazy danych**
Nazwa w języku angielskim:	Database
Język wykładowy:	polski
Kierunek studiów, dla którego przedmiot jest oferowany:	informatyka
Jednostka realizująca:	Instytut Informatyki
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia
Rok studiów:	drugi
Semestr:	trzeci
Liczba punktów ECTS:	**5**
Imię i nazwisko koordynatora przedmiotu:	Dr Artur Niewiadomski
Imię i nazwisko prowadzących zajęcia:	Dr Artur Niewiadomski
Założenia i cele przedmiotu:	Zakłada się, że student ma opanowane podstawy logiki matematycznej i teorii zbiorów, a także zna podstawy programowania. Celem zajęć jest zapoznanie studentów z podstawami teoretycznymi oraz z zasadami projektowania baz danych, a także nabycie umiejętności praktycznych w środowisku MySQL.

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	podstawowe zagadnienia z zakresu baz danych, wykorzystywanych modeli, projektowania relacyjnych baz danych i języków zapytań	**K_W08**
W_02	podstawowe modele oraz podstawowe zasady projektowania baz danych oraz język zapytań SQL	**K_W08**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	pozyskiwać informacje na temat relacyjnych baz danych z literatury oraz innych źródeł, w tym zwłaszcza internetowych; potrafi integrować uzyskane informacje, dokonywać ich interpretacji, a także wyciągać wnioski oraz formułować i uzasadniać opinie	**K_U01**
U_02	zaprojektować, zaimplementować oraz przetestować prosty bazodanowy system informatyczny	**K_U10, K_U16, K_U19**
U_03	rozwiązywać praktyczne zadania inżynierskie w dziedzinie relacyjnych baz danych wymagające korzystania ze standardów i norm inżynierskich oraz stosowania technologii informatycznych, wykorzystując doświadczenie zdobyte w środowisku zajmującym się zawodowo działalnością inżynierską	**K_U24**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	krytycznej oceny posiadanej wiedzy oraz jest gotów do uznawania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych z zakresu informatyki	**K_K01**

Forma i typy zajęć:	studia stacjonarne: wykłady (30 godz.), ćwiczenia laboratoryjne (30 godz.) studia niestacjonarne: wykłady (18 godz.), ćwiczenia laboratoryjne (21 godz.)

Wymagania wstępne i dodatkowe:

Warunkiem uczestnictwa w zajęciach jest wcześniejsze uzyskanie zaliczenia z następujących przedmiotów:

- Matematyka I, Matematyka II, Matematyka III
- Architektura systemów komputerowych, Podstawy programowania, Programowanie obiektowe

lub znajomość literatury obowiązującej w tych przedmiotach. Student musi mieć opanowane podstawy logiki matematycznej i teorii zbiorów. Ponadto wymagana jest znajomość podstaw strukturalnego lub obiektowego języka programowania oraz architektury systemów komputerowych.

Treści modułu kształcenia:

Wykład:

1. **Wprowadzenie do baz danych.** Historia systemów baz danych. Rodzaje baz danych (hierarchiczne, sieciowe, relacyjne, NoSQL).Zalety stosowania DBMS. Przykłady zastosowań.
2. **Relacyjny model danych.** Podstawowe pojęcia: tabela, rekord, atrybut. Klucze podstawowe i obce. Operacje na relacjach. Spójność i integralność danych.
3. **Algebra relacyjna i zapytania proste.** Operacje algebry relacyjnej. Selekcja, projekcja, złączenia, różnica, iloczyn kartezjański. Wprowadzenie do SQL.
4. **Język SQL: SELECT i filtracja danych.** Składnia SELECT, WHERE, ORDER BY. Operatory logiczne i porównania. DISTINCT, aliasy, funkcje tekstowe i arytmetyczne.
5. **Agregacja i grupowanie danych.** Funkcje agregujące (SUM, AVG, COUNT, MIN, MAX). GROUP BY i HAVING. Podzapytania skalarne.
6. **Złączenia.** Złączenia wewnętrzne (INNER JOIN). Złączenia zewnętrzne (LEFT, RIGHT, FULL). Złączenia z warunkiem i z aliasami.
7. **Podzapytania i zapytania złożone.** Podzapytania w klauzulach FROM, WHERE. EXISTS, IN, NOT IN. ANY i ALL. Podstawy CTE (Common Table Expressions).
8. **Projektowanie baz danych: modelowanie ER.** Diagramy ER (Entity-Relationship). Encje, atrybuty, relacje. Kardynalność i atrybuty złożone.
9. **Przekształcanie modelu ER do relacyjnego.** Mapowanie encji i relacji. Uwzględnianie kluczy obcych. Złożone zależności.
10. **Normalizacja baz danych.** Redundancja i anomalia danych. Formy normalne: 1NF, 2NF, 3NF, BCNF. Przykłady dekompozycji.
11. **Język SQL: operacje DML i DDL.** INSERT, UPDATE, DELETE. Tworzenie i modyfikacja tabel (CREATE, ALTER, DROP). Klucze i ograniczenia (PRIMARY, FOREIGN, CHECK, UNIQUE).
12. **Indeksy i optymalizacja zapytań.** Tworzenie indeksów, wpływ na wydajność. Wyjaśnianie zapytań (EXPLAIN). Strategie optymalizacji.
13. **Transakcje i kontrola współbieżności.** Transakcje: ACID. COMMIT, ROLLBACK, SAVEPOINT. Blokady, problemy współbieżności.
14. **Bezpieczeństwo i uprawnienia.** Użytkownicy i role. GRANT, REVOKE. Backup i przywracanie danych. Perspektywy (Views).
15. **Bazy nierelacyjne (NoSQL) – wprowadzenie.** Modele dokumentowe (MongoDB), klucz-wartość, grafowe. Kiedy warto stosować NoSQL. Krótkie porównanie z RDBMS.

Laboratorium:

1. **Wprowadzenie do środowiska MySQL.** Instalacja i konfiguracja. Tworzenie bazy danych i pierwszej tabeli. Praca z GUI i terminalem.
2. **Podstawowe zapytania SELECT.** Wybieranie danych z tabel. Użycie WHERE, ORDER BY. Operatory logiczne i porównania.
3. **Agregacje i grupowanie.** Funkcje agregujące. GROUP BY i HAVING. Operacje na liczbach, tekstach, NULL.

4. **Złączenia.** INNER JOIN – dwa i więcej źródeł danych. LEFT/RIGHT JOIN – różnice i przykłady.
5. **Złożone złączenia i podzapytania.** Zagnieżdżone zapytania. EXISTS, IN, NOT IN. Przykłady zapytań raportowych.
6. **Operacje modyfikujące dane.** INSERT INTO. UPDATE z warunkiem. DELETE z podzapytaniem.
7. **Tworzenie i modyfikacja struktury bazy.** CREATE TABLE z ograniczeniami (PRIMARY, FOREIGN). ALTER TABLE – dodawanie kolumn, indeksów. DROP TABLE, TRUNCATE.
8. **Projektowanie modelu danych (cz. 1).** Projektowanie diagramu ER. Identyfikacja encji i relacji.
9. **Projektowanie modelu danych (cz. 2).** Tworzenie relacyjnego schematu z modelu ER. Implementacja bazy w SQL.
10. **Normalizacja – ćwiczenia praktyczne.** Analiza błędnych schematów. Dekompozycja do 3NF. Weryfikacja integralności danych.
11. **Indeksy i wydajność.** Tworzenie i testowanie indeksów. Analiza zapytań z EXPLAIN. Przykłady zapytań z dużą liczbą rekordów.
12. **Transakcje i współbieżność.** Testowanie ACID. Symulacja błędów i ROLLBACK. Blokady i konflikty.
13. **Zarządzanie użytkownikami i uprawnieniami.** Tworzenie użytkowników. Nadawanie i odbieranie uprawnień. Testowanie ograniczeń dostępu.
14. **Projekt zaliczeniowy – przygotowanie.** Projekt bazy danych wg własnego scenariusza. Diagram ER + schemat SQL. Przykładowe zapytania.
15. **Projekt zaliczeniowy – prezentacja.** Prezentacja projektów studentów. Omówienie zastosowanych technik. Zaliczenie laboratorium.

Literatura podstawowa:

1. M. J. Hernandez: Projektowanie baz danych dla każdego. Przewodnik krok po kroku. Wydanie IV, Helion 2022
2. V. M. Grippa, S. Kuzmichev, MySQL. Jak zaprojektować i wdrożyć wydajną bazę danych. Wydanie II, Helion 2022
3. A. Beaulieu, Wprowadzenie do SQL. Jak generować, pobierać i obsługiwać dane. Wydanie III, Helion 2021

Literatura dodatkowa:

1. P. DuBois, MySQL. Vademecum profesjonalisty. Wydanie V. Helion 2014.
2. M. Lis, MySQL. Darmowa baza danych. Ćwiczenia praktyczne. Wydanie II. Helion 2013

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, laboratoria – praktyczna praca na komputerze. Zamieszczanie na stronie internetowej zagadnień teoretycznych i zadań ćwiczeniowych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Egzamin pisemny, przykładowe zadanie: Wyjaśnij pojęcia: baza danych, relacyjny model danych.
W_02	Egzamin pisemny, przykładowe zadanie: Zdefiniuj zestaw zapytań SQL dla podanej relacyjnej bazy danych
U_01	Sprawdzane sukcesywnie i oceniane po każdych laboratoriach oraz przy obronie projektu. Przykładowe zadanie laboratoryjne: Uwzględniając strukturę bazy danych i podane zależności funkcyjne określ w jakiej postaci normalnej jest podana baza danych.

U_02	Sprawdzane sukcesywnie i oceniane po każdych laboratoriach oraz przy obronie projektu. Przykładowe zadanie laboratoryjne: Podaj polecenie SQL umożliwiające pobranie z bazy danych rekordów studentów o stypendium większym od średniej kwoty stypendium liczonej dla studentów danego roku i kierunku.
U_03	Sprawdzane sukcesywnie i oceniane po każdych laboratoriach oraz przy obronie projektu. Przykładowe zadanie laboratoryjne: Zaproponuj strukturę podanej bazy danych właściwą dla 1NF, 2NF, 3NF postaci normalnej.
K_01	Obserwacja postawy studenta, jego umiejętności analizy i syntezy poszczególnych informacji podczas zajęć i konsultacji.

Forma i warunki zaliczenia:

Moduł kończy się egzaminem. Do egzaminu mogą przystąpić osoby, które uzyskały zaliczenie laboratorium. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim oraz z samodzielnie wykonanego zadania indywidualnego według schematu:

- regularne zajęcia – 30 pkt.
- zadanie indywidualne – 20pkt.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania powyżej połowy punktów za regularne zajęcia (15 pkt) i co najmniej połowę punktów za zadanie indywidualne (10pkt.). Na tej formie zajęć student może maksymalnie uzyskać 50 pkt.
Egzamin jest egzaminem pisemnym. Można na nim uzyskać do 50 pkt. Egzamin będzie zaliczony w przypadku uzyskania powyżej 25 pkt.
Ocena końcowa, w zależności od sumy uzyskanych punktów (maksymalnie 100 pkt) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 50 pkt: niedostateczna (F),
- 51 – 60 pkt: dostateczna (E),
- 61 – 70 pkt: dostateczna plus (D),
- 71 – 80 pkt: dobra (C),
- 81 – 90 pkt: dobra plus (B),
- 91 – 100 pkt: bardzo dobra (A).

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	30 godz.
Udział w ćwiczeniach laboratoryjnych	30 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych i opracowanie projektu	37 godz.
Udział w konsultacjach godz. z przedmiotu	2 godz.
Obecność na egzaminie	2 godz.
Przygotowanie się do egzaminu	24 godz.
Sumaryczne obciążenie pracą studenta	125 godz.
Punkty ECTS za przedmiot	**5**

Studia niestacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	18 godz.
Udział w ćwiczeniach laboratoryjnych	21 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych i opracowanie projektu	48 godz.
Udział w konsultacjach godz. z przedmiotu	2 godz.
Obecność na egzaminie	2 godz.
Przygotowanie się do egzaminu	34 godz.
Sumaryczne obciążenie pracą studenta	125 godz.
Punkty ECTS za przedmiot	**5**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Programowanie niskopoziomowe**	
Nazwa w języku angielskim:	Low level programming	
Język wykładowy:	polski	
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka	
Jednostka realizująca:	Instytut Informatyki	
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	fakultatywny	
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia	
Rok studiów:	drugi	
Semestr:	trzeci	
Liczba punktów ECTS:	3	
Imię i nazwisko koordynatora przedmiotu:	dr Andrzej Salamończyk	
Imię i nazwisko prowadzących zajęcia:	dr Andrzej Salamończyk mgr Wojciech Nabiałek	
Założenia i cele przedmiotu:	Zakłada się, że student zna podstawy architektury komputerów i potrafi programować w języku wysokiego poziomu. Celem modułu jest zapoznanie ze specyfiką programowania niskopoziomowego na przykładzie asemblera MASM dla procesora x86.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	architekturę procesorów z rodziny x86 i sposoby programowania niskopoziomowego na tych procesorach	**K_W05**
W_02	rolę i zastosowanie asemblerów w systemach informatycznych	**K_W05**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	posługiwać się językiem asemblera MASM używając: instrukcji warunkowych, pętli, operacji na liczbach całkowitych, tablic i łańcuchów znaków	**K_U10, K_U22**
U_02	tworzyć w języku asemblera aplikacje konsolowe i okienkowe	**K_U10**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	krytycznej oceny posiadanej wiedzy oraz jest gotów do uznawania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych z zakresu programowania niskopoziomowego	**K_K01**

Forma i typy zajęć:	studia stacjonarne: wykłady (21 godz.), ćwiczenia laboratoryjne (24 godz.) studia niestacjonarne: wykłady (15 godz.), ćwiczenia laboratoryjne (15 godz.)

Wymagania wstępne i dodatkowe:

1. Wiedza na temat architektury komputerów
2. Wiedza na temat podstaw programowania oraz umiejętność programowania w jednym z języków wysokiego poziomu

Treści modułu kształcenia:

Wykład

1. Podstawy asemblera. Rola i znaczenie asemblerów. Narzędzia programowania. Tworzenie programu w języku asemblera. Reguły zapisu programu w języku asemblera.
2. Systemy komputerowe (komputery) na bazie procesorów firmy Intel. Rejestry procesorów Intel. Rozkazy procesorów. Adresowanie operandów. Znaczniki wyników operacji.
3. Assembler Microsoft Macro Assembler (MASM). Elementy języka asemblera MASM. Instrukcje. Dyrektywy. Wskaźnik pozycji. Przesyłanie danych.
4. Operacje arytmetyczne i logiczne. Operacje bitowe.
5. Sterowanie wykonaniem programu w języku Macro Assembler (MASM). Porównania. Skoki. Pętle.
6. Procedury. Makroinstrukcje. Tworzenie własnych procedur i makr.
7. Programowanie z zastosowaniem funkcji API Win32. Programowanie aplikacji konsolowej. Operacje na wierszach. Operacje na plikach.
8. Stosowanie jednostki zmiennoprzecinkowej i jednostki MMX. Alokacja i przesyłanie danych. Operacje arytmetyczne. Operacje trygonometryczne. Operacje porównania.
9. Programowanie aplikacji graficznych. Współdziałanie aplikacji graficznej z systemem Windows. Tworzenie okna. Standardowe obiekty graficzne. Kontekst urządzenia. Korzystanie z zasobów.
10. Programowanie wstawek asemblerowych w języku C (C++). Wywoływanie w języku C (C++) funkcji napisanych w języku asemblera. Wywoływanie w asemblerze funkcji napisanych w języku C (C++).
11. Przegląd języków niskopoziomowych.

Laboratorium

1. Tworzenie i uruchamianie programów asemblerowych. Przejście do trybu konsolowego. Kompilacja, opcje kompilatora. Konsolidacja, opcje konsolidatora. Opracowanie aplikacji konsolowej
2. Zarządzanie danymi. Przesyłanie danych. Praca z łańcuchami. Operacje na stosie. Tryby adresowania.
3. Operacje arytmetyczne i logiczne. Przesuwanie i rotacja bitów.
4. Sterowanie przebiegiem wykonania programu. Porównania i skoki warunkowe. Pętle.
5. Podprogramy i makrodefinicje.
6. Operacje na plikach i katalogach. Tworzenie plików i katalogów. Otwieranie i zapisywanie.
7. Obsługa sprzętu. Klawiatura i mysz. Tworzenie okna konsolowego.
8. Koprocesor i jednostka MMX.
9. Tryb graficzny. Tworzenie okna. Obiekty graficzne.
10. Korzystanie z plików zasobów.
11. Wstawki asemblerowe.
12. Prezentacja i zaliczenie zadania indywidualnego.

Literatura podstawowa:

1. Timofiejew. Praktyczny kurs programowania w językach asemblerów. Wydawnictwo UPH 2012
2. Vlad Pirogow. Asembler. Podręcznik programisty. Helion 2005

Literatura dodatkowa:

1. Błaszczyk. Win32ASM. Asembler w Windows. Helion 2004
2. K. R. Irvine. Asembler dla procesorów Intel – Vademecum profesjonalisty. Helion 2003

3. E. Wróbel. Programowanie w języku asemblera MASM. Wydaw. Pol. Śląskiej, 2006

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi. Zamieszczanie na stronach internetowych zadań i materiałów do laboratoriów.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się

W_01:	Efekt weryfikowany na egzaminie pisemnym. Przykładowe pytanie:
	Określ czy następująca operacja jest dopuszczalna w języku MASM (wybrane przykłady):
	mov si, dword ptr [esi]
	mov zmA,zmB
	Określ jaka będzie zawartość rejestru eax po wykonaniu następującego fragmentu kodu (przykład)
	xor eax,eax
	mov ecx,10
	petla:
	inc eax
	loop petla

W_02	Efekt weryfikowany na egzaminie pisemnym. Przykładowe pytania: Jakie są główne zastosowanie programu asemblerowego? Na czym polega etap asemblacji i konsolidacji i jak jest realizowany w MASM32.
U_01	Efekt weryfikowany w czasie zadań laboratoryjnych. Przykładowe zadanie: Napisz program w asemblerze MASM, który: Wczytuje do pamięci tablicę 10 liczb całkowitych (możesz zainicjalizować ją wartościami na stałe). Oblicza sumę wszystkich liczb w tablicy, wykorzystując pętlę. Sprawdza, czy suma jest większa od określonej wartości progowej (np. 100) — użyj instrukcji warunkowej. Jeśli suma jest większa niż próg, wyświetla komunikat tekstowy „Suma jest duża”, w przeciwnym razie „Suma jest mała”.
U_02	Efekt weryfikowany w czasie zadań laboratoryjnych. Przykładowe zadanie: Napisz prostą aplikację okienkową w asemblerze MASM, która: Tworzy okno z tytułem Reaguje na zdarzenie zamknięcia okna (przycisk „X”), kończąc program. Wyświetla w oknie prosty tekst (np. „Witaj w oknie!”) po naciśnięciu przycisku.
K_01	Obserwacja postaw studenta w czasie zadań laboratoryjnych.

Forma i warunki zaliczenia:

Ocena końcowa jest wystawiana na podstawie zajęć laboratoryjnych i kolokwium pisemnego. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na laboratoriach oraz z samodzielnie wykonanego zadania indywidualnego:

- Oceniane laboratoria (10 zajęć po 10 pkt.)
- Zadanie indywidualne 40 pkt

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej połowy punktów tj. co najmniej 71 pkt. Za pisemne kolokwium można na nim uzyskać 100 pkt. Ostateczny wynik punktowy modułu oblicza się wg wzoru:

P=60(L/140)+40(E/100),

gdzie P-końcowy wynik punktowy (maksymalnie 100 pkt.) , L-punkty uzyskane z części laboratoryjnej,

E-punktowy wynik pisemnego kolokwium

Ocena z zajęć zależy od końcowego wyniku punktowego i wyznacza się w następujący sposób.

- 0-50 punktów – 2
- 51-60 punktów – 3
- 61-70 punktów - 3,5
- 71-80 punktów – 4
- 81-90 punktów – 4,5
- 91-100 punktów – 100

Sposób uzyskania punktów:

Laboratorium

Ocena udziału w laboratoriach oraz przygotowania się do tych zajęć: 140 pkt. (14 zajęć po 10 pkt.).

Wykład

Kolokwium pisemne: 100 pkt.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	21 godzin
Udział w ćwiczeniach laboratoryjnych	24 godziny
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	14 godziny
Udział w konsultacjach godz. z przedmiotu	3 godziny
Przygotowanie się do kolokwium	13 godzin
Sumaryczne obciążenie pracą studenta	**75 godzin**
Punkty ECTS za przedmiot	**3 ECTS**

Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godzin
Udział w ćwiczeniach laboratoryjnych	15 godziny
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	21 godzin
Udział w konsultacjach godz. z przedmiotu	3 godziny
Przygotowanie się do się do kolokwium	21 godzin
Sumaryczne obciążenie pracą studenta	**75 godzin**
Punkty ECTS za przedmiot	**3 ECTS**

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Wybrane paradygmaty programowania**
Nazwa w języku angielskim:	Selected Programming Paradigms
Język wykładowy:	Polski
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka
Jednostka realizująca:	Instytut Informatyki
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	fakultatywny
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia):	pierwszego stopnia
Rok studiów:	Drugi
Semestr:	Trzeci
Liczba punktów ECTS:	4
Imię i nazwisko koordynatora przedmiotu:	dr hab. inż. Jerzy Tchórzewski, prof. Uczelni
Imię i nazwisko prowadzących zajęcia:	dr hab. inż. Jerzy Tchórzewski, prof. uczelni dr Artur Niewiadomski mgr Wojciech Nabiałek
Założenia i cele przedmiotu:	Zakłąda się, że student zna podstawy programowania, wybrane zagadnienia z algorytmów i struktur danych oraz posiada umiejętność samodzielnego rozwiązywania problemów i ich implementacji. Celem zajęć jest zapoznanie studentów z wybranymi paradygmatami programowania oraz wykorzystanie ich do praktycznej implementacji wybranych problemów

Symbol efektu	Efekty uczenia się: WIEDZA Student zna i rozumie:	Symbol efektu kierunkowego
W_01	zagadnienia na temat popularnych i niszowych paradygmatów programowania i zna ich charakterystykę.	**K_W06**
W_02	zagadnienia na temat podstaw programowania deklaratywnego, w tym zna: klauzulową postać programów, rachunek zdań, rachunek predykatów, rezolucję i dowodzenie, programowanie w logice, mechanizmy wnioskowania, obiekty i relacje.	**K_W06**
W_03	zagadnienia z zakresu programowania w języku Prolog, w tym zna: składnię, znaki i operatory, struktury danych (m.in. pojęcie atomu, termu, zmiennej, struktury, drzewa, listy, fakty, itp.), zapytania, zmienne, koniunkcje, reguły, arytmetykę, itp. oraz posiada wiedzę o innych językach programowania deklaratywnego takich jak: LISP, CLIPS, itp.	**K_W06**
W_04	zagadnienia z zakresu: celów, nawracania i odcięcia, równości i unifikacji, przeszukiwania i porównywania rekurencyjnego,	**K_W06**

	odwzorowania, łączenie struktur, akumulatorów, struktur różnicowych, itp.	
W_05	zagadnienia na temat urządzeń wejścia i wyjścia, czytania i pisania termów, zdań, plików, predykatów wbudowanych, tworzenia celów złożonych, obsługi plików, itp.	**K_W06**
W_06	zagadnienia z zakresu: przetwarzania list, zbiorów, sortowania, tworzenia bazy wiedzy, przeszukiwania grafów, śledzenia wykonywania programów, itp.	**K_W06**
W_07	podstawowe konstrukcje i typy danych wykorzystywane w programowaniu skryptowym, proceduralnym i obiektowym, wspierane przez język Python.	**K_W06**
W_08	charakterystyczne cechy funkcyjnego paradygmatu programowania. Zna podstawowe konstrukcje i typy danych wykorzystywane w programowaniu funkcyjnym wspierane przez język Python	**K_W06**
W_09	koncepcję lambda-wyrażeń, częściowej aplikacji argumentów funkcji, funkcji wyższego rzędu oraz typowe funkcje wyższego rzędu wspierane przez język Python, m.in. map, filter, reduce, zip.	**K_W06**
W_10	ideę rekurencji, akumulatora, oraz spamiętywania i uleniwiania obliczeń.	**K_W06**
Symbol efektu	**Efekty uczenia się: UMIEJĘTNOŚCI** **Student potrafi:**	**Symbol efektu kierunkowego**
U_01	samodzielnie projektować i implementować proste systemy ekspertowe w środowisku SWI Prolog, przygotowywać projekty w postaci drzewa genealogicznego i na tej podstawie budować Bazę Wiedzy, w tym Bazę Faktów, Bazę Reguł, Bazę Zapytań oraz umie poprawiać i usuwać błędy w programach prologowych.	**K_U22**
U_02	zaprojektować z wykorzystaniem drzewa celów i zaimplementować w SWI Prolog oraz przetestować zaawansowany System Ekspertowy z wykorzystaniem list, arytmetyki, rekurencji, predykatów wbudowanych oraz posiada umiejętność modyfikacji Bazy Wiedzy, w tym umiejętność jej rozbudowywania, a także umie rozwijać swoje umiejętności programowania w językach programowania deklaratywnego.	**K_U22**
U_03	projektować proste programy z wykorzystaniem inspirowanych kwantowo algorytmów klasycznych i je implementować na klasycznych komputerach z wykorzystaniem języków programowania bardzo wysokiego poziomu, takich jak m.in. j. Matlab lub j. Python.	**K_U22, K_U19**
U_04	sprawnie korzystać ze środowiska PyCharm (lub alternatywnego) w zakresie tworzenia aplikacji w języku Python.	**K_U11**
U_05	definiować funkcje w języku Python, w tym funkcje rekurencyjne i funkcje wyższego rzędu. Potrafi wykorzystać deklarację sygnatury funkcji w celu lepszej kontroli nad typami danych.	**K_U22**

U_06	wykorzystać mechanizmy typowe dla paradygmatu funkcyjnego, obiektowego, strukturalnego do rozwiązania postawionego problemu.	**K_U22**
Symbol efektu	**KOMPETENCJE SPOŁECZNE Student jest gotów do:**	**Symbol efektu kierunkowego**
K_01	krytycznej oceny posiadanej wiedzy oraz jest przygotowany do uznania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych związanych z zagadnieniami programowania deklaratywnego, funkcyjnego, a także inspirowanych kwantowo zagadnień programowania obiektowego, systemowego i strukturalnego.	**K_K01**

Forma i typy zajęć:	studia stacjonarne: wykłady (21 godz.), ćwiczenia laboratoryjne (24 godz.) studia niestacjonarne: wykłady (15 godz.), ćwiczenia laboratoryjne (15 godz.)

Wymagania wstępne i dodatkowe:

1. Wiedza z podstaw logiki matematycznej, rachunku zdań i rachunku predykatów.
2. Znajomość podstaw programowania, w tym imperatywnego, obiektowego, strukturalnego i systemowego oraz wybranych zagadnień z algorytmów i struktur danych.
3. Umiejętność samodzielnego rozwiązywania problemów i ich implementacji w dowolnych środowiskach programistycznych.

Treści modułu kształcenia:

Część I.1 Wybrane Paradygmaty Programowania Deklaratywnego

1. **Przegląd paradygmatów programowania.** Programowanie imperatywne, systemowe, strukturalne, obiektowe, deklaratywne, funkcyjne, reaktywne, sterowane przepływem danych, itp. Rys historyczny i ewolucja języków programowania (1 godz).
2. **Projekty w języku Prolog oraz wybrane paradygmaty programowania deklaratywnego, w tym projektowanie Bazy Wiedzy w języku Prolog.** Języki programowania deklaratywnego, w tym język Prolog**.** Fakty i reguły w reprezentacji wiedzy. Definicja termu (stała, zmienna, struktura). Formuły atomowe. Prolog i programowanie w logice. Predykat podstawową jednostką języka Prolog. Fakty i reguły w Prologu. Zapytania. Składnia w Prologu. Obiekty i relacje. Równość i unifikacja, itp., konstruowanie Bazy Wiedzy (Drzewo Celów, Baza Faktów, Baza Reguł, Baza Zapytań, itp.). Spełnianie celów w regułach. Reguły proste i złożone. Przetwarzanie podprogramów. Klauzulowa postać Bazy Wiedzy (2 godz).
3. **Rachunek predykatów i rachunek zdań.** Opis obiektów w Prologu. Proste predykaty niedeterministyczne i deterministyczne. Predykaty wejścia i wyjścia. Predykaty wbudowane. Predykaty standardowe. Postać klauzulowa. Zapis klauzul. Rezolucja i dowodzenie twierdzeń. Klauzule Horna. Rachunek predykatów. Przekształcenia w rachunku predykatów prowadzące od postaci predykatowej do postaci klauzulowej. Formalny opis języka predykatów (2 godz.).
4. **Mechanizmy nawrotu i odcięcia w Prologu.** Deklaratywność Prologu. Mechanizmy inteligentnego uzgadniania. Relacje i reguły. Koniunkcje. Nawracanie. Cele i nawracanie. Zapytania i nawroty. Typowe zastosowanie odcięcia. Potwierdzenie wyboru reguły. Użycie predykatu odcięcia. Korzystanie ze struktur danych**.** Arytmetyka i tautologie w Prologu**.** Struktury a drzewa. Listy. Łączenie struktur. Akumulatory. Struktury różnicowe. Złożone struktury danych. Przetwarzanie drzew i list. Przetwarzanie zbiorów (2 godz.).

Część I.2 Wybrane Paradygmaty Programowania Kwantowego

5. **Informatyka Kwantowa.** Kudit, w tym kubit**,** interpretacje kubitu, stany systemu i stany kwantowe, kwantowe stany mieszane, superpozycja stanów wykluczających się, pomiar (odczyt) stanu kwantowego, ogólne operacje kwantowe, przestrzeń Hilberta, w tym macierze unitarne, postulaty informatyki kwantowej, obliczenia kwantowe z wykorzystaniem bramek kwantowych, liczby zespolone w obliczeniach kwantowych, obwody kwantowe w obliczeniach kwantowych, kwantowy stan splątany, dostępne symulatory obliczeń kwantowych, w tym symulator obliczeń kwantowych w środowisku MATLAB-a i Simulink-a, itp.

6. **Obliczenia kwantowe na bramkach kwantowych.** Kwantowe przetwarzanie informacji, rola operatorów w informatyce kwantowej, w tym wartości własnych operatorów hermitowskich, macierz gęstości, w tym wyznaczanie śladu, funkcja falowa, obserwable, synteza obwodów kwantowych, algorytmy kwantowe, teleportacja kwantowa, ewolucja unitarna, rozkład Schmidta stanów kwantowych, inspiracje kwantowe metod klasycznych, itp.

Część II. Wybrane Pradygmaty Programowania Funkcyjnego

7. **Wprowadzenie do języka Python.** Podstawowe typy danych. Operatory. Wyrażenia i ich wartościowanie. Funkcje i ich argumenty. Wyrażenia warunkowe. Pętle.
8. **Funkcje.** Definiowanie funkcji w języku Python. Zwracanie wartości przez funkcję. Sygnatury funkcji. Funkcje czyste. Efekty uboczne funkcji. Funkcje lambda. Domknięcie funkcji. Funkcje rekurencyjne. Rekursja a wykorzystanie stosu.
9. **Typy danych.** Krotki. Zbiory. Listy. Słowniki. Obiekty. Elementy programowania strukturalnego i obiektowego.
10. **Listy.** Własności list. Operacje na listach. Przetwarzanie list za pomocą funkcji rekurencyjnych. Rekursja ogonowa i wykorzystanie akumulatora.

Funkcje wyższego rzędu i obliczenia leniwe. Definicja i przykłady funkcji wyższego rzędu (FWR). Przetwarzanie list za pomocą FWR. Funkcje z modułu List. Obliczenia leniwe. Generatory. Iteratory.

Literatura podstawowa:

1. M. Bramer: Logic Programming with Prolog. Springer-Verlag, London 2013, p. 256.
2. W. F. Clocksin, C. S. Mellish: Prolog. Programowanie. Helion. Warszawa 2003, stron 276.
3. M. Gorelick, I. Ozsvald: Wysoko wydajny Python: efektywne programowanie w praktyce. Helion 2021.
4. E. Matthes: Python. Instrukcje dla programisty. Helion 2016.

Literatura dodatkowa

1. M. Ben-Ari: Logika matematyczna w informatyce. WNT. Warszawa 2001, stron 346.
2. G. Brzykcy, A. Meissner: Programowanie w Prologu i programowanie funkcyjne: materiały do ćwiczeń. Wyd. Politechniki Poznańskiej. Poznań 1999, stron 110.
3. U. Nilsson, J. Małuszynski: Logic. Programming and Prolog. John Wiley & Sons, 1990.
4. J. Tchórzewski: Metody sztucznej inteligencji i informatyki kwantowej w ujęciu teorii sterowania i systemów. Wydawnictwo UPH. Siedlce 2021.
5. Tchórzewski J.: Wykłady i instrukcje laboratoryjne z wybranych paradygmatów programowania deklaratywnego i kwantowego. Monografia dydaktyczna w przygotowywaniu do złożenia. WN UPH, Siedlce (wersja elektroniczna z września 2016 r., ostatnia aktualizacja rozszerzona o paradygmaty kwantowe z września 2022, jest dostępna dla studentów w postaci instrukcji laboratoryjnych oraz Print Screen-ów z wykładów przygotowanych w MS Power Point).

Planowane formy/działania/metody dydaktyczne:

Wykład z aktywną komunikacją wspomagany technikami multimedialnymi. Udostępnianie studentom scanów wykładów w postaci prezentacji w MS Power Point oraz instrukcji do poszczególnych tematów ćwiczeń laboratoryjnych.

Ćwiczenia laboratoryjne są prowadzone według następującego układu tematów i laboratoriów:

Część I. Wybrane Paradygmaty Programowania Deklaratywnego i Kwantowego

Temat 1 (Lab1-2): Projektowanie Drzewa Geneaologicznego i implementacja Bazy Wiedzy w środowisku SWI Prolog na temat rodziny (2 x 2godz.).

Temat 2 (Lab3-4): Projektowanie Drzewa Celów i implemetacja Systemu Ekspertowego w SWI Prolog na wskazany przez prowadzącego problem (2 x 2 godz.).

Temat 3 (Lab5-6): Projektowanie inspirowanych kwantowo prostych algorytmów klasycznych z wykorzystaniem j. Matlab/j. Python/j. Sphinx, itp. lub z wykorzystaniem symulatora środowiska MATLAB i Simulin, jak np. Qubit4MATLA, w uzgodnionym z prowadzącym zakresie (2 x 2 godz.).

Część II.

Temat 4 (Lab 7): Wprowadzenie do języka Python. Środowisko PyCharm, Operatory. Wyrażenia i ich wartościowanie. Funkcje i ich argumenty. Wyrażenia warunkowe. Pętle (2 godz).

Temat 5 (Lab 8): Funkcje. Definiowanie funkcji w języku Python. Zwracanie wartości przez funkcję. Sygnatury funkcji. Funkcje czyste. Efekty uboczne funkcji. Funkcje lambda. Domknięcie funkcji (2 godz.).

Temat 6 (Lab 9-10): Typy danych. Krotki. Zbiory. Listy. Słowniki. Obiekty. Elementy programowania strukturalnego i obiektowego (2 x 2godz.).

Temat 7. (Lab 11): Rekurencja. Funkcje rekurencyjne. Rekursja a wykorzystanie stosu. Własności list. Operacje na listach. Przetwarzanie list za pomocą funkcji rekurencyjnych, wykorzystanie akumulatora (2 godz.).

Temat 8. (Lab 12): Funkcje wyższego rzędu i obliczenia leniwe. Definicja i przykłady funkcji wyższego rzędu (FWR). Przetwarzanie list za pomocą FWR. Obliczenia leniwe. Generatory. Iteratory (2 godz.).

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	**Metody weryfikacji efektu uczenia się**
W_01 – W_10	Będą sprawdzane w formie kolokwium i na egzaminie
U_01 – U_06	Będą sprawdzane przy zaliczeniu każdego z tematów laboratoryjnych.
K_01	Będzie sprawdzany przy każdym kontakcie ze studentem na laboratorium, wykładach, konsultacjach, egzaminie.

Forma i warunki zaliczenia:

Moduł kończy się **egzaminem.**

<u>Zaliczenie laboratorium.</u>

Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na zajęciach z nauczycielem akademickim, za które można łącznie uzyskać maksymalnie 100 p., odpowiednio w części I oraz w części II do 50p. według zasady:

Część I:

ćwiczenia laboratoryjne 1-2 oraz 3-4 (2 tematy projektów indywidualnych x 20 p.) - 40 p.
ćwiczenie laboratoryjne 5-6 (1 temat ćwiczenia laboratoryjnego x 10p) - 10 p.
Razem – 50 p.

Część II:

ćwiczenia laboratoryjne 7, 8, 11, 12 (4 tematy ćwiczeń laboratoryjnych x 5 p.) - 20 p.
ćwiczenie laboratoryjne 9-10 (jeden temat ćwiczeń laboratoryjnych x 10 p.) – 10 p.
kolokwium - 20 p.
Razem – 50 p.

Ogółem (2 x 50p) - 100 p.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania więcej niż połowy punktów z każdej części labratorium (od 51%, to jest **od 26 p**.) ze wszystkich form aktywności studenta odrębnie w zakresie Wybranych paradygmatów programowania deklaratywnego i kwantowego oraz odrębnie w zakresie Wybranych paradygmatów programowania funkcyjnego.

Egzamin

Egzamin pisemny (test lub zadania problemowe) składa się z dwóch części, odrębnie w z Wybranych paradygmatów programowania deklaratywnego i kwantowego oraz odrębie z Wybranych paradygmatów porgramowania funkcyjnego. Za każdą z części egzaminu student może uzyskać do 50 p. Wykłady będą zaliczone w przypadku uzyskania z każdej części egzaminu odrębnie przynajmniej 26 p.

Razem (2 x 50 p) – 100p.

Ocena z każdej części przedmiotu

Ocena z Wybranych paradygmatow programowania deklaratywnego i kwantowego oraz ocena z Wybranych paradygmatów programowania funkcyjnego jest wyznaczana jako średnia ocen uzyskanych z ćwiczeń laboratoryjnych oraz z wykładów odrębnie odpowiedniej części.

Oceny z zaliczenia poszczególnych części przedmiotu (odrębnie z Wybranych paradygmatów programowania deklaratywnego i kwantowego oraz odrębnie z Wybranych paradygmatów programowania funkcyjnego).

Z każdej części przedmiotu student może uzyskać do 100 pkt (do 50 p. z egzaminu i do 50 p. z laboratoriów), przy czym istnieje możliwość uzyskania dodatkowych punktów z tytułu różnych aktywności studentów na ćwiczeniach laboratoryjnych, wykładach oraz po zajęciach w tematyce związanej z realizowanym programem w ramach modułu. W zależności od liczby punktów zdobytych w ramach ocenianej części, odrębnie ćwiczeń laboratoryjnych oraz odrębnie odpowiedniej części wykładów, student otrzymuje ocenę:

- 0-25 pkt. - 2 (ndst)
- 26-30 pkt. - 3 (dst)
- 31-35 pkt. – 3,5 (dst+)
- 36-40 pkt. - 4 (db)
- 41-45 pkt. - 4,5 (db+)
- 46-50 pkt. - 5 (bdb).

Ocena z danej części przedmiotu jest średnią ocen z laboratorium oraz wykładów i wynosi:

- 2,0; 2,5 - ndst (F)
- 3,00 - dst (E)
- 3,25; 3,5 - dst+ (D)
- 3,75; 4,0 - db (C)
- 4,25; 4,5 - db+ (B)
- 4,75; 5,0 - bdb (A).

Ocena końcowa z przedmiotu

Ocena końcowa z przedmiotu jest średnią z obu części modułu, to jest z Wybranych paradygmatów programowania deklaratywnego i kwantowego oraz z Wybranych paradygmatów programowania funkcyjnego. Ocena końcowa z przedmiotu w zależności od uzyskanej średniej jest następująca (w nawiasach ocena wg skali ECTS:

2,0; 2,5 - niedostateczna (F),
3,0 - dostateczna (E),
3,25; 3,5 - dostateczna plus (D),
4,0 - dobra (C),
4,25; 4,5 - dobra plus (B),
5,0 - bardzo dobra (A).

Poprawy:

Istnieje możliwość jednorazowej poprawy kolokwium w toku zajęć laboratoryjnych oraz jednej poprawy w sesji egzaminacyjnej, a także dwukrotnej poprawy jednego z tematów ćwiczeń laboratoryjnych w toku trwania semestru. Poprawa ćwiczeń laboratoryjnych nie jest możliwa po terminie wpisów do USOS.

- Egzamin poprawkowy jest jednokrotny w sesji egzaminacyjnej, odrębnie z każdej części przedmiotu.

Bilans punktów ECTS:	
Studia stacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	21 godzin

Udział w ćwiczeniach laboratoryjnych	24 godziny
Samodzielne przygotowanie się do tematów ćwiczeń laboratoryjnych i realizacja projektów indywidualnych, a także opracowanie sprawozdań	25 godzin
Udział w konsultacjach	3 godziny
Studia indywidualne z tematów realizowanych na wykładach i na ćwiczeniach laboratoryjnych oraz końcowe przygotowanie się do egzaminu pisemnego	25 godzin
Obecność na egzaminie	2 godziny
Sumaryczne obciążenie pracą studenta	**100 godzin**
Punkty ECTS za przedmiot	**4 ECTS**
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godzin
Udział w ćwiczeniach laboratoryjnych	15 godzin
Samodzielne przygotowanie się do tematów ćwiczeń laboratoryjnych i realizacja projektów indywidualnych, a także opracowanie sprawozdań	30 godzin
Udział w konsultacjach z przedmiotu	3 godziny
Studia indywidualne z tematów realizowanych na wykładach i na ćwiczeniach laboratoryjnych oraz końcowe przygotowanie się do sprawdzianu pisemnego	35 godzin
Obecność na egzaminie	2 godziny
Sumaryczne obciążenie pracą studenta	**100 godzin**
Punkty ECTS za przedmiot	**4 ECTS**