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 5

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Przedsiębiorczość indywidualna**
Nazwa w języku angielskim:	**Individual entrepreneurship**
Język wykładowy:	Język polski
Kierunek studiów, dla którego przedmiot jest oferowany:	Informatyka
Jednostka realizująca:	Instytut Nauk o Zarządzaniu i Jakości
Rodzaj przedmiotu/modułu kształcenia (obowiązkowy/fakultatywny):	obowiązkowy
Poziom modułu kształcenia (np. pierwszego lub drugiego stopnia, jednolitych magisterskich):	pierwszego stopnia
Rok studiów:	trzeci
Semestr:	piąty
Liczba punktów ECTS:	**2**
Imię i nazwisko koordynatora przedmiotu:	dr inż. Stanisław Szarek
Imię i nazwisko prowadzących zajęcia:	dr inż. Stanisław Szarek
Założenia i cele przedmiotu:	Celem przedmiotu jest przyswojenie przez studentów wiedzy z zakresu uwarunkowań przedsiębiorczości we współczesnym świecie, nabycie przez nich przedsiębiorczych postaw, sprawdzenia predyspozycji w prowadzeniu własnej firmy, docenienie roli innowacji w gospodarce jak również ukierunkowanie studentów do samodzielnego pogłębiania wiedzy, doskonalenia umiejętności i bycia świadomym uczestnikiem rynku i społeczeństwa

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	zasady funkcjonowania gospodarki rynkowej, identyfikuje rolę państwa i sektora finansowego w gospodarce	K_W13
W_02	atrybuty człowieka przedsiębiorczego, zna sposób zakładania i formy prowadzenia działalności gospodarczej	K_W13
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	przedstawić ograniczenia ludzi i firm związane z ich funkcjonowaniem na rynku	K_U05, K_U06, K_U23,
U_02	wybrać najlepszą formę do prowadzenia różnych rodzajów działalności gospodarczej	K_U23
U_03	współdziałać w grupie, przyjmując w niej różne role	K_U23

	KOMPETENCJE SPOŁECZNE Student jest gotów do:	
K_01	rozpoznania przyczyn i skutków podejmowania różnych decyzji przez jednostki	K_K03,
K_02	wykazania się kreatywnością i angażuje się w poszukiwanie rozwiązania problemów	K_K01, K_K02

Forma i typy zajęć:	Studia stacjonarne: ćwiczenia (30 godz.) Studia niestacjonarne: ćwiczenia (18 godz.)

Wymagania wstępne i dodatkowe:

Brak

Treści modułu kształcenia:

1. Tworzenie grup realizujących projekt i wirtualną firmę z wykorzystaniem testu Belbina i testu na inteligencje wielorakie
2. Zasady funkcjonowania gospodarki rynkowej.
3. Czynniki produkcji w gospodarce (ziemia, praca , kapitał i zarządzanie).
4. Uwarunkowania prawne podejmowania działalności gospodarczej.
5. Źródła finansowania działalności gospodarczej i projektów innowacyjnych.
6. Innowacje w przedsiebiorczości
7. Prowadzenie własnej firmy w środowisku wirtualnym

Literatura podstawowa:

1. Garbacik, K., & Żmiejko, M. (2018). *Przedsiębiorczość na czasie.* (3. wyd.). Warszawa: Nowa Era. ISBN 978-83-267-2371-1.
2. Markowski W., ABC small businessu, Wyd. XIII, Marcus s.c., Łódź 2010
3. Cieślik J., Przedsiębiorczość dla ambitnych. Jak uruchomić własny biznes, Wydawnictwa Akademickie i Profesjonalne, Warszawa 2006

Literatura dodatkowa:

1. Kapusta F., Przedsiębiorczość – teoria i praktyka, Wydawnictwo Forum Naukowe, Poznań – Wrocław 2006
2. Dolna-Ciemniakowska M., A. Wesołowska, Zakładamy firmę, Wyd. Difin, Warszawa 2007
3. Laszczak M., Kierowanie małą firmą-tajniki przedsiębiorczości, Wyd. Poltex, Warszawa 2004

Planowane formy/działania/metody dydaktyczne:

Ćwiczenia z prezentacją multimedialną; przygotowanie i udział w symulacjach biznesowych - prowadzenie wirtualnej firmy

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Wiedzę sprawdza praca pisemna w formie kolokwium - 30 pytań zamkniętych. Przykładowe pytanie: Filary gospodarki rynkowej
W_02	Wiedzę sprawdza praca pisemna w formie kolokwium - 30 pytań zamkniętych. Przykładowe pytanie: Procedura rejestracji firmy w CEIDG
U_01	Umiejętność sprawdzana jest poprzez ocenę wyniku osiągniętego w prowadzeniu firmy w środowisku wirtualnym - branżowe symulacje biznesowe oraz w przygotowaniu projektu grupowego. Przykładowe pytanie: Czynniki konkurencyjności

U_02	Umiejętność sprawdzana jest poprzez ocenę wyniku osiągniętego w prowadzeniu firmy w środowisku wirtualnym - branżowe symulacje biznesowe oraz w przygotowaniu projektu grupowego. Przykładowe pytanie: Najlepsza forma działalności gospodarczej osoby fizycznej
U_03	Umiejętność sprawdzana jest poprzez ocenę wyniku osiągniętego w prowadzeniu firmy w środowisku wirtualnym - branżowe symulacje biznesowe oraz w przygotowaniu projektu grupowego. Przykładowe pytanie: Określenie własnej roli w zespole projektowym
K_01	Kompetencje weryfikowane są poprzez aktywność na ćwiczeniach: zaangażowanie, zadawanie pytań, własna znajomość zagadnień z nauczanego przedmiotu, kreatywność Przykładowe pytanie: Przyczyny utraconej sprzedaży
K_02	Kompetencje weryfikowane są poprzez aktywność na ćwiczeniach: zaangażowanie, zadawanie pytań, własna znajomość zagadnień z nauczanego przedmiotu, kreatywność Przykładowe pytanie: Która z decyzji miała największy wpływ na wynik finansowy

Forma i warunki zaliczenia:

Ćwiczenia - zaliczenie z oceną. Na zaliczenie przedmiotu składa się uczestnictwo na ćwiczeniach oraz uzyskanie co najmniej oceny dostatecznej (2,51). Wiedzę sprawdza praca pisemna. Czas pisania odpowiedzi - 30 minut. Liczba pytań zamkniętych wynosi 30. Umiejętności i kompetencje sprawdza wynik ekonomiczny i finansowy osiągnięty w prowadzeniu wirtualnej firmy, za który można uzyskać 10 pt,; Również 10 pt student uzyskuje za przygotowanie projektu grupowego. Można też uzyskać 5 pt. za aktywność.
Na końcową ocenę składa się suma punktów uzyskana z testu, wyniku symulacji biznesowych i projektu.
Uzyskać można maksymalnie 55 pt. (30 pt. test, 10 pt. - prowadzenie wirtualnej firmy, projekt 10 pt., 5 pt. aktywność)

Kryterium oceny dla zaliczenia:

- 0-24 pkt – ocena ndst,
- 25-30 pkt – ocena dst,
- 31-36 pkt – ocena dst plus,
- 37-42 pkt – ocena db,
- 43-48 pkt – ocena db plus,
- >48 pkt – ocena bdb.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w ćwiczeniach, kolokwium	30 godzin
Udział w konsultacjach	2 godziny
Czytanie zadanych wybranych fragmentów literatury	5 godzin
Przygotowanie i prowadzenie wirtualnej firmy	5 godzin
Przygotowanie projektu grupowego	5 godzin
Przygotowanie do testu	3 godzin
Sumaryczne obciążenie pracą studenta	50 godzin
Punkty ECTS za przedmiot	2 ECTS

Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w ćwiczeniach, kolokwium	15 godzin
Udział w konsultacjach	2 godziny
Czytanie zadanych wybranych fragmentów literatury	10 godzin
Przygotowanie i prowadzenie wirtualnej firmy	10 godzin
Przygotowanie projektu grupowego	10 godzin
Przygotowanie do testu	3 godzin
Sumaryczne obciążenie pracą studenta	50 godzin
Punkty ECTS za przedmiot	2 ECTS

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Bezpieczeństwo systemów komputerowych**
Nazwa w języku angielskim:	Computer security
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:	trzeci
Semestr:	piąty
Liczba punktów ECTS:	3
Imię i nazwisko koordynatora przedmiotu:	dr Piotr Świtalski
Imię i nazwisko prowadzących zajęcia:	dr Piotr Świtalski, mgr inż. Dominik Filipczuk
Założenia i cele przedmiotu:	Założeniem tego przedmiotu jest znajomość przez słuchaczy zagadnień związanych z podstawami architektury komputerów, systemami operacyjnymi oraz sieciami komputerowymi. Wskazana jest również wiedza dotycząca protokołów warstwy aplikacji w stosie sieciowym. Celem przedmiotu jest zaznajomienie studentów z problematyką bezpieczeństwa systemów komputerowych, m.in.: ● podstawami szyfrów oraz zasadami tworzenia i weryfikacji podpisu cyfrowego, ● znanymi atakami wymierzonymi w systemy komputerowe, ● metodami przeciwdziałania atakom i zabezpieczania systemów komputerowych.

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	podstawowe techniki szyfrowania danych oraz zagadnienia z zakresu tworzenia i weryfikacji podpisu cyfrowego	**K_W05**
W_02	mechanizmy uwierzytelniania użytkownika w systemach komputerowych	**K_W05, K_W07**
W_03	metody przeprowadzania ataków wymierzonych w systemy komputerowe	**K_W05, K_W11**
W_04	zasady zabezpieczania sieci komputerowych oraz wykrywania anomalii w tych sieciach	**K_W07**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	sprawnie wyszukiwać w dokumentacji producenta istotne parametry i tryby pracy aplikacji służącej do zabezpieczenia systemu	**K_U01**
U_02	sprawnie wykorzystać narzędzia bezpieczeństwa systemów komputerowych	**K_U15, K_U17**
U_03	dobrać adekwatne zabezpieczenia w stosunku do zagrożenia w systemie komputerowym	**K_U15, K_U17, K_U25**

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 uprzednie zaliczenie następujących przedmiotów: „Architektura systemów komputerowych", „Systemy operacyjne", „Podstawy technologii WWW", „Sieci komputerowe" lub znajomość literatury obowiązującej w tych przedmiotach.

Treści modułu kształcenia:

Treści dotyczące wykładu:

1. **Koncepcja bezpieczeństwa komputerowego.** Podstawowe zasady. Właściwości i klasyfikacja koncepcji bezpieczeństwa: integralność, dostępność, poufność, niezaprzeczalność, odpowiedzialność. Podstawowe pojęcia. Model bezpieczeństwa sieciowego. Minimalne standardy bezpieczeństwa. Polityka bezpieczeństwa.
2. **Ataki w systemach komputerowych.** Specyfika systemów informatycznych. Klasyfikacja zagrożeń. Szkodliwe oprogramowanie. Sieci botnet. Ataki na współczesne procesory. Statystyka typowych zagrożeń. Przestępstwa ujawnione i nieujawnione. Obiekty, typy i sprawcy przestępstw. Piractwo komputerowe, sabotaż, wywiad gospodarczy, szpiegostwo, przestępstwa bankowe. Inżynieria społeczna. Phising. Inżynieria odwrotna. Organizacje przeciwdziałające przestępczości komputerowej.
3. **Szkodliwe oprogramowanie.** Problemy związane z nieautoryzowanym dostępem do systemów komputerowych. Infekcje systemów komputerowych. Bezpieczny system operacyjny. Podatność systemów operacyjnych. Charakterystyka szkodliwego oprogramowania: tylne drzwi, bomby logiczne, konie trojańskie, wirusy, robaki, eksploity i rootkity, keyloggery. Przeciwdziałanie szkodliwemu oprogramowaniu.
4. **Klasyczne techniki szyfrowania.** Dziedzina kryptografii i podstawowe pojęcia. Podstawowe techniki szyfrowania: technika podstawieniowa, szyfr Cezara, szyfry mono i polialfabetyczne, szyfr Playfaira, szyfr Vigenère'a. Techniki przestawieniowe, szyfr zygzakowy, maszyny wirnikowe. Szyfrowanie symetryczne. Ataki siłowe przeprowadzane na algorytmy szyfrowania.
5. **Współczesne szyfry komputerowe.** Szyfry strumieniowe. Szyfr strumieniowy RC4. Szyfry blokowe. Struktura i szyfr Feistela. Standard DES, efekt lawiny. Algorytm AES. Tryby operacyjne szyfrów blokowych. Szyfrowanie asymetryczne. Algorytm RSA.
6. **Podpis cyfrowy.** Idea podpisu cyfrowego. Wymagania stawiane podpisom cyfrowym. Mechanizm uwierzytelniania komunikatów. Kody uwierzytelnienia komunikatów MAC. Podpis cyfrowy ElGamal. Standard DSS. Algorytm DSA. Kryptograficzne funkcje haszujące. Algorytm SHA-512. Paradoks urodzin.
7. **Uwierzytelnianie.** Pojęcie uwierzytelniania. Uwierzytelnianie przez hasło. Strategie wyboru haseł. Inne metody uwierzytelniania. Protokoły uwierzytelniania: protokół challenge and response. Atak „człowiek pośrodku". Dowód z wiedzą zerową. Uwierzytelnianie wieloskładnikowe. Hasło jednorazowe. Generowanie haseł jednorazowych – protokół S/KEY.
8. **Kontrola dostępu.** Zasady kontroli dostępu. Podmiot, obiekt i prawa dostępu. Kontrola dostępu uznaniowa (DAC). Kontrola dostępu bazująca na rolach (RBAC). Kontrola dostępu bazująca na atrybutach (ABAC). Przykład kontroli dostępu: SELinux.

9. **Protokoły i standardy bezpieczeństwa w Internecie.** Bezpieczeństwo poczty elektronicznej. S/MIME. Protokoły SSL/TLS. Ataki na protokoły TLS. Protokół HTTPS. Nagłówki w protokole HTTP. Architektura IPsec.
10. **Zapory sieciowe (firewalle).** Model ogólny zapory sieciowej. Charakterystyka firewalli. Ograniczenia firewalli. Firewall filtrujący pakiety. Firewall filtrujący pakiety z badaniem stanu pakietu. Brama aplikacyjna, brama transmisyjna. Implementacja firewalla. Strefa zdemilitaryzowana (DMZ). Przykładowa konfiguracja firewalla z DMZ.
11. **Systemy wykrywania intruzów.** Zachowania intruzywne. Wzorce zachowań intruzów. Wykrywanie intruzów. Statystyczna analiza zachowania. Wykrywanie intruzów w oparciu o reguły. Systemy IDPS. Audyt w systemach IDPS. Pułapki (Honeypoty).
12. **Bezpieczeństwo aplikacji internetowych cz. 1.** Ataki w warstwie aplikacji. Niewłaściwa kontrola dostępu do aplikacji. Błędy kryptograficzne. Wstrzykiwanie kodu. Ataki SQL Injection. Atak XSS. Atak CSRF.
13. **Bezpieczeństwo aplikacji internetowych cz. 2.** Niebezpieczne projektowanie. Błędy w konfiguracji oprogramowania. Błędy w uwierzytelnianiu użytkownika i zarządzania jego sesją. Niewłaściwe zabezpieczenie wrażliwych danych. Awarie oprogramowania. Niewystarczające logowanie i monitorowanie aplikacji. Atak SSRF.
14. **Bezpieczeństwo systemów mobilnych.** Model izolowania procesów. Piaskownica. Uprawnienia Androida. Android Package. Manifest aplikacji i integralność pakietu. Weryfikacja manifestu. Zagrożenia i podatności w aplikacjach mobilnych. Ewolucja złośliwego oprogramowania w systemach mobilnych. Ataki na sprzęt. Bezpieczeństwo urządzeń mobilnych. Technologie zarządzania urządzeniami mobilnymi.
15. **Zapewnianie dostępności danych.** Utrzymanie ciągłości zasilania. Sposoby zapobiegania problemom zasilania, zasilacze awaryjne UPS. Ochrona danych przed utratą. Systemy macierzowe RAID. Kopie bezpieczeństwa.

Treści dotyczące laboratorium:

1. **Podsłuchiwanie ruchu sieciowego.** Podsłuchiwanie ruchu ICMP, przechwytywanie danych połączenia telnet, analiza pakietów dla ruchu SSH.
2. **Bezpieczeństwo w systemach operacyjnych cz. 1**. Zarządzanie kontami i grupami użytkowników w systemie Windows. Inspekcja. Rejestr systemu Windows. Wykrywanie potencjalnych zagrożeń.
3. **Bezpieczeństwo w systemach operacyjnych cz. 2**. Zarządzanie użytkownikami systemu Linuks. Postarzanie haseł. Polecenia sudo i su. Bity SUID i GUID. Aktualizacja systemu Linuks.
4. **Eksploitowanie systemu Windows**. Konfiguracja środowiska testowego dla Metasploita. Wykorzystanie eksploita. Instalacja backdoora i keyloggera.
5. **Szyfrowanie danych cz. 1 - szyfry i ich łamanie**. Entropia. Szyfr Cezara. Szyfr przestawieniowy.
6. **Szyfrowanie danych cz. 2 – szyfry asymetryczne i podpis cyfrowy**. Generowanie kluczy: prywatnego i publicznego. Tworzenie certyfikatu CA. Tworzenie i podpisanie certyfikatu użytkownika. Konfiguracja serwera HTTP dla bezpiecznej wymiany danych przez SSH.
7. **Funkcje haszujące**. Długość generowanych wyników funkcji haszujących. Badanie własności funkcji haszujących. Solenie. Łamanie haszy za pomocą narzędzia hashkiller.io. Szybkość obliczania haszy. Funkcja HMAC.

8. **Łamanie haseł.** Sposoby przechowywania haseł kont użytkowników w SO. Programy wykorzystywane do łamania haseł: John the Ripper i hashcat. Łamanie haseł systemu Windows. Łamanie haseł systemu Linuks. Łamanie haszy w programie hashcat.
9. **Atak Man in the Middle (MitM).** Konstrukcja ataku. Atak MitM wymierzony w użytkownika przeglądarki internetowej. Przeprowadzenie ataku MitM w środowisku kontenerów.
10. **SELinux.** Wprowadzenie do SELinux. Narzędzia SELinux. Polityka SELinux. Kontekst bezpieczeństwa.
11. **Zapora sieciowa**. Zasady tworzenia reguł dla firewalla iptables. Przykłady reguł iptables. Konfiguracja zapory sieciowej. Filtrowanie pakietów przychodzących. Filtrowanie według adresu IP oraz MAC. Logowanie odrzuconych pakietów.
12. **System wykrywania intruzów**. Przygotowanie środowiska testowego. Tryby pracy Snorta. Tworzenie reguł użytkownika. Tworzenie reguł związanych z atakami.
13. **Ataki na aplikacje internetowe cz. 1**. Przygotowanie środowiska testowego. Atak trwały XSS. Atak odbity XSS. Atak XSS bazujący na obiektach DOM. Atak Cross Site Request Forgery (CSRF). Atak SQL Injection.
14. **Ataki na aplikacje internetowe cz. 2**. Atak Blind SQL Injection. Podatność na ataki związane z przesyłaniem plików. Powłoki wielofunkcyjne. Powłoka odwrotna. Złośliwe pliki SVG. Obejście zabezpieczenia na podstawie rozszerzeń. Obejście zabezpieczenia na podstawie typu MIME.
15. **Kopie bezpieczeństwa.** Program rsync. Kopia przyrostowa w rsync. Automatyzacja procesu tworzenia kopii bezpieczeństwa.

Literatura podstawowa:

1. Stallings W., Brown L.: Bezpieczeństwo systemów informatycznych. Zasady i praktyka, Wydanie IV. Tom 1, Wyd. Helion, Gliwice, 2023.
2. Khawaja G.: Kali Linux i testy penetracyjne. Biblia. Wyd. Helion, Gliwice, 2022.

Literatura dodatkowa:

1. Stallings W., Brown L.: Bezpieczeństwo systemów informatycznych. Zasady i praktyka, Wydanie IV. Tom 2, Wyd. Helion, Gliwice, 2023.
2. Janca T.: Alicja i Bob. Bezpieczeństwo aplikacji w praktyce. Wyd. Helion, Gliwice, 2021.
3. Tevault D. A.: Bezpieczeństwo systemu Linux. Hardening i najnowsze techniki zabezpieczania przed cyberatakami, Wyd. Helion, Gliwice, 2024.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany jest technikami multimedialnymi. Ćwiczenia laboratoryjne – zajęcia praktyczne z wykorzystaniem wybranych narzędzi programowych. Na stronie internetowej prowadzącego zamieszczane są materiały z problemami i zadaniami laboratoryjnymi.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „Jak weryfikowany jest podpis cyfrowy?”.
W_02	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „Jakiego rodzaju składnik materialny może być użyty w uwierzytelnianiu wieloskładnikowym (MFA)?”.

W_03	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „W jaki sposób przeprowadzany jest atak DDoS?”.
W_04	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „Przedstaw statystyczną analizę zachowania w systemach IDPS”.
U_01	Efekt realizowany w trakcie zajęć laboratoryjnych. Sprawdzane będą umiejętności korzystania z dokumentacji danego rozwiązania. Przykładowe zadanie: „Sprawdź w dokumentacji programu selinux jak można zweryfikować prawidłowość kontekstu danego pliku względem jego położenia w systemie plików”.
U_02	Efekt realizowany w trakcie zajęć laboratoryjnych. Sprawdzane będą umiejętności konfiguracji zabezpieczeń w systemie komputerowym. Przykładowe zadanie: „Wygeneruj i zaimplementuj certyfikat SSL w wybranym serwerze HTTP”.
U_03	Efekt realizowany w trakcie zajęć laboratoryjnych. Student będzie realizował zadanie związane z instalacją i konfiguracją określonego zabezpieczenia. Przykładowe zadanie: „Zabezpiecz usługi systemu operacyjnego Linuks przy pomocy wybranej zapory sieciowej.

Forma i warunki zaliczenia:

Moduł kończy się egzaminem. Ocena z przedmiotu składa się z dwóch ocen cząstkowych:

- oceny z zajęć laboratoryjnych,
- oceny z egzaminu końcowego.

Na ocenę z zajęć laboratoryjnych składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim, za które można uzyskać sumarycznie 50 pkt. Zaliczenie zajęć laboratoryjnych możliwe po uzyskaniu co najmniej 51% liczby punktów z tej formy zaliczenia

Egzamin jest egzaminem pisemnym. Do egzaminu mogą przystąpić osoby, które uzyskały zaliczenie laboratorium. Na egzaminie można uzyskać maksymalnie 50 pkt. Egzamin będzie zaliczony w przypadku uzyskania co najmniej 51% liczby punktów z tej formy zaliczenia.

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.
Przygotowanie się do egzaminu	6 godz.
Obecność na egzaminie	1 godz.
Przygotowanie się do ćwiczeń laboratoryjnych	6 godz.

Udział w konsultacjach z przedmiotu	2 godz.
Sumaryczne obciążenie pracą studenta	**75 godz.**
Punkty ECTS za przedmiot	**3 ECTS**
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Przygotowanie się do egzaminu	21 godz.
Obecność na egzaminie	1 godz.
Przygotowanie się do ćwiczeń laboratoryjnych	21 godz.
Udział w konsultacjach z przedmiotu	2 godz.
Sumaryczne obciążenie pracą studenta	**75 godz.**
Punkty ECTS za przedmiot	**3 ECTS**

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Platformy programowania**
Nazwa w języku angielskim:	Programing platforms
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:	trzeci
Semestr:	piąty
Liczba punktów ECTS:	**3**
Imię i nazwisko koordynatora przedmiotu:	dr Grzegorz Terlikowski
Imię i nazwisko prowadzących zajęcia:	dr Grzegorz Terlikowski
Założenia i cele przedmiotu:	Zakłada się, że student posiada umiejętności dotyczące programowania obiektowego i znajomość technologii HTML/JS/CSS. Celem przedmiotu jest nabycie przez studentów podstawowej wiedzy i umiejętności dotyczących wytwarzania oprogramowania w ekosystemie Java, ze szczególnym uwzględnieniem platformy Spring i środowiska IntelliJ IDEA Ultimate.

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	trójwarstwowy model aplikacji webowej; Potrafi wymienić i opisać sposoby dostępu do danych stosowane na platformie Spring; Zna najważniejsze technologie dostępu do danych platformy Jakarta EE i Spring takie jak: JDBC, JTA, Hibernate,	K_W05, K_W06
W_02	zagadnienia związane z paradygmatem MVC i REST API. Zna i rozumie funkcjonowanie poszczególnych elementów aplikacji zgodnej z tym paradygmatem;	K_W05, K_W06
W_03	strukturę i mechanizmy działania aplikacji budowanej przy pomocy frameworka Spring takie jak wstrzykiwanie zależności.	K_W05, K_W06
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	posługiwać się w sposób efektywny środowiskiem programistycznym InteliJ Idea Ultimate oraz zaimplementować aplikację webową w Spring Boot.	K_U01, K_U07, K_U12
U_02	realizować dostęp do danych zapisanych w bazie danych oraz je filtrować i walidować.	K_U01, K_U07, K_U12
U_03	zaimplementować warstwę bezpieczeństwa z wykorzystaniem Spring Security	K_U01, K_U07, K_U12
U_04	opracować i zaimplementować aplikację o strukturze zgodnej z paradygmatem MVC i CRUD przy pomocy frameworka Spring MVC. Potrafi tworzyć interfejsy RESTfull.	K_U01, K_U07, K_U12
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	

K_01	podejmowania decyzji i krytycznej oceny własnych rozwiązań w rozwiązywaniu zadań projektowych z zakresu tworzenia aplikacji webowych.	K_K01, K_K03
K_02	uznania znaczenia wiedzy w rozwiązywaniu zadań projektowych z zakresu aplikacji webowych w architekturze trójwarstwowej.	K_K01, K_K03

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:

Znajomość języka programowania Java oraz podstawy programowania obiektowego.

Treści modułu kształcenia:

Wykłady:

1. **Wprowadzenie do Jakarta Enterprise Edition**. Historia, przegląd modułów, najważniejsze funkcjonalności.
2. **Wprowadzenie do Spring Framework, Spring Boot i Spring MVC**. Historia, przegląd modułów i możliwości Spring Framework/Spring Boot.
3. **Spring MVC a programowanie reaktywne w Web Flux**. Omówienie paradygmatu MVC. Przepływ sterowania. Kontrolery i mapowania żądań.
4. **Obsługa formularzy i walidacja danych**. Integracja aplikacji z Bean Validation API. Walidatory w Spring Framework. Przepływ sterowania.
5. **Spring IoC**. Kontener odwrócenia kontroli. Wstrzykiwanie zależności. Spring Beans. Konfiguracje.
6. **Dostęp do danych w Spring**. Wzorce DAO i Repozytorium. Spring JDBC, JPA, Encje, Mapowanie obiektowo-relacyjne. Obsługa transakcji.
7. **Dostęp do danych w Spring – Spring Data**. Hierarchia interfejsów. Spring Data JPA. JPQL. Sposoby tworzenia zapytań o dane.
8. **Dostęp do danych w Spring – Spring Data JPA**. Wzorzec otwartej sesji w widoku. System pamięci podręcznej. Zagadnienie audytu i sposoby jego rozwiązania.
9. **Wprowadzenie do REST**. Formaty wiadomości, mapowania żądań, obsługa sytuacji wyjątkowych. Spring Data REST.
10. **Bezpieczeństwo aplikacji z wykorzystaniem Spring Security**. Elementy architektury. Przepływ sterowania związany z uwierzytelnianiem i autoryzacją. Możliwości frameworka i możliwości integracyjne.
11. **Kolokwium zaliczeniowe**.

Ćwiczenia:

1. **Wprowadzenie do frameworków Spring Boot, Spring MVC i Thymeleaf**. Pierwszy projekt aplikacji webowej w oparciu o paradygmat MVC w IntelliJ Ultimate.
2. **Komponenty Java Beans oraz wyrażenia i atrybuty Thymeleaf**. Dostęp do zmiennych kontekstowych (m.in, komponentów Java Beans) z wykorzystaniem atrybutów oraz wyrażeń Thymeleaf.
3. **Zaawansowane mapowanie żądań HTTP.** Realizacja CRUD oraz obsługa prostych formularzy.
4. **Formatowanie, walidacja i niestandardowe wiązanie danych**. Formatowanie danych z użyciem adnotacji oraz edytorów właściwości i formaterów danych. Wykrywanie błędów w formularzu i ich wyświetlanie. Błędy związane z wiązaniem danych. Wykorzystanie Bean Validation API do walidacji danych.
5. **Wstęp do Spring Data i wstrzykiwania zależności**. Konfiguracja źródła danych i silnika ORM (Object-Ralational Mapping). Opisywanie modelu dziedziny z wykorzystaniem adnotacji JPA. Spring Data JPA – repozytoria JPA. Wstrzykiwanie zależności z wykorzystaniem adnotacji @Autowired i @Bean.

6. **Relacja Wiele-Do-Wielu i wstęp do własnych zapytań o dane**. Projekt Lombok. Rozbudowa modelu dziedziny – relacja Wiele-Do-Wielu. Wstęp do tworzenia własnych zapytań o dane.
7. **Zaawansowane metody tworzenia zapytań o dane**. Zapytania nazwane. Zapytania z użyciem adnotacji @Query. Używanie parametrów nazwanych. Wyrażenia SpEL. Wykorzystanie Stream API. Wykorzystanie grafów encji. JPA Criteria API. Wzorzec otwartej sesji w widoku.
8. **Bezpieczeństwo aplikacji z użyciem Spring Security**. Podstawowa konfiguracja silnika Spring Security. Konfiguracja HTTPSecurity i własny formularz logowania. Obsługa braku dostępu do zasobów. Audyt zdarzeń. Atrybuty i wyrażenia Thymeleaf do obsługi Spring Security.
9. **Warstwa usług, profile konfiguracyjne, zarządzanie transakcjami**. Adnotacje @Component i @Service. Warstwa usług na przykładzie usługi dostarczającej informacji o użytkowniku. Zarządzanie profilami i transakcjami - adnotacja @Transactional. Obsługa sytuacji wyjątkowych.
10. **Przykładowe usługi**. Internacjonalizacja aplikacji. Wysyłanie maili i formatowanie ich treści. Przesyłanie plików na serwer i zapis do bazy i na dysk.
11. **Rest API**. Kontrolery restowe, obsługa żądań, typy mediów, statusy odpowiedzi. DTO (Data Transfer Object) – transformacja międzywarstwowa. Narzędzie Swagger do dokumentowania i testowania Rest API.
12. **Obrona projektów indywidualnych**. Testowanie i weryfikacja poprawności realizacji projektu indywidualnego. Możliwe niewielkie modyfikacje oprogramowania, w celu weryfikacji samodzielności jego wykonania.

Literatura podstawowa:

1. Peter Späth. ***Pro Jakarta EE 10***. 2023.

2. Rheinwerk Publishing, Inc, Christian Ullenboom. ***Spring Boot 3 and Spring Framework 6.*** Helion 2023.

Literatura dodatkowa:

1. Sourabh Sharma. **Modern API Development with Spring 6 and Spring Boot 3**. Design scalable, viable, and reactive APIs with REST, gRPC, and GraphQL using Java 17 and Spring Boot 3 - Second Edition (ebook). 2023.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, ćwiczenia wspomagane sprzętem laboratoryjnym. Zamieszczanie na stronie internetowej zadań ćwiczeniowych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Sprawdzane podczas pisemnego zaliczenia wykładu na ocenę. Przykładowe pytania: 1. Czym jest Encja? Przedstaw cykl życia encji (diagram stanów) w standardzie JPA. Wymień i scharakteryzuj poziomy caschowania danych w silniku Hibernate.
W_02	Sprawdzane podczas pisemnego zaliczenia wykładu na ocenę. Przykładowe pytania: 1. Opisz krok po kroku jak jest obsługiwane żądanie HTTP przez silnik Spring MVC. Jak Spring Boot ułatwia mapowanie żądań HTTP na metody kontrolera? Wyjaśnij na przykładzie działania adnotacji takich jak @GetMapping, @PostMapping, itp.
W_03	Sprawdzane podczas pisemnego zaliczenia wykładu na ocenę. Przykładowe pytania: 1. Co to jest kontener IoC? Wymień jego wady i zalety. Wymień rodzaje i podaj przykłady wstrzykiwania zależności na platformie Spring.

U_01	Systematycznie sprawdzane na zajęciach laboratoryjnych. Materiały na następne laboratorium muszą być dostępne co najmniej tydzień przed zajęciami. Student, na podstawie podanej literatury, musi się do nich samodzielnie, lub korzystając z konsultacji, przygotować. Przykładowe zadanie: W środowisku InteliJ IDEA Ultimate załóż projekt typu Spring Boot, wskazując odpowiednie zależności (biblioteki). Opracuje encję Book, a następnie wygeneruj dla niej metody dostępowe (gettery i settery) z wykorzystaniem odpowiednich opcji tego środowiska.
U_02	Systematycznie sprawdzane na zajęciach laboratoryjnych. Przykładowe zadanie: Opracuj encję Book i połącz ją relacją ManyToOne z encją CoverType oraz relacją ManyToMany z Category. Odwzoruj je z wykorzystaniem adnotacji JPA. Dodaj odpowiednie adnotacje z Bean Validation API, w celu walidacji danych.
U_03	Systematycznie sprawdzane na zajęciach laboratoryjnych. Przykładowe zadanie: Dodaj do projektu zależność Spring Security, uruchom aplikację i zaloguj się do aplikacji na dane wyświetlone na konsoli.
U_04	Systematycznie sprawdzane na zajęciach laboratoryjnych. Przykładowe zadanie: Dla encji Book opracuj opracuje interfejsy RESTful realizujące CRUD.
K_01	Weryfikowany, w oparciu o posiadaną wiedzę i umiejętności w czasie zajęć laboratoryjnych, podczas zaliczania zadania indywidualnego, a także będą sprawdzane na egzaminie.
K_02	Weryfikowany, w oparciu o posiadaną wiedzę i umiejętności w czasie zajęć laboratoryjnych, podczas zaliczania zadania indywidualnego.

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem na ocenę. Ocena końcowa jest wystawiana na podstawie zajęć laboratoryjnych i jednego kolokwium pisemnego przeprowadzonego na ostatnim wykładzie. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim oraz z samodzielnie wykonanego zadania indywidualnego według schematu:

- Regularne zajęcia – 35 pkt.,
- Obrona zadania indywidualnego – 15 pkt.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej połowy punktów z poszczególnych form aktywności studenta: regularne zajęcia – co najmniej 16 pkt., obrona indywidualnego zadania – co najmniej 8 pkt. Na tej formie zajęć student może maksymalnie uzyskać 50 pkt.

Kolokwium ma formę pisemną. Można na nim uzyskać do 50 pkt. Egzamin będzie zaliczony w przypadku uzyskania co najmniej 26 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).

Poprawy:

- Poprawa dwóch ćwiczeń w trakcie zajęć w semestrze.
- Jednorazowa poprawa projektu indywidualnego przed sesją egzaminacyjną.
- Jednorazowa poprawa kolokwium przed sesją egzaminacyjną.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Samodzielne przygotowanie się do zajęć laboratoryjnych	12 godz.
Praca nad projektem indywidualnym.	8 godz.
Udział w konsultacjach godz. z przedmiotu	2 godz.
Przygotowanie się do kolokwium	8 godz.
Sumaryczne obciążenie pracą studenta	**75 godz.**

Studia niestacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do zajęć laboratoryjnych	20 godz.
Praca nad projektem indywidualnym.	13 godz.
Udział w konsultacjach godz. z przedmiotu	2 godz.
Przygotowanie się do kolokwium	10 godz.
Sumaryczne obciążenie pracą studenta	**75 godz.**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Sztuczna inteligencja**	
Nazwa w języku angielskim:	Artificial Intelligence	
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):		pierwszego stopnia
Rok studiów:	trzeci	
Semestr:	piąty	
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 mgr inż. Dariusz Ruciński	
Założenia i cele przedmiotu:	Zakłada się, że student ma wiedzę z matematyki oraz algorytmów i złożoności. Zna również podstawy programowania deklaratywnego, w tym języka Prolog oraz programowania w językach sztucznej inteligencji, jak m.in. j. Python oraz j. C#. Celem zajęć jest zapoznanie studentów ze sztuczną inteligencją, w tym zwłaszcza podstawowymi metodami sztucznej inteligencji: systemami ekspertowymi, sztucznymi sieciami neuronowymi, algorytmami ewolucyjnymi, sztucznymi systemami immunologicznymi i innymi stosowanymi do programowania, i identyfikacji systemów sztucznej inteligencji. Także celem zajęć jest poznanie przez studentów środowiska MATLAB i Simulink w tym wybranych toolbox-ów oraz j. Matlab w zakresie wspomagania implementacji metod i systemów sztucznej inteligencji.	

Symbol efektu	**Efekty uczenia się** **WIEDZA** **Student zna i rozumie:**	**Symbol efektu kierunkowego**
W_01	metody tworzenia bazy wiedzy, w tym w szczególności z wykorzystaniem drzewa celów oraz zna metody wnioskowania wykorzystywane w maszynach wnioskujących, jak np. wnioskowanie do przodu, wnioskowanie do tyłu, wnioskowanie mieszane, itp. i rozumie podstawowe zagadnienia modelowania analitycznego, neuralnego i identyfikacyjnego systemów, procesów, obiektów,	**K_W10**

	itp. na poziomie niezbędnym do otrzymywania złożonych modeli w środowisku MATLAB-a i Simulink-a.	
W_02	metody projektowania systemów ekspertowych, sztucznych sieci neuronowych oraz systemów ewolucyjnych i rozumie podstawowe zagadnienia z zakresu sztucznej inteligencji, w tym zna i rozumie jej podstawowe metody: systemy ekspertowe, sztuczne sieci neuronowe, algorytmy ewolucyjne, algorytmy immunologiczne, algorytmy mrówkowe, systemy rozmyte, algorytmy kwantowe, metody reprezentacji wiedzy, metody akwizycji wiedzy, metody wnioskowania, itp.	**K_W10**
W_03	funkcjonowanie takich środowisk programistycznych wspomagających tworzenie systemów sztucznej inteligencji jak MATLAB i Simulink (w tym język Matlab, Simulink oraz takie toolboxa-y jak m.in.: System Identification Toolbox, Deep Learning Toolbox, Fuzzy Logic Toolbox, Control System Toolbox, Optimization Toolbox, Global Optimization Toolbox, itp.) oraz rozumie zagadnienia z zakresu metodyki i technik programowania w językach bardzo wysokiego poziomu takich jak język Matlab oraz j. Python.	**K_W10**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	projektować i przeprowadzać eksperymenty badawcze systemów rzeczywistych z wykorzystaniem środowiska MATLAB i Simulink, w tym potrafi pozyskać dane do eksperymentu, przygotować eksperyment praktyczny z wykorzystaniem środków informatycznych, przeprowadzić eksperyment oraz zinterpretować uzyskane wyniki i wyciągać wnioski.	**K_U07**
U_02	opracować dokumentację dotyczącą realizacji zadania inżynierskiego i przygotować tekst zawierający omówienie wyników realizacji tego zadania, w tym dokumentację z realizowanych laboratoriów i instrukcję użytkowania opracowanego własnego programu, potrafi przygotować dane do identyfikacji i przeprowadzić identyfikację prowadzącą do uzyskania modelu systemu, potrafi przygotować plik uczący i zaprojektować Sztuczną Sieć Neuronową, potrafi przygotować populację początkową i zaprogramować Algorytm Ewolucyjny, potrafi przygotować model symulacyjny systemu rzeczywistego i zaprogramować model symulacyjny w Simulinku.	**K_U05**
U_03	Przy identyfikowaniu i formułowaniu specyfikacji zadań inżynierskich oraz przy ich rozwiązywaniu potrafi wykorzystywać metody analityczne, identyfikacyjne, neuronalne, ewolucyjne oraz symulacyjne i eksperymentalne, dostrzegać ich aspekty systemowe i pozatechniczne oraz dokonać wstępnej oceny ekonomicznej proponowanych rozwiązań i podejmowanych działań inżynierskich oraz potrafi wykorzystać środowisko MATLABA i Simulinka, w tym: j. Matlab, Simulink oraz toolbox-y takie jak m.in.: System Identification Toolbox, Deep Learning Toolbox, Fuzzy Logic Toolbox, Control System Toolbox, Optimization Toolbox, Global Optimization Toolbox, Simulink, itp. do projektowania, testowania i symulacji złożonych systemów sztucznej inteligencji.	**K_U06**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	podejmowania decyzji, krytycznej oceny działań własnych i studentów z grupy laboratoryjnej, w której uczestniczy. Jest gotowy do przyjmowania odpowiedzialności za skutki własnej pracy wykonywanej na zajęciach laboratoryjnych.	**K_K02**
K_02	uznania znaczenia wiedzy w rozwiązywaniu problemów oraz jest gotowy do konstruktywnej krytyki w stosunku do działań swoich i innych osób. Ma	**K_K03**

	świadomość znaczenia zachowania się w sposób profesjonalny, konieczności przejawiania inicjatywy oraz przestrzegania zasad etyki zawodowej i inżynierskiej.	
K_03	odpowiedzialnego pełnienia roli zawodowej i społecznej informatyka, w tym do przestrzegania zasad etyki zawodowej i etyki społecznej oraz wymagania tego od innych oraz potrafi dbać o dorobek i tradycję zawodu informatyka.	**K_K04**

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:

1. Wiedza z podstaw logiki matematycznej, rachunku zdań, rachunku predykatów, algorytmów i złożoności.
2. Znajomość podstaw programowania deklaratywnego, w tym języka Prolog oraz programowania w językach sztucznej inteligencji, jak m.in. j. Python oraz j. C#.
3. Umiejętność samodzielnego programowania w dowolnych środowiskach programistycznych.

Treści modułu kształcenia:

Treści dotyczące wykładu:

1. **Wstęp do sztucznej inteligencji.** Inteligencja a sztuczna inteligencja, algorytmika a heurystyka, logika a metalogika, wiedza a baza wiedzy, życie a sztuczne życie, kod genetyczny a kod informacyjny, test Turinga, automatyka, cybernetyka jako sterowanie i komunikacja wewnętrzna w systemie, informatyka a sztuczna inteligencja, metody, języki i narzędzia sztucznej inteligencji, projektowanie i implementacja systemów sztucznej inteligencji, zastosowania systemów sztucznej inteligencji.
2. **Metody reprezentacji wiedzy i systemy ekspertowe.** Projektowanie i programowanie systemów ekspertowych. Metody akwizycji wiedzy. Języki systemów ekspertowych, inżynieria wiedzy i architektura systemów ekspertowych, Zastosowania systemów ekspertowych, systemy ekspertowe czasu rzeczywistego, metody reprezentacji wiedzy: rachunek zdań, stwierdzenia, regułowa reprezentacja wiedzy, rachunek predykatów, sieci semantyczne, reprezentacja wiedzy za pomocą ram, inne metody reprezentacji wiedzy, metody akwizycji wiedzy, moduły objaśniania wiedzy, itp.
3. **Metody wnioskowania.** wnioskowanie w przód, sterowanie wnioskowaniem, wnioskowanie wstecz, wnioskowanie mieszane, wnioskowanie rozmyte, podstawowe zagadnienia uczenia się maszyn, metodyka tworzenia i badania poprawności bazy wiedzy, itp.
4. **Sztuczne sieci neuronowe I.** Idea sztucznych sieci neuronowych, Charakterystyka sztucznego neuronu oraz sztucznej sieci neuronowej, modele neuronu, rodzaje sieci neuronowych, struktura sieci neuronowych, podstawowe metody uczenia sieci, reguły uczenia sieci neuronowych, metody uczenia sztucznych sieci neuronowych (uczenie z nadzorem, bez nadzoru, z krytykiem), reguły uczenia: Reguła Hebba, Reguła Perceptronowa, Reguła Delta, Reguła Widrowa – Hoffa, Reguła WTA i WTM, Reguła gwiazdy wyjść, nowoczesne metody uczenia, parametry uczenia sztucznych sieci neuronowych, itp.
5. **Sztuczne sieci neuronowe II .** Funkcjonowanie Sztucznej Sieci Neuronowej, sieci liniowe i nieliniowe, sieci jednokierunkowe i sieci rekurencyjne, sieci komórkowe, sieci jednowarstwowe i wielowarstwowe, przykłady sztucznych sieci neuronowych: SSN PERCEPTRON, SSN HOPFIELDA, SSN HAMMINGA, SSN Kohonena, sztuczne sieci neuronowe chaotyczne, sztuczne sieci neuronowe ontogeniczne, sztuczne sieci neuronowe dualne, głębokie sztuczne sieci neuronowe, itp.
6. **Sztuczne sieci neuronowe III.** Sztuczne sieci neuronowe w zastosowaniach praktycznych, w tym**: uczenie głębokie, uczenie rozmyte, pamięci asocjacyjne,** Metody uczenia głębokiego, metody kognitywistyczne, systemy uczące się, wnioskowanie wartości funkcji logicznej z przykładów, uczenie drzew decyzyjnych, uczenie Bayesowskie, pamięci asocjacyjne, w tym heteroasocjacyjne, itp.
7. **Algorytmy genetyczne i ewolucyjne I**. Klasyczny algorytm genetyczny, algorytmy ewolucyjne, pojęcie krzepkości algorytmów genetycznych, populacja początkowa, operatory genetyczne krzyżowania i mutacji, metody selekcji, rola funkcji przystosowania, zarządzanie populacją, itp.
8. **Algorytmy genetyczne i ewolucyjne II.** Matematyczne podstawy algorytmów genetycznych**, t**eoria schematów, liczność i długość schematu, przystosowanie schematu, twierdzenie o schematach, hipoteza bloków budujących, zapobieganie przedwczesnej zbieżności, itp., strategie ewolucyjne,

programowanie genetyczne i programowanie ewolucyjne, algorytmy koewolucyjne podpopulacyjne i komórkowe, genetyczne systemy uczące się, modyfikacje algorytmów ewolucyjnych, itp.

9. **Systemy kognitywne.** Inteligencja kognitywna, systemy jako układy pamiętające, język maszyny a dialog człowieka z maszyną, widzenie komputerowe, mimika a naturalne odruchy twarzy humanoidalnej, problematyka umysłu a możliwości modelowania umysłu, sens poznania, rozumienie mowy i obrazu w psychologii i neurobiologii, podejście do systemu kognitywnego w filozofii, informatyce, lingwistyce i antropologii, itp.
10. **Sztuczne systemy immunologiczne.** Detekcja jako dopasowywanie elementów, pamięć immunologiczna, funkcjonowanie i metadynamika układu odpornościowego, typy struktur w systemie, binarny klasyfikator, rola strzelców, samoorganizacja pamięci immunologicznej, pociski (limfocyty wysyłane przez strzelców), system immunologiczny jako system ewolucyjny, itp.
11. **Sztuczne systemy transportowe**, w tym sztuczne systemy rojowe (algorytmy mrówkowe, algorytmy kolonii pszczół, algorytmy ruchu, algorytmy grawitacyjne, itp.).
12. **Metody eksploracji danych i metody analizy skupień,** w tym hierarchiczne i niehierarchiczne metody analizy skupień, miary odległości, metody i techniki eksploracji danych, techniki odkrywania asocjacji, metody klasyfikacji, metody grupowania, itp.
13. **Metody maszynowego uczenia się**, w tym uczenie ze wzmocnieniem, wnioskowanie wartości funkcji logicznej z przykładów, uczenie drzew decyzyjnych, uczenie Bayesowskie, uczenie z przykładów, uczenie się zbioru reguł, analityczne uczenie, połączenie indukcyjnego i analitycznego uczenia, uczenie przez wzmacnianie, itp.
14. **Inspiracje kwantowe metod sztucznej inteligencji**, w tym inspiracje kwantowe sztucznych sieci neuronowych i algorytmów ewolucyjnych, kwantyzacja i dekwantyzacja, obliczenia kwantowe z wykorzystaniem algebry liniowej i rachunku wektorowo-macierzowego, bramki i obwody kwantowe, itp.
15. **Nowe metody sztucznej inteligencji.** Fabryki bezludne, sztuczne życie, statystyczne systemy uczące się, rozproszone oraz równoległe i rozmyte systemy sztucznej inteligencji, itp.

Treści bloków zadań laboratoryjnych:

Zadanie 1: Opracowanie eksperymentu badawczego, pozyskanie danych do badań oraz przeprowadzenie identyfikacji systemu w środowisku MATLABA i Simulinka z wykorzystaniem System Identification Toolbox-a i Control System Toolbox-a e celu uzyskania modelu parametrycznego systemu z elementami sztucznej inteligencji.

Zadanie 2: Przygotowanie pliku uczącego oraz zaprojektowanie Sztucznej Sieci Neuronowej w środowisku MATLAB-a i Simulink-a z wykorzystaniem Deep Learning Toolbox-a lub Neural Network Toolbox-a oraz Fuzzy Logic Tooolbox-a w celu uzyskania modelu neuralnego systemu.

Zadanie 3: Zaprojektowanie Algorytmu Ewolucyjnego na bazie uzyskanego modelu identyfikacyjnego lub modelu neuralnego (utworzenie Populacji Początkowej, opracowanie uzgodnionego z prowadzącym zajęcia algorytmu krzyżowania, algorytmu mutacji, algorytmu selekcji, algorytmu funkcji krzepkości/przystosowania, itp.) oraz implementacja Algorytmu Ewolucyjnego w ustalonym z prowadzącym zajęcia języku programowania systemów sztucznej inteligencji, m.in. w j. Matlab w postaci m-pliku lub z wykorzystaniem System Optimization Toolbox-a (lub Global Optimization Toolbox-a), języku Python i jego bibliotek, języku Sphinx, itp., a także przetestowanie zaimplementowanego Algorytmu Ewolucyjnego.

Zadanie 4: Zaprojektowanie i implementacja w Simulink-u modelu symulacyjnego z elementami systemu sztucznej inteligencji na bazie uzyskanych modeli: parametrycznego, neuralnego, ewolucyjnego, itp. oraz wykorzystanie ich do przeprowadzenia badań symulacyjnych, komparatystycznych i testujących poprawność uzyskanych modeli, a m.in. do ich porównywania między sobą oraz w odniesieniu do systemu rzeczywistego z wykorzystaniem błędu bezwzględnego i błędu względnego.

Zadanie 5: Każdy student otrzymuje do samodzielnego poznania środowisko programistyczne z zakresu sztucznej inteligencji i robotyki humanoidalnej lub mobilnej. Samodzielnie opracowuje przykład, implementuje go z wykorzystaniem uzgodnionego z prowadzącym zajęcia środowiska programowania oraz przygotowuje sprawozdanie z przeprowadzonych badań, a także instrukcję obsługi programu. Możliwymi do wykorzystania są m.in.:

- j. Matlab z dostępnymi bibliotekami takimi jak m.in.: Bioinformatics Toolbox, Control System Toolbox, Deep Learning Toolbox, Fuzzy Logic Toolbox, Global Optimization Toolbox, Image Processing Toolbox,

Mapping Toolbox, Optimization Toolbox, Signal Processing Toolbox, Statistics and Machine Learning Toolbox, Symbolic Math Toolbox, Wavelet Toolbox, itp.

- j. Python z dostępnymi bibliotekami,

Wszyscy studenci samodzielnie poznają podstawy implementacji systemów z wykorzystaniem jednego z ww. języków programowania i ich bibliotek.

Literatura podstawowa:

1. R. Pratap: MATLAB dla naukowców i inżynierów. PWN. Warszawa 2015.
2. L. Rutkowski: Metody i techniki sztucznej inteligencji. PWN. Wyd. II zmienione. Warszawa 2012, 2020.
3. J. Tchórzewski: Metody sztucznej inteligenci i informatyki kwantowej w ujęciu teorii sterowania i systemów. Wydawnictwo Naukowe UPH, Siedlce 2021.
4. R. Tadeusiewicz: Archipelag sztucznej inteligencji. OW EXIT, Warszawa 2021.

Literatura dodatkowa:

1. M. Lawrence: Sztuczna inteligencja i uczenie maszynowe dla programistów. Praktyczny przewodnik po sztucznej inteligencji. HELION, Warszawa 2021.
2. J. Patterson, A. Gipson: Deep Learning. Praktyczne wprowadzenie. Helion. Gliwice 2018.
3. H. de Ponteves: Sztuczna inteligencja. HELION, Warszawa 2021.
4. S. Wierzchoń, M. Kłopotek M: Algorytmy analizy skupień. WNT. Warszawa 2015.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi. Udostępnianie studentom treści wykładów w postaci Print Screen-ów prezentacji przygotowanych w MS Power Point oraz instrukcji do ćwiczeń laboratoryjnych przygotowanych w wersji pdf. Ćwiczenia laboratoryjne realizowane są w 5-ciu blokach tematycznych. Na każdy blok tematyczny składają się trzy ćwiczenia laboratoryjne:

- przygotowanie przez studenta własnego zadania do zaprogramowania w środowisku MATLABA i Simulinka (przygotowanie danych rzeczywistych na zadany przez prowadzącego zajęcia temat, sposobu rozwiązania zadania oraz poznanie środowiska programowania,

- zaprojektowanie własnego zadania w konsultacji z prowadzącym zajęcia z wykorzystaniem środowiska MATLAB i Simulink oraz jego bibliotek,

- opracowanie sprawozdania z wykonanego samodzielnie zadania/instrukcji obsługi programu oraz zaliczenie tematu (praktyczne i teoretyczne).

Dodatkowym zadaniem jest opracowanie projektu indywidualnego poza zajęciami i jego implementacja w wybranym środowisku implementacji systemów sztucznej inteligencji (m.in. w j. Matlab, j. Python, w j. Sphinx, itp. na uzgodniony z prowadzącym zajęcia temat.

Sposoby weryfikacji efektów kształcenia osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01 - W_03	sprawdzane są na czterech zajęciach podsumowujących bloki tematyczne na zajęciach laboratoryjnych sprawdzającym wiedzę praktyczną w odniesieniu do wiedzy teoretycznej. Pytania dotyczyć będą praktycznego użycia wybranych metod sztucznej inteligencji oraz związanych z nimi zadań szczegółowych. **Przykładowe zadania**:

	● Dana jest struktura pliku uczącego. Należy zaprojektować Sztuczną Sieć Neuronową w środowisku MATLABA z wykorzystaniem Deep Learning Toolbox-a do rozpoznawania cyfr rzymskich. ● Dany jest model parametryczny postaci A1(q) $y_1(t)$ = B1(q) $u_1(t)$ + B2(q) $u_2(t)$, gdzie: A1(q) = 1 + 0,2 q^{-1} + 0,3 q^{-2} + 0,4 q^{-3}), B1(q) = 0,2 q^{-1} +0,4 q^{-2}, B2(q) = 0,1 q^{-1} +0,3 q^{-2}. Należy utworzyć Populację Początkową dla potrzeb Algorytmu Ewolucyjnego oraz napisać program w: j. Matlab w postaci m-pliku, j. Python, j. Sphinx, itp. Z implementacją Algorytmu Ewolucyjnego zawierającego jednopunktowe krzyżowanie, mutację poprzez zmianę znaku genu oraz selekcję koła ruletki. ● Dany jest plik uczący, struktura systemu rzeczywistego oraz reguła uczenia. Należy przygotować i przeprowadzić obliczenia uczenia Sztucznej Sieci Neuronowej jako modelu systemu rzeczywistego oraz przedstawić uzyskany model neuronalny złożony z odpowiednich sumatorów i układów odwzorowujących. ● Dany jest model matematyczny systemu, np. w postaci modelu parametrycznego arx. Należy zbudować model symulacyjny w Simulinku. Zaproponować sposób na przeprowadzenie badań symulacyjnych, komparatystycznych oraz badania wrażliwości. ● Scharakteryzuj architektury SSN wykorzystywane do budowy systemów kognitywnych. Omów rolę funkcji aktywacji neuronów. Podaj przykład zapisu ogólnego modelu matematycznego neuronu. Sprawdzenie wiedzy teoretycznej dokona się jednokrotnie na ostatnim wykładzie w postaci testu z jednokrotnym wyborem lub w postaci zadań problemowych.
U_01 - U_03	sprawdzane są cztery razy, to jest przy zaliczaniu każdego zadania ćwiczeń laboratoryjnych
K_01 - K_03	będą weryfikowane, w oparciu o posiadaną wiedzę i umiejętności w czasie zajęć laboratoryjnych, a także podczas zaliczania tematów laboratoriów oraz zadania indywidualnego.

Forma i warunki zaliczenia:

Moduł kończy się egzaminem. Ocena końcowa jest wystawiana na podstawie ocen z pięciu zadań laboratoryjnych wykonanych samodzielnie przez studentów na zajęciach laboratoryjnych (dopuszcza się realizację zadanie 5, tj. projektu w zespołach dwuosobowych). Z całości zajęć (wykłady oraz laboratoria) przewidywany jest egzamin pisemny sprawdzający wiedzę teoretyczną i praktyczną w postaci sprawdzianu lub alternatywnie zadań problemowych. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim, w tym projektu wykonanego samodzielnie (zadanie 5), według schematu:

regularne zajęcia (5 tematów x 20 pkt) – 100 pkt, przy czym każde zadanie oceniane jest w zakresie: wiedzy praktycznej (wykonanie ćwiczenia laboratoryjnego) – do 10 pkt., obrony wykonanego zadania i związanej z nim wiedzy teoretycznej – do 5 pkt. oraz sprawozdania z wykonanego zadania (zadania 1-4) oraz z przygotowanej instrukcji korzystania z odpowiedniego toolbox-a (zadanie 5) – do 5 pkt.,

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania ze wszystkich form aktywności studenta na zajęciach laboratoryjnych co najmniej 51 pkt. Na tej formie zajęć student może maksymalnie uzyskać do 100 pkt.

Za egzamin pisemny w postaci testu lub zadań problemowych można uzyskać do 100 pkt. Zaliczenie egzaminu jest możliwe po uzyskaniu co najmniej 51 pkt. Każdy student może uzyskać dodatkowe punkty m.in. za aktywność na zajęciach laboratoryjnych i na wykładach (w części podsumowującej wykłady). Ocena końcowa z modułu (po zaliczeniu wszystkich części składowych), jest średnią oceną z laboratorium oraz z egzaminu, przy czym zarówno z ćwiczeń laboratoryjnych jak też z egzaminu, w zależności od sumy uzyskanych punktów (suma ze względu na dodatkową ocenę aktywności studentów może przekroczyć 100 pkt.) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 100 pkt: niedostateczna (F),

- 101 – 120 pkt: dostateczna (E),
- 121 – 140 pkt: dostateczna plus (D),
- 141 – 160 pkt: dobra (C),
- 161 – 180 pkt: dobra plus (B),
- 181 – 200 pkt: bardzo dobra (A).

W przypadku przekroczenia przez studenta wymaganej liczby maksymalnej 100 pkt. student może uzyskać od koordynatora przedmiotu dyplom wyróżnienia z metod sztucznej inteligencji.

Poprawy: Istnieje możliwość jednorazowej poprawy egzaminu (przeprowadzonego w postaci testu sprawdzającego wiedzę teoretyczną oraz wiedzę praktyczną lub w postaci zadań problemowych) w trakcie trwania sesji egzaminacyjnej. Ćwiczenia laboratoryjne praktyczne w każdym bloku tematycznym można jednokrotnie dodatkowo zaliczać w trybie poprawkowym na konsultacjach w trakcie semestru.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	**Obciążenie studenta**
Udział w wykładach	30 godz.
Udział w ćwiczeniach laboratoryjnych, w tym przygotowanie danych liczbowych do eksperymentów praktycznych, zaprojektowanie/zaprogramowanie i wykonanie eksperymentów praktycznych, sprawdzenie poprawności uzyskanych wyników, zaliczenie poszczególnych zadań i projektu indywidualnego	30 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych – zapoznanie się z instrukcją laboratoryjną oraz wykonanie sprawozdań z laboratoriów (4 bloki tematyczne x 1 godz.), a także wykonanie projektu indywidualnego ze sprawozdaniem/instrukcją obsługi (1 blok tematyczny x 2 godz.)	19 godz.
Udział w konsultacjach z przedmiotu	1 godz.
Przygotowanie się do egzaminu z wiedzy teoretycznej i wiedzy praktycznej	19 godz.
Udział w egzaminie	1 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	**4 ECTS**

Studia niestacjonarne

Aktywność	**Obciążenie studenta**
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych – zapoznanie się z instrukcją laboratoryjną, przygotowanie danych liczbowych do eksperymentów praktycznych oraz sprawozdania z zajęć (4 bloki tematyczne x 4 godz.), a także wykonanie projektu indywidualnego ze sprawozdaniem i instrukcja obsługi (1 blok x 6 godz.)	35 godz.
Udział w konsultacjach z przedmiotu	1 godz.
Przygotowanie się do egzaminu z wiedzy teoretycznej i wiedzy praktycznej	35 godz.

Udział w egzaminie	1 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	**4 ECTS**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Programowanie w sieciach komputerowych**	
Nazwa w języku angielskim:	Programming in Computer Networks	
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:	trzeci	
Semestr:	piąty	
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 nowe trendy w architekturze aplikacji sieciowych. Celem modułu jest zaznajomienie studentów z zaawansowanymi protokołami i technologiami sieci komputerowych, oraz wprowadzeniem studentów do programowania z użyciem tych protokołów	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	zasadę działania współczesnych zaawansowanych sieciowych protokołów komunikacyjnych	**K_W07**
W_02	Zasadę działania architektury CNApp	**K_W07**
W_03	zasadę projektowania i działania sieci SDN.	**K_W07**
W_04	wybrane aspekty bezpieczeństwa w sieciach komputerowych.	**K_W07**
W_05	wybrane elementy programowania sieciowego (stosowane w CNApp oraz Service Mesh) w oparciu o język Java.	**K_W07**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	projektować realizować aplikacje sieciowe CNApps na mikroserwisach w architekturze chmury obliczeniowej	**K_U10, K_U11, K_U12, K_U19**
U_02	projektować i zrealizować elastyczność i niezawodność CNApp	**K_U10, K_U21**
U_03	zaimplementować Service Mesh do zarządzania CNApp	**K_U10, K_U21**
U_04	zaprojektować i zaimplementować bezpieczną komunikacje pomiędzy użytkownikami a API Gateway na podstawie protokołu PGP	**K_U10, K_U23**

	KOMPETENCJE SPOŁECZNE Student jest gotów do:	
K_01	podejmowania decyzji i krytycznej oceny własnych rozwiązań w rozwiązywaniu problemów w istniejących sieciach komputerowych uwzględniając istniejące standardy sieciowe	**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. Umiejętność programowania w języku obiektowym (Java).
2. Znajomość podstawowych pojęć technologii sieciowych (zakres przedmiotu Technologie Sieciowe).

Treści modułu kształcenia:

Treści wykładu:

1. **Współczesna architektura aplikacji sieciowej oparta na mikro-serwisach w Chmurach obliczeniowych (CNApp)**
2. **PGP – podpis cyfrowy i AES**
3. **Bezpieczeństwo o CNApp oparte na PGP: protokół rejestracji i logowania**
4. **Data plane - control plane**
5. **MPLS i uogólniomy forwarding: flow tables**
6. **SDN oraz MFV**
7. **Protokół OpenFlow**
8. **Data Centers networking**
9. **Service Mesh ogólnie:** architektura i podstawowa funkcjonalność.
10. **Przegład istniejących open source platform dla service mesh:** Linkerd, Istio, Cilium, Consul, Kuma
11. **Service Mesh:** skalowanie CNApp czyli równoważenie obciążenia i zarządzanie wydajnością oraz zasobami
12. **Service Mesh:** odporność na awarie, czyli rekonfiguracja i odtwarzanie mikrosewisów, które uległy awarii
13. **Protokół SSMMP (https://github.com/sambrosz/SSMMP-a-simple-protocol-for-Service-Mesh-management/tree/main)**
14. **ELONET - Starlink**
15. **Sieci optyczne**
16. **Aplikacje multimedialne**
17. **Zarządzanie sieciami, cześć 1.** Wprowadzenie do zarządzania sieciami, motywacje, główne komponenty. Infrastruktura do zarządzania sieciami (internetowymi). MIB: management information base, SMI: Structure of Management Information - język do definiowania struktur danych.
18. **Zarządzanie sieciami, cześć 2.** SNMP: protokół do zarządzania sieciami. Bezpieczeństwo i administrowanie sieciami. Zdalny nadzór sieci – RMON.
19. **Omówienie najnowszych technologii sieciowych, SDN i 5G.** Przegląd nowych rozwiązań i technologii w sieciach komputerowych. Określenie trendów i perspektyw rozwoju nowych technologii w sieciach komputerowych i aplikacjach sieciowych.

Treści zajęć laboratoryjnych

1. **Implementacja wielowątkowego serwera TCP, separacja logiki wątku od logiki wykonania.**
2. **Asynchroniczna transmisja danych w Javie.**
3. **Zaawansowane mechanizmy transmisji danych: klasy strumieniowe, dzielenie i łączenie strumieni, buforowanie i formatowanie przesyłanych danych.**
4. **Przesyłanie zawartości plików: bajtowo i za pomocą stringów (kodowanie w Base64)**
5. **Programowanie aplikacji sieciowych wykorzystujących bazy danych. Interfejs JDBC do połączeń z bazami danych. Przykład prostej aplikacji sieciowej wykorzystującej bazę danych.**
6. **Architektura CNApp: projekt prostej aplikacji**

7. **Implementacja tej aplikacji CNApp**
8. **Programowanie na PGP - cz. 1.** Generowanie kluczy RSA. Realizacja prostej aplikacji klient serwer z wykorzystaniem bibliotek Java do uwierzytelniania.
9. **Programowanie na PGP - cz. 2.** Klucz symetryczny AES, szyfrowanie danych, a następnie połączenie z uwierzytelnianiem.
10. **Implementacja protokołu rejestracji i logowania w CNApp opartego na PGP**
11. **Prosta implementacja skalowania CNApp**
12. **Prosta implementacja odporności CNApp na awarie**
13. **Założenia do implementacji protokołu SSMMP**
14. **SSMMP: Service API**
15. **SSMMP: Agent API**
16. **SSMMP: Menadżer API**
17. **Implementacja całości SSMMP mna prostej Aplikacji testowej CNApp**
18. **Oddanie i zaliczenie projektu z programowania sieciowego: implementacja prostej CNApp w oparciu o service mesh.**

Literatura podstawowa:

1. James Kurose, Keith Ross - Computer Networking_ A Top-Down Approach, 7th Edition (2017) and 8th Edition (2022)
2. Tom Laszewski, Kamal Arora, Erik Farr, Piyum Zonooz. Cloud Native Architectures: Design high-availability and cost-effective applications for the cloud. 1st ed. Packt Publishing, 2018. Web. 14 Oct. 2022.
3. 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
4. George Miranda. The Service Mesh. Publisher(s): O'Reilly Media, Inc. August 2018. ISBN: 9781492031314

Literatura dodatkowa:

1. K. Krysiak. Sieci Komputerowe - Kompendium. Wydawnictwo Helion 2005
2. T. Sheldon. Wielka Encyklopedia Sieci Komputerowych. Wydawnictwo Robomatic s.c. 1999.Akademia Sieci Cisco. CCNA Exploration, Semestr 1. PWN, Warszawa 2011
3. Guo Deke. Data Center Networking: Network Topologies and Traffic Management in Large-Scale Data Centers, 1st ed. 2022 Edition. Wydawca Springer Nature. ISBN 9789811693700

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 kształcenia osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
U_02	jest sprawdzany przy obronie programistycznego zadania.
U_01, U_3, U_04	sprawdzane w czasie ocenianych zadań na laboratoriach
U_02, W_01 – W_05 i K_01	sprawdzane są na egzaminie. Przykładowe pytania: 1. *Opisać architekturę CNApp.* 2. *Zasady działania sieci SDN.* 3. *PGP i jego zastosowanie w architekturze CNApp* 4. *Jak działa Service Mesh?*
K_02	niektóre zadania na laboratoriach są wykonywane są w grupach,

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 pisemnym. Można na nim uzyskać do 60 pkt. Egzamin będzie zaliczony w przypadku uzyskania co najmniej 30 pkt. Ocena końcowa z modułu (wystawiana po zaliczeniu wszystkich części składowych), w zależności od sumy uzyskanych punktów (maksymalnie 100pkt.) jest następująca (w nawiasach ocena wg skali ECTS):

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

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	**Obciążenie studenta**
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	34 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Przygotowanie się do egzaminu	16 godz.
Obecność na egzaminie	2 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	**4 ECTS**

Studia niestacjonarne

Aktywność	**Obciążenie studenta**
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	47 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Przygotowanie się do egzaminu	18 godz.
Obecność na egzaminie	2 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	**4 ECTS**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Komunikacja i przetwarzanie w chmurze**	
Nazwa w języku angielskim:	Cloud communications and computing	
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:	trzeci	
Semestr:	piąty	
Liczba punktów ECTS:	4	
Imię i nazwisko koordynatora przedmiotu:	dr Dariusz Mikułowski	
Imię i nazwisko prowadzących zajęcia:	mgr Wojciech Nabiałek	
Założenia i cele przedmiotu:	Studenci przystępujący do tego przedmiotu powinni znać programowanie obiektowe oraz potrafić efektywnie korzystać ze środowiska programistycznego Visual Studio. Celem przedmiotu jest zapoznanie studentów z technologiami związanymi z komunikacją i przetwarzaniem w chmurach; z projektowaniem i implementowaniem aplikacji w chmurze, ich debugowania, monitorowaniem i skalowaniem, z przechowywaniem różnego typu informacji w chmurach. Studenci będą również potrafić wykorzystywać poznane technologie i narzędzia do tworzenia aplikacji w chmurze i używać chmury w celu składowania i zarządzania różnego rodzaju danymi.	

Symbol efektu	**Efekty uczenia się: WIEDZA** **Student zna i rozumie:**	**Symbol efektu kierunkowego**
W_01	funkcjonowanie narzędzi i metodyk związanych z projektowaniem i programowaniem aplikacji w chmurze i migrowania istniejących aplikacji do chmury. Rodzaje modeli dostępu do usług w chmurze (IAAS, PAAS, SAAS, IAAAS), rodzaje usług oferowanych w chmurze, najważniejszych dostawców.	**K_W06**
W_02	sposoby przechowywania różnego rodzaju danych w chmurze, usługi do przetwarzania dużych zbiorów danych.	**K_W06**

W_03	Sposoby komunikacji i skalowalności w chmurze, sposoby zapewnienia bezpieczeństwa i monitorowania aplikacji w chmurze.	**K_W06**
W_04	sposoby tworzenia, debugowania, monitorowania i zarządzania aplikacjami w chmurze.	**K_W07, K_W12**
Symbol efektu	**UMIEJĘTNOŚCI** **Student potrafi:**	**Symbol efektu kierunkowego**
U_01	Programistycznie wykorzystać różne sposoby korzystania z chmury do przechowywania różnego rodzaju danych.	**K_U02**
U_02	Zarządzać, debugować i monitorować aplikacjami w chmurze.	**K_U11**, **K_U12**
U_03	Migrować lokalnie tworzone aplikacje do chmury.	**K_U21**
U_04	Zaprojektować, zaimplementować oraz przetestować proste aplikacje w chmurze.	**K_U01**, **K_U02**
Symbol efektu	**KOMPETENCJE SPOŁECZNE** **Student jest gotów do:**	**Symbol efektu kierunkowego**
K_01	Jest gotów do krytycznej oceny posiadanej wiedzy związanej z korzystaniem z infrastruktur dostępnych w chmurze.	**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. Dobra znajomość programowania w językach obiektowych
2. Podstawowa praktyczna znajomość środowisk programistycznych Visual Studio oraz InteliJ Idea,

Treści modułu kształcenia:

Treści dotyczące wykładu:

1. Wprowadzenie do komunikacji i przetwarzania w chmurze: Modele usług chmurowych, PAS, IAAS, SAAS, główni dostawcy usług w chmurze, fizyczne rozmieszczenie centrów (regiony i pary regionów),
2. Wprowadzenie do Microsoft Azure: dostępne usługi na platformie Azure, maszyny wirtualne i ich konfiguracja, tworzenie sieci i podsieci dla maszyn wirtualnych Azure, bezpieczeństwo aplikacji w chmurze Azure.
3. Azure Storage – Tabele: przechowywanie różnego rodzaju danych - strukturalnych, pół-strukturalnych i niestrukturalnych na platformie Azure, dostępne usługi przechowywania danych, blooby kontra object storage, usługi kopii zapasowych i kolejek komunikatów.
4. Aplikacja w chmurze: Migracja aplikacji do chmury, koncepcja i architektura mikroserwisów, zarządzanie usługami.
5. Infrastructure as a code: opisywanie usług w formacie JSON, zarządzanie usługami za pomocą skryptów: pojęcie konteneryzacji (docker composer), zarządzanie usługami za pomocą Kubernetes.
6. Komunikacja i skalowanie w chmurze: konfigurowanie dostępu sieciowego dla maszyn wirtualnych Azure, tworzenie sieci i podsieci wirtualnych, kolejkowanie komunikatów, optymalizacja komunikacji na platformie Azure.
7. Obliczenia w chmurze Azure: programowanie bezserwerowe (serverless), usługi uczenia maszynowego na platformie Azure, Azure functions.

8. Zarządzanie, debugowanie i monitorowanie aplikacji w Microsoft Azure, monitorowanie wykorzystania zasobów, bezpieczeństwo przetwarzania danych.
9. Wprowadzenie do platformy amazon Web services: kategorie usług w AWS, migracja aplikacji do AWS, różne perspektywy wdrażania aplikacji w chmurze AWS, organizacje AWS, obliczenia w AWS, Infrastructure as a code, modele płatności w AWS.
10. Magazyny danych na platformie aws: przechowywanie i zarządzanie danymi.
11. Uczenie maszynowe w chmurze: usługi uczenia maszynowego na platformie Azure, uczenie maszynowe na platformie AWS.
12. Migracja aplikacji do chmury: zaplanowanie migracji, zmiana architektury aplikacji, przeprowadzenie migracji na platformie Azure, przeprowadzenie migracji na platformie AWS. komponowanie usług z różnych platform.

Treści dotyczące zajęć laboratoryjnych:

1. Microsoft Azure: Logowanie do portalu, nawigacja, utworzenie maszyny wirtualnej, połączenie zdalne do maszyny przez RDP, usunięcie zasobów.
2. Bazy danych w chmurze: Stworzenie bazy danych oraz maszyny wirtualnej, połączenie się do bazy danych, stworzenie prostej aplikacji klienckiej, usunięcie zasobów.
3. Magazyny danych w chmurze: Stworzenie magazynu danych oraz maszyny wirtualnej, połączenie się do magazynu danych, stworzenie prostej aplikacji klienckiej, usunięcie zasobów.
4. Komunikacja przy pomocy kolejek danych: Stworzenie kolejek, połączenie się do monitora, stworzenie prostej aplikacji klienckiej, usunięcie zasobów.
5. Tworzenie aplikacji MVC dla Azure: Stworzenie zasobów, stworzenie prostej aplikacji MVC, usunięcie zasobów.
6. Publikowanie aplikacji MVC w chmurze: Automatyczne publikowanie aplikacji MVC z poziomu Visual Studio do chmury Azure.
7. Wstęp do konteneryzacji w chmurze: Tworzenie obrazów docker, repozytorium obrazów w Azure. Tworzenie obrazu docker.
8. Konteneryzacja aplikacji w chmurze: Utworzenie magazynu docker-compose.
9. Konteneryzacja na przykładzie chmury Azure: Utworzenie klastra Azure kubernetes.
10. Uczenie maszynowe w chmurze: Korzystanie z usług uczenia maszynowego w Azure, realizacje w Python i .NET.
11. Monitorowanie wykorzystania zasobów, bezpieczeństwo przetwarzania danych na przykładzie chmury Azure.

Literatura podstawowa:

1. Windows Azure. Wprowadzenie do programowania w chmurze Zbigniew Fryźlewicz, Daniel Nikończuk. Wydawnictwo Helion.
2. Kubernetes - rozwiązania chmurowe w świecie DevOps. Tworzenie, wdrażanie i skalowanie nowoczesnych aplikacji chmurowych John Arundel, Justin Domingus. Wydawnictwo Helion.
3. Amazon Web Services. Podstawy korzystania z chmury AWS Mark Wilkins. Wydawnictwo Helion.

Literatura dodatkowa

1. Chmura Azure. Praktyczne wprowadzenie dla administratora. Implementacja, monitorowanie i zarządzanie ważnymi usługami i komponentami IaaS/PaaS Mustafa Toroman
2. Kubernetes i Docker w środowisku produkcyjnym przedsiębiorstwa. Konteneryzacja i skalowanie aplikacji oraz jej integracja z systemami korporacyjnymi Scott Surovich, Marc Boorshtein

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, ćwiczenia laboratoryjne na stanowiskach komputerowych. Zamieszczanie na stronach internetowych problemów i zadań ćwiczeniowych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektu uczenia się
W_01	Efekt będzie realizowany podczas egzaminu pisemnego w formie testu online z pytaniami otwartymi i zamkniętymi. Przykładowe pytanie: Opisz jednym zdaniem na czym polega dostarczanie usług w chmurze w modelu IAAS.
W_02	Efekt będzie realizowany podczas egzaminu pisemnego w formie testu online z pytaniami otwartymi i zamkniętymi. Przykładowe pytanie: Wymień Najważniejsze usługi do przechowywania danych w chmurze Azure.
W_03	Efekt będzie realizowany podczas egzaminu pisemnego w formie testu online z pytaniami otwartymi i zamkniętymi. Przykładowe pytanie: W jaki sposób można zabezpieczyć swoją aplikację webową w chmurze Azure?
W_04	Efekt będzie realizowany podczas egzaminu pisemnego w formie testu online z pytaniami otwartymi i zamkniętymi. Przykładowe pytanie: Na czym polega technologia Infrastructure as a code której można użyć do zarządzania zasobami w chmurze?
U_01	Efekt będzie weryfikowany podczas zajęć laboratoryjnych oraz poprzez wykonanie przez studenta samodzielnego projektu końcowego. Przykładowe zadanie: Na platformie Azure utwórz maszynę wirtualną a następnie połącz się z nią przez RDP.
U_02	Efekt będzie weryfikowany podczas zajęć laboratoryjnych oraz poprzez wykonanie przez studenta samodzielnego projektu końcowego. Przykładowe zadanie: Utwórz prostą aplikację webową w .NET i wdróż ją w chmurze Azure korzystając z możliwości środowiska Visual studio.
U_03	Efekt będzie weryfikowany podczas zajęć laboratoryjnych oraz poprzez wykonanie przez studenta samodzielnego projektu końcowego. Przykładowe zadanie: Utwórz prostą aplikację we frameworku ANgular i umieść ją w chmurze Azure.
U_04	Efekt będzie weryfikowany podczas zajęć laboratoryjnych oraz poprzez wykonanie przez studenta samodzielnego projektu końcowego. Przykładowe zadanie: Utwórz dwie usługi webowe komunikujące się za pomocą kolejek dostępnych w chmurze Azure.
K_01	Efekt będzie realizowany podczas realizacji zadań na laboratorium.

Forma i warunki zaliczenia:

Przedmiot kończy się egzaminem pisemnym, który ma formę testu elektronicznego zawierającego pytania otwarte i zamknięte. Do egzaminu mogą przystąpić osoby, które uzyskały zaliczenie ćwiczeń. Na zaliczenie ćwiczeń składają się oceny cząstkowe uzyskane podczas ćwiczeń z nauczycielem akademickim, maksymalnie 110 punktów (10 punktów za każde z ćwiczeń oprócz ćwiczeń ostatnich oraz projekt końcowy, za który można uzyskać maksymalnie 40 punktów.

Ćwiczenia laboratoryjne będą zaliczone w wypadku uzyskania powyżej połowy punktów z każdych ćwiczeń oraz powyżej połowy punktów z projektu końcowego.

Podczas egzaminu można uzyskać maksymalnie 50 punktów. Egzamin będzie zaliczony w przypadku uzyskania powyżej połowy punktów. Ocena końcowa z modułu (wystawiana po zaliczeniu wszystkich części składowych), w zależności od sumy uzyskanych punktów (maksymalnie 200 pkt.) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 100 punktów: niedostateczna (F),
- 101 – 120 punktów: dostateczna (E),
- 121 – 140 punktów: dostateczna plus (D),
- 141 – 160 punktów: dobra (C),
- 161 – 180 punktów: dobra plus (B),
- 180 – 200 punktów: bardzo dobra (A).

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	21 godzin
Udział w ćwiczeniach laboratoryjnych	24 godziny
Udział w konsultacjach z przedmiotu	3 godziny
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	42 godzin
Przygotowanie się do egzaminu i	9 godzin
Obecność na egzaminie	1 godz.
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
Udział w konsultacjach godz. z przedmiotu	5 godzina
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	55 godzin
Przygotowanie się do egzaminu	9 godzin
Obecność na egzaminie	1 godz.
Sumaryczne obciążenie pracą studenta	**100 godzin**
Punkty ECTS za przedmiot	**4 ECTS**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Projektowanie obiektowe**	
Nazwa w języku angielskim:	Object oriented design	
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:	trzeci	
Semestr:	piąty	
Liczba punktów ECTS:	**3**	
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 Javie. Potrafią również konstruować i implementować algorytmy. Celem przedmiotu jest zapoznanie studentów z podstawowymi zagadnieniami związanymi z refaktoryacją kodu oraz projektowaniem obiektowym tj. m.in. wzorcami projektowymi oraz zasadami projektowania obiektowego.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	zagadnienia z zakresu projektowania obiektowego.	**K_W05**
W_02	zagadnienia z zakresu wzorców projektowych i refaktoryzacji kodu.	**K_W05**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	dokonać poprawnie refaktoryzacji kodu źródłowego.	**K_U12**
U_02	zastosować zasady projektowania obiektowego	**K_U12**
U_03	poprawnie zastosować wzorce projektowe w realizowanych zadaniach programistycznych	**K_U12**
	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	krytycznej oceny posiadanej wiedzy oraz do uznawania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych z zakresu informatyki	**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:	

- Znajomość podstaw programowania (języki Python lub Java)

Treści modułu kształcenia:

Wykład

1. Zasady projektowania obiektowego SOLID
2. Wprowadzenie do refaktoryzacji kodu
3. Refaktoryzacja - konstrukcja metod
4. Refaktoryzacja - przenoszenie składowych pomiędzy obiektami
5. Refaktoryzacja - organizacja danych
6. Wzorce projektowe - strategia, singleton, dekorator
7. Wzorce projektowe - łańcuch zobowiązań, polecenie iterator, prototyp
8. Wzorce projektowe - mediator, pamiątka, obserwator, budowniczy
9. Wzorce projektowe - stan, metoda szablonowa, odwiedzający, metoda wytwórcza
10. Wzorce projektowe - adapter, most, kompozyt, fabryka abstrakcyjna
11. Wzorce projektowe - fasada, pyłek, pełnomocnik

Ćwiczenia laboratoryjne

Podczas zajęć laboratoryjnych studenci będą implementować zadania dotyczące tematu omawianego na wykładzie.

Literatura podstawowa:

1. R. C. Martin, Czysta architektura. Struktura i design oprogramowania. Przewodnik dla profesjonalistów, Helion, 2018
2. E. Freeman, E. Robson, Wzorce projektowe. Rusz głową! Tworzenie rozszerzalnego i łatwego w utrzymaniu oprogramowania obiektowego. Wydanie II, Helion, 2021

Literatura dodatkowa:

1. M. Fowler, Architektura systemów zarządzania przedsiębiorstwem. Wzorce projektowe., Helion, 2005

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, ćwiczenia laboratoryjne wspomagane technikami multimedialnymi. Zamieszczanie na stronach internetowych problemów i zadań laboratoryjnych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01, W_02	będą sprawdzane podczas oceny realizacji zadań na laboratorium, student musi wykazać się wiedzą w zakresie znajomości zasad projektowania obiektowego i wzorców projektowych. Przykładowe pytania: wyjaśnij na czym polega zasada jednej odpowiedzialności, otwarte/zamknięte, podstawienia Liskov, segregacji interfejsów,

	odwrócenia zależności. Scharakteryzuj wzorzec projektowy: metoda szablonowa, stan, obserwator. Omów zasady refaktoryzacji kodu związane z metodami.
U_01	Przykładowe zadanie praktyczne - refaktoryzacja kodu: ● Student otrzymuje kod z powtarzalnością, złą strukturą, brakiem enkapsulacji lub z duplikacją. ● Zadanie: przeprowadzenie refaktoryzacji z uzasadnieniem zastosowanych zmian (np. ekstrakcja metod, eliminacja powtórzeń, uproszczenie logiki, eliminacja kodu martwego).
U_02	Zadanie projektowe – implementacja klas zgodnie z zasadami OOP: ● Student ma zaprojektować i zaimplementować system z użyciem dziedziczenia, enkapsulacji, polimorfizmu, kompozycji (np. system rezerwacji, aplikacja do zarządzania danymi biologicznymi), zachowując zasady SOLID
U_03	Zadanie praktyczne – implementacja wzorca: ● Student ma zaimplementować konkretny wzorzec (np. Singleton, Factory, Observer, Strategy) w określonym kontekście problemowym.
K_01	Obserwacja postawy studenta, jego umiejętności analizy i syntezy poszczególnych informacji podczas zajęć i konsultacji

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem na ocenę. Zaliczenie przedmiotu odbywa się na podstawie zaliczenia laboratorium, za które można otrzymać maksymalnie 100 punktów. Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej 51 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	21 godzin
Udział w ćwiczeniach laboratoryjnych	24 godzin
Udział w konsultacjach godz. z przedmiotu	1 godziny
Przygotowanie się do realizacji zadań na laboratorium	29 godzin
Sumaryczne obciążenie pracą studenta	75 godzin
Punkty ECTS za przedmiot	3

Studia niestacjonarne

Aktywność	Obciążenie studenta

Udział w wykładach	15 godzin
Udział w ćwiczeniach laboratoryjnych	15 godzin
Udział w konsultacjach godz. z przedmiotu	1 godziny
Przygotowanie się do realizacji zadań na laboratorium	44 godzin
Sumaryczne obciążenie pracą studenta	75 godzin
Punkty ECTS za przedmiot	3

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Analiza danych**	
Nazwa w języku angielskim:	Data analysis	
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:	trzeci	
Semestr:	piąty	
Liczba punktów ECTS:	3	
Imię i nazwisko koordynatora przedmiotu:	dr Anna Wawrzyńczak-Szaban	
Imię i nazwisko prowadzących zajęcia:	dr Anna Wawrzyńczak-Szaban	
Założenia i cele przedmiotu:	Zakłada się, że student posiada umiejętność programowania oraz znajomość statystyki i analizy matematycznej. Celem przedmiotu jest zapoznanie studentów z pojęciami i metodami analizy oraz ich praktycznego zastosowania w wybranym środowisku obliczeniowym.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	podstawowe zagadnienia związane z analizą danych.	**K_W01, K_W10,**
W_02	założenia poszczególnych modeli analizy danych oraz potrafi wybrać odpowiedni, zaimplementować i zastosować do konkretnego zagadnienia.	**K_W08, K_W10**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	analizować i przygotować odpowiedni zbiór danych do wybranej metody analizy danych i ją zaimplementować.	**K_U01, K_U22**
U_02	tworzyć wybrane modele analizy danych w wybranym środowisku obliczeniowym i ocenić ich efekty.	**K_U07, K_U12, K_U22**
U_03	wykorzystać poznane metody do rozwiązywania problemów klasyfikacji i predykcji. Potrafi dokonać ich estymacji oraz zwizualizować i zinterpretować wyniki.	**K_U01, K_U20**

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:	
Umiejętność programowania, znajomość statystyki i analizy matematycznej	
Treści modułu kształcenia:	

1. Wstęp do analizy danych. Cele stosowania. Technologie pokrewne. Rodzaje analityki danych. Etapy procesu analizy danych. Metody reprezentacji informacji.
2. Eksploracyjna analiza danych. Wstępne przetwarzanie danych. Analiza statystyczna. Czyszczenie, transformacja, redukcja danych. Algorytm k-najbliższych sąsiadów.
3. Reprezentacja wiedzy w analizie danych. Reprezentowanie danych wejściowych i interpretacja danych wyjściowych.
4. Rozkład prawdopodobieństwa. Prawdopodobieństwo warunkowe. Twierdzenie Bayesa. Naiwny klasyfikator Bayesowski.
5. Hipotezy i wnioski. Analiza korelacji i regresji. Testowanie hipotez.
6. Tworzenie modeli opartych na danych. Regresja liniowa. Regresja wieloraka. Poprawność modelu.
7. Redukcja wymiarowości. Analiza głównych składowych. Ekstrakcja i selekcja cech
8. Drzewa decyzyjne. Entropia. Lasy losowe.
9. Analiza skupień. Miary odległości. Grupowanie metodą k-średnich. Algorytm DBSCAN. Algorytm centroidów.
10. Grupowanie hierarchiczne. Grupowanie obiektów i cech wstępujące i zstępujące.
11. Sztuczne sieci neuronowe. Perceptrony. Jednokierunkowe sieci neuronowe. Propagacja wsteczna.

Literatura podstawowa:

1. Joel Grus „Data science od podstaw” 2020, Helion S.A
2. Deborah Nolan Joseph Gonzalez Sam Lau, „Poznaj Data Science”, 2024, Promise
3. Vaughan, Daniel, Data science : wyzwania i rozwiązania : jak zostać ekspertem analizy danych, 2025, Helion S.A

Literatura dodatkowa:

1. Geron Aurelien, Uczenie maszynowe z użyciem Scikit-Learn, Keras i TensorFlow, Wydanie III, Helion 2023
2. Research Practitioner’s Handbook on Big Data Analytics. S. Sasikala, PhD, D. Renuka Devi, & Raghvendra Kumar, PhD (Editor)© 2023 Apple Academic Press, Inc. Co-published with CRC Press (Taylor & Francis)
3. Jugal K. Kalita, Dhruba K. Bhattacharyya, Swarup Roy „Fundamentals of Data Science”, Academic Press, 2023

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, laboratorium komputerowe wykorzystujące środowisko umożliwiające prowadzenie analizy danych np. MatLab, R, Python. Zamieszczanie na stronach internetowych problemów, zadań oraz materiałów ćwiczeniowych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01-W_02	Efekt będzie weryfikowany poprzez kolokwium pisemne na ostatnim wykładzie
U_01- U_03	Efekt będzie weryfikowany na dwóch kolokwiach w trakcie zajęć laboratoryjnych.

Forma i warunki zaliczenia:

Przedmiot kończy się zaliczeniem z oceną. Ocena końcowa ustalana jest na podstawie wyników uzyskanych podczas zajęć laboratoryjnych oraz pisemnego kolokwium końcowego.

W trakcie semestru odbywają się dwa kolokwia w ramach laboratoriów oraz jedno kolokwium pisemne przeprowadzane na ostatnim wykładzie. Warunkiem zaliczenia każdego z kolokwiów jest uzyskanie co najmniej 51% możliwych do zdobycia punktów.

Ocena końcowa wyliczana jest jako średnia ważona, w której:

- 60% stanowi suma punktów uzyskanych z dwóch kolokwiów laboratoryjnych,
- 40% stanowią punkty uzyskane z kolokwium pisemnego z wykładu.

Sumarycznie można zdobyć 100 punktów.

Ocena końcowa, w zależności od sumy uzyskanych punktów (maksymalnie 100 pkt) jest następująca (w nawiasach ocena wg skWali 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	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Samodzielne przygotowanie się do ćwiczeń i kolokwiów	17 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Przygotowanie się do kolokwium z wykładu	10 godz.
Sumaryczne obciążenie pracą studenta	75 godz.
Punkty ECTS za przedmiot	**3**

Studia niestacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do ćwiczeń i kolokwiów	27 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Przygotowanie się do kolokwium	15 godz.
Sumaryczne obciążenie pracą studenta	75 godz.
Punkty ECTS za przedmiot	**3**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Cyfrowe przetwarzanie obrazu i dźwięku**	
Nazwa w języku angielskim:	Digital Image and Sound Processing	
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:	czwarty	
Liczba punktów ECTS:	3	
Imię i nazwisko koordynatora przedmiotu:	dr Mirosław Szaban	
Imię i nazwisko prowadzących zajęcia:	dr Andrzej Salamończyk, mgr inż Monika Wereszczeńska	
Założenia i cele przedmiotu:	Zakłada się, że student zna podstawy programowania. Celem przedmiotu jest nabycie przez studentów wiedzy z zakresu przekształcania cyfrowych form danych obrazowych oraz cyfrowego przetwarzania dźwięku. Studenci zdobędą umiejętności wykorzystania algorytmów poprawy jakości obrazów, usuwania uszkodzeń form obrazowych, filtrowania danych obrazowych, wykrywania cech cyfrowego obrazu a także analizy i obróbki dźwięku.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	pojęcia i metody związane z pozyskiwaniem obrazów cyfrowych, ich opisem i reprezentacją.	**K_W01, K_W02, K_W03, K_W09**
W_02	metody i podstawowe algorytmy związane z przetwarzaniem obrazów w postaci cyfrowej. Poprawą ich jakości w zakresie podstawowym i uniwersalnym, manipulacją obrazem poprzez operacje jedno i wieloargumentowe.	**K_W01, K_W02, K_W03, K_W09**
W_03	metody i podstawowe algorytmy związane z operacjami, filtrowaniem liniowym i nieliniowym obrazów, w tym usuwaniem szumu, segmentacją i operacjami morfologicznymi.	**K_W01, K_W02, K_W03, K_W09**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	posłużyć się wybranymi popularnymi aplikacjami do przetwarzania obrazu i dźwięku oraz narzędziami w nich dostępnymi. Potrafi korzystać z wybranych środowisk programistycznych i ich bibliotek pod katem ich wykorzystania w obróbce obrazu i dźwięku.	**K_U01, K_U02, K_U06, K_U07, K_U11, K_U22**
U_02	wybrać właściwy algorytm naprawy uszkodzonego obrazu i zastosować go w celu usunięcia usterek. Potrafi poprawić jakość pliku obrazu (jasność, kontrast, nasycenie barw). Potrafi usunąć wybrany rodzaj szumu, stosuje wybrane filtry poprawy jakości obrazu.	**K_U01, K_U02, K_U06, K_U07, K_U11, K_U22**

U_03	pracować z plikami dźwiękowymi korzystając w wybranych aplikacji i środowisk programistycznych.	**K_U01, K_U02, K_U06, K_U07, K_U11, K_U22**

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. Umiejętność podstaw programowania.

Treści modułu kształcenia:

1. **Wprowadzenie do środowiska MatLab.** Zapoznanie z metodyką pracy w oprogramowaniu.
2. **Wstęp do przetwarzania obrazów.** Metody grafiki komputerowej. Formy danych obrazowych. Przekształcenia form danych obrazowych. Podstawowe przekształcenia geometryczne. Barwa, modelowanie, animacja.
3. **Dyskretyzacja obrazów z gradacją kontrastu.** Algorytmy dyskretyzacji obrazów z gradacją kontrastu: próbkowanie, kwantowanie, rozdzielczość obrazu.
4. **Przetwarzanie obrazów z gradacją kontrastu.** Histogram i korekcja histogramu. Algorytmy macierzy sąsiedztwa. Algorytmy filtrowania obrazów. Filtry kierunkowe, dwuczęściowe, aproksymacji funkcyjnej.
5. **Operacje punktowe jednoargumentowe:** LUT, negacja, progowanie, zmiana jasności i kontrastu, korekcja Gamma, poprawa jakości z użyciem histogramu: rozciągnięcie i wyrównanie histogramu, zmiana intensywności kanałów, balans kolorów, przesuniecie kolorów.
6. **Operacje punktowe wieloargumentowe:** dodawanie obrazów proporcjonalne i ważone, kanał alfa, dodawanie obrazów z saturacją, odejmowanie obrazów, obrazy różnicowe, usuwanie tła, mnożenie i potęgowanie obrazów, maskowanie obrazów, dzielenie obrazów – wykrywanie ruchu.
7. **Operacje kontekstowe: filtry liniowe**, konwolucja i splot, wygładzanie obrazu, usuwanie szumu białego i typu „Salt and Pepper", filtr Gaussa, rozmycie kierunkowe, wyostrzanie obrazów, gradient i Laplasjan, wykrywanie linii.
8. **Operacje kontekstowe: filtry nieliniowe**, filtr medianowy, usuwanie szumu, wygładzanie konserwatywne, filtry wartości środkowej, uśredniające i adaptacyjne, filtr odplamiający Crimminsa, inne filtry nieliniowe…
9. **Segmentacja.** Rodzaje i algorytmy segmentacji. Segmentacja dwupoziomowa i wielopoziomowa. Progowanie. Wykrywanie cech w obrazach cyfrowych.Pojęcie krawędzi. Operator krzyżowy Robertsa. Operator Sobela. Kąt i kierunek gradientu. Operator kompasowy. Maski Prewitta. Operator Kirscha. Detekcja krawędzi oparta na laplasjanie. Detektor Frei'ai Chen'a. Detektor krawędzi Canny'ego. Detekcja linii. Filtr wykrywający linie.
10. **Nieliniowe przetwarzanie obrazów (morfologia matematyczna).** Pojęcie łączności (spójności). Suma Minkowskiego. Element strukturalny. Operacje podstawowe: Dylatacja, Erozja. Operacje złożone: otwarcie morfologiczne (opening), zamknięcie morfologiczne (closing). Operacja „Hit-and-Miss" („Hit-or-Miss"). Operacje „większość białych" i „większość czarnych". Operacje pogrubiania. Rozszerzenie morfologii matematycznej na obrazy w skali szarości.
11. **Tworzenie, edycja i przetwarzanie plików dźwiękowych.** Odczytywanie, modyfikacja i zapis plików dźwiękowych w wybranych formatach w środowisku Julia lub MatLab. Wykonywanie podstawowych efektów dźwiękowych. Standardy zapisu dźwięku. Podstawowe operacje na plikach dźwiękowych. **Ukrywanie obrazu w pliku dźwiękowym.**

Podsumowanie wykładu. Omówienie zagadnień egzaminacyjnych.

Literatura podstawowa:

1. Michał Choraś, Ryszard S. Choraś Editors, Image processing and communications challenges. 9, Advances in Intelligent Systems and Computing, 2194-5357 ; 681, Springer 2018
2. Witold Malina, Maciej Smiatacz, Cyfrowe przetwarzanie obrazów, EXIT, 2012

3. Anna Korzyńska, Małgorzata Przytulska, Przetwarzanie obrazów – ćwiczenia, Wydawnictwo PJWSTK, 2006

Literatura dodatkowa:

1. Theo Pavlidis, Grafika i przetwarzanie obrazów, WNT, 1987.
2. Z. Wróbel, R. Koprowski, Praktyka przetwarzania obrazów w programie Matlab, wyd. EXIT 2004.
3. Tomasz P. Zieliński, Cyfrowe przetwarzanie sygnałów. Od teorii do zastosowań, WKŁ, 2014

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, laboratorium komputerowe wykorzystujące środowiska i aplikacje obróbki obrazu i dźwięku. Zamieszczanie na stronach internetowych problemów, zadań oraz materiałów ćwiczeniowych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01- W_03	Kolokwium pisemne na ostatnim wykładzie. Przed kolokwium studenci będą mieli dostęp do pełnej listy pytań. Przykładowe pytania: • Opisz, na czym polega kwantowanie. (W_01) • Na czym polega operacja rozciągania histogramu i jakie ma zastosowanie? (W_02) • Czym jest filtr medianowy i jakie ma zastosowanie? (W_03)
U_01- U_03	Na bieżąco, na każdych zajęciach poza pierwszym i ostatnim w postaci zadań praktycznych. Tematyka następnego laboratorium będzie podana tydzień przed zajęciami. Student, na podstawie podanej literatury, musi się do nich przygotować samodzielnie lub korzystając z konsultacji. Przykładowe zadanie (sprawdza efekty: U_01, U_02): Korzystając z programu Julia lub MatLab, dokonać poprawy jakości obrazów z plików pomocy (krokodyl.png, morze.jpg oraz twarze.jpg), poprzez: • rozciąganie histogramu (imadjust('filename')), • wyrównanie histogramu (histeq('filename')), • adaptacyjne wyrównanie histogramu (adapthisteq('filename')). Przykłady poleceń MatLab: imadjust('obraz.jpg'), histeq('obraz.jpg'), adapthisteq('obraz.jpg'), imadjust(f), histeq (f), adapthisteq (f), imadjust(f(:, :, i)), i=1, 2, 3, histeq(f(:, :, i)), i=1, 2, 3, adapthisteq(f(:, :, i)), i=1, 2, 3 Polecenie do samodzielnego wykonania: Utwórz galerię (tabelę) 2×2 (subplot(2, 2, n);), w której komórkach umieść obrazy po manipulacji histogramem: • komórka 1: krokodyl.png, komórka 2: krokodyl.png + rozciągnięcie histogramu, • komórka 3: krokodyl.png + wyrównanie histogramu, komórka 4: krokodyl.png + adaptacyjne wyrównanie histogramu, Odpowiedź na pytanie: Która metoda daje najlepszą jakość? Przykładowe zadanie (sprawdza efekt: U_03): Korzystając z programu MATLAB: 1. Jeżeli masz możliwość nagrania własnej próbki dźwiękowej(najlepiej głosu),

	nagraj ją oraz przekonwertuj ją do rozszerzenia .wav, w innym przypadku korzystaj z pliku dwa.wav 2. Wczytaj plik dźwiękowy ([y,Fs]=audioread('filename')). Co znajdzie się w zmiennej y i Fs? 3. Sprawdź rozmiar macierzy wczytanego pliku (size(y)). Dźwięk jest mono czy stereo? 4. Wyświetl wykres częstotliwości (plot(y)) 5. Uruchom plik dźwiękowy (sound(y,Fs))

Forma i warunki zaliczenia:

Przedmiot kończy się zaliczeniem na ocenę. Ocenę końcowa zależy od liczby uzyskanych punktów w stosunku 60% z laboratorium oraz 40% wykład (kolokwium z wykładu).

Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim. Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej połowy punktów z poszczególnych form aktywności studenta: regularne zajęcia – co najmniej 31 pkt. Na tej formie zajęć student może maksymalnie uzyskać 60 pkt.

Z kolokwium z wykładu można uzyskać maksymalnie 40 pkt. Wykład będzie zaliczony w przypadku uzyskania co najmniej 21 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	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	10 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Przygotowanie się do kolokwium	17 godz.
Sumaryczne obciążenie pracą studenta	75 godz.
Punkty ECTS za przedmiot	**3 ECTS**

Studia niestacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	20 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Przygotowanie się do kolokwium	22 godz.
Sumaryczne obciążenie pracą studenta	75 godz.
Punkty ECTS za przedmiot	**3 ECTS**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Wprowadzenie do systemów kognitywnych**	
Nazwa w języku angielskim:	Introduction to cognitive systems	
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:	trzeci	
Semestr:	piąty	
Liczba punktów ECTS:	4	
Imię i nazwisko koordynatora przedmiotu:	dr hab. inż. Jerzy Tchórzewski	
Imię i nazwisko prowadzących zajęcia:	mgr inż. Dariusz Ruciński	
Założenia i cele przedmiotu:	Zakłada się, że student posiada wiedzę i umiejętności z przedmiotów: „Algorytmy i złożoność", „Wybrane Paradygmaty Programowania", „Sztuczna inteligencja". Celem zajęć jest nabycie wiedzy i umiejętności na temat projektowania i implementacji współczesnych rozwiązań stosowanych w systemach kognitywnych.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	podstawowe założenie, teorię oraz zasady projektowania i implementacji systemów kognitywnych	**K_W10** **K_W06**
W_02	istotę metodologii modelowania systemów kognitywnych, w tym sposobów stosowania metod w celu uzyskiwania różnych modeli systemów kognitywnych	**K_W10**
W_03	możliwości zastosowania technologii kognitywnych w robotyce humanoidalnej, bioinformatyce, inżynierii biomedycznej, itp.	**K_W03**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	wykorzystywać metody i algorytmy kognitywne w zakresie projektowania i implementacji wiedzy rozmytej, asocjacyjnej, skojarzeniowej, rekurencyjnej, itp. w procesie modelowania systemów kognitywnych w środowisku j. Matlab/j. Python	**K_U22**
U_02	stosować metody modelowania systemów kognitywnych takie jak uczenie maszynowe, uczenie neuronalne, analiza skupień, itp. dostępne w środowisku j. Matlab/j. Python	**K_U11**
U_03	samodzielnie rozwiązywać problemy w zakresie projektowania i implementacji systemów kognitywnych, w tym potrafi w sposób aktywny korzystać z nowoczesnych narzędzi dostępnych w środowisku j. Matlab/j. Python, np. w zakresie robotów humanoidalnych, sztucznych organów ludzkich (np. sztuczna ręka, sztuczna kończyna, sztuczne serce, itp.), a	**K_U02**

	też w miarę potrzeby z narzędzi chmurowych dostarczanych przez twórców różnych środowisk programistycznych (np. Microsoft Azure)	
Symbol efektu	**Kompetencje społeczne Student jest gotowy do:**	**Symbol efektu kierunkowego**
K_01	do podejmowania decyzji, krytycznej oceny działań własnych i studentów z grupy laboratoryjnej, w której uczestniczy. Jest gotowy do przyjmowania odpowiedzialności za skutki własnej pracy wykonywanej na zajęciach laboratoryjnych, w tym zwłaszcza przy realizacji zadania indywidualnego.	**K_K02**
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:

Ułatwieniem w studiowaniu jest znajomość zagadnień z przedmiotów: „Algorytmy i złożoność", „Wybrane Paradygmaty Programowania", „Sztuczna inteligencja", umiejętność programowania z wykorzystaniem języków bardzo wysokiego poziomu z rozbudowanymi bibliotekami lub znajomość literatury obowiązującej w tym przedmiotach (podstawy algorytmiki i heurystyki, kombinatoryki i statystyki, liczb rozmytych, algebry liniowej i rachunku wektorowo-macierzowego, topologii i teorii miary, itp.).

Treści modułu kształcenia:

Treści dotyczące wykładu:

1. **Wprowadzenie do systemów kognitywnych.** Kognitywistyka jako nauka o procesach poznawczych, w tym o ich modelowaniu**.** Myślenie, rezolucja (np. wnioskowanie), proces obliczeniowy. Architektura systemu kognitywnego. Podstawowe podsystemy systemu kognitywnego: System sztucznej inteligencji (inteligentny system sterowania) jako model procesów myślowych człowieka wytwarzającego sztuczne myśli (procesy), System inteligentny (wykonawczy) jako model poznania stanów zewnętrznych (wielkości wejściowych) oraz System organizacji wewnętrznej jako model poznania stanów wewnętrznych (wielkości zmiennych stanu). Cechy charakterystyczne procesu myślenia. Gramatyka kognitywna. Sztuczny mózg i sztuczne neurony. Asocjacyjne Neurony Pulsacyjne. Model neuronów asocjacyjno-pulsacyjnych, itp.
2. **Modelowanie rozmyte systemów kognitywnych**. Logika rozmyta i liczby rozmyte. Fuzyfikacja i defuzyfikacja. Badania nad logiką rozmytą. Funkcja przynależności. Podstawy zbiorów rozmytych i możliwości ich wykorzystania m.in. w sztucznych sieciach neuronowych. Możliwości programowania systemów rozmytych z wykorzystaniem środowiska MATLAB/PYTHON. Budowa systemu rozmytego stosowanego w modelowaniu podsystemów kognitywnych. Podstawowe modele wnioskowania: model wnioskowania Mamdaniego-Zadeha, model Takagi-Sugeno-Kanga. Rozmyte SSN: Rozmyta SSN TSK, Rozmyta SSN Wanga-Mendela. Projektowanie systemu rozmytego w j. Matlab/j. Python.
3. **Modelowanie asocjacyjnych systemów kognitywnych.** Rekurencyjne SSN. Asocjacyjne o Autoasocjacyjne SSN. Algorytmy uczenia Asocjacyjnych SSN. Rodzaje Asocjacyjnych SSN (Hopfielda, Bolzmana, Hybrydowe SSN asocjacyjne, Dwukierunkowa SSN heteroautoasocjacyjna BAM, Rozrzedzona pamięć rozproszona SDM. Neurodynamika pamięci asocjacyjnych.
4. **Sztuczne systemy skojarzeniowe.** Mechanizmy skojarzeniowe (asocjacyjne). Reguły asocjacyjne. Klasy asocjacyjne. Powiązania asocjacyjne pomiędzy neuronami. Semassele jako jednostki semantyczno-skojarzeniowe. Efekt pamięci w systemach skojarzeniowych. Asocjacyjne modele elementów AAS. Biologiczne podstawy modelowania neuronów. Dwukierunkowa pamięć asocjacyjna (BAM). Przykłady praktyczne.
5. **Wielokierunkowe (MAM), w tym Dwukierunkowa pamięć asocjacyjna (BAM).** Istota Wielokierunkowej pamięci asocjacyjne MAM, w tym dwukierunkowej BAM. Mechanizmy skojarzeniowe w SSN BAM. Pojęcie łańcucha skojarzeń mentalnych. Biwalentne wzorce

wejściowe. Architektura i algorytm uczenia BAM. Zapis i odczyt pamięci dwukierunkowej. Ograniczenia SSN BAM. Własności funkcji energii. Przykład pamięci i ocena jakości. Odległość Hamminga. Zastosowania pamięci asocjacyjnych.

6. **Metody hierarchiczne analizy skupień w systemach kognitywnych.** Techniki eksploracji danych. Odkrywanie asocjacji. Istota klasyfikacji i grupowania. Hierarchiczne algorytmy analizy skupień. Rola różnego typu miar odległości. Rola dendrogramu. Techniki aglomeracyjne i techniki rozdrobnieniowe. Przykład metody Warda.
7. **Metody niehierarchiczne analizy skupień w systemach kognitywnych.** Istota, wady i zalety metod niehierarchicznych. Istota metody k-średnich. Metoda grupowania wokół centroidów. Sztuczne sieci neuronowe Kohonena. Porównanie metod hierarchicznych oraz niehierarchicznych. Metody oceny poprawności przeprowadzonej analizy. Przykład analizy skupień za pomocą metody SOM. Przykład analizy skupień za pomocą metody Competitive Learning.
8. **Uczenie maszynowe w systemach kognitywnych.** Algorytmy uczenia maszynowego. Historia uczenia maszynowego. Cele uczenia maszynowego. Metody uczenia maszynowego (indukcyjne, probabilistyczne, z przykładów, uczenie ze wzmocnieniem, regresyjne metody uczenia maszynowego, drzewa decyzyjne, lasy losowe, itp.). Problemy uczenia maszynowego. Uczenie półnadzorowane. Przykład regresyjnego uczenia w systemach kognitywnych.
9. **Uczenie maszynowe ze wzmocnieniem w systemach kognitywnych**. Istota uczenia maszynowego przez wzmacnianie**.** Pojęcie środowiska, agenta, bufora. Uczenie wsadowe i przyrostowe. Metody Markowa, Bellmana, inne metody programowania dynamicznego. Uczenie się agenta (systemu) z wartościującym sprzężeniem zwrotnym. Uczenie z opóźnieniem oraz uczenie z natychmiastowym wzmocnieniem.
10. **Rola dynamiki-rozwoju-chaosu w Systemach Kognitywnych**. Wznoszenia się stanu systemu na wyższe, kolejne progi informacyjne. Naturalny wzrost dynamiki rozwoju. Rola częstotliwości w rozwoju systemów kognitywnych, w tym wibracji na poziomie wyższych częstotliwości. Trójwymiarowa sztuczna świadomość. Energia jako łącznik między informacją i materią w systemach kognitywnych. Teoria chaosu, w tym efekt motyla. Koncepcja „liczby Lyapunova" jako narzędzia do określania stopnia chaosu w systemie dynamicznym. Nieliniowe systemy dynamiczne. Właściwości pamięciowe SSN chaotycznych.
11. **Architektury kognitywne, czyli jak zbudować sztuczny umysł**. Architektury symboliczne. Architektury emergentne. Architektury hybrydowe o dwóch typach: lokalno-rozproszone oraz symboliczno-konekcjonistyczne. Realizacje praktyczne systemów kognitywnych. Model organizacji kontroli motorycznej w układzie nerwowym oparty na podwójnym sterowaniu: ośrodkowym i obwodowym. Organizacja neuronalnej kontroli trajektorii ruchu opartej na wewnętrznych modelach inwersyjnych.

Treści dotyczące laboratorium:

1. **Modelowania wiedzy rozmytej w systemach kognitywnych.** Nabycie praktycznych umiejętności w zakresie projektowania i implementacji systemów kognitywnych w środowisku j. Python/j. Matlab z wykorzystaniem możliwości analizy skupień w zakresie klasyfikacji, regresji i klasteryzacji zwanej grupowaniem.
2. **Projektowanie asocjacyjnych sztucznych sieci neuronowych w środowisku j. Python/j. Matlab**. Nabycie praktycznych umiejętności w zakresie projektowania systemów kognitywnych w środowisku j. Python oraz w środowisku j. Matlab z wykorzystaniem możliwości modelowania systemów, procesów i obiektów w przestrzeni pamięci asocjacyjnych wraz z narzędziami dostępnymi w odpowiednich bibliotekach środowisk programistycznych.

3. **Analiza skupień w środowisku j. Matlab/j. Python z wykorzystaniem dostępnych bibliotek programów**. Nabycie praktycznych umiejętności w zakresie projektowania i implementacji systemów kognitywnych w środowisku j. Python/j. Matlab z wykorzystaniem możliwości metod analizy skupień systemów w zakresie: klasyfikacji, regresji i klasteryzacji zwanej grupowaniem.
4. **Realizacja indywidualnych projektów systemów kognitywnych w środowisku j. Matlab/j. Python z wykorzystaniem odpowiednich bibliotek programów**. Nabycie praktycznych umiejętności i kompetencji społecznych w zakresie projektowania i implementacji systemów kognitywnych, w tym z pamięcią asocjacyjną, skojarzeniową, rozmytą, rekurencyjną, neuronalną, uczenia maszynowego, itp. w środowisku j. Python/j. Matlab. Na tych laboratoriach studenci otrzymują od prowadzącego laboratoria indywidualne zadania do zaprojektowania i implementacji różnego typu systemów kognitywnych, to jest m.in. z uwzględnieniem systemów rozmytych, systemów asocjacyjnych, systemów skojarzeniowych, systemów rekurencyjnych, systemów neuronalnych i systemów uczenia maszynowego, itp.

Literatura podstawowa:

1. W. Duch, Czym jest kognitywistyka, https://www.fizyka.umk.pl/~duch/cog-book/kognitywistyka.htm [dostęp: wrzesień 2024, lipiec 2025].
2. A. Horzyk, Sztuczne systemy skojarzeniowe i asocjacyjna sztuczna inteligencja. AOW EXIT, Warszawa 2013, stron 277.
3. J. Tchórzewski, Metody sztucznej inteligencji i informatyki kwantowej w ujęciu teorii sterowania i systemów, OW UPH, Siedlce 2021, pages 343.

Literatura dodatkowa:

1. P. Deitel, H. Deitel, Python dla programistów z analizami przypadków wprowadzających w tematykę sztucznej inteligencji, Helion, Gliwice 2020, stron 707.
2. Embedded MATLABTM, User's Guide, Copyright 2002-2025, The MathWorks, Inc, Natic, USA.
3. S. Wierzchoń, M. Kłopotek, Algorytmy analizy skupień, PWN, Warszawa 2017.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany jest technikami multimedialnymi. Ćwiczenia laboratoryjne – zajęcia praktyczne z wykorzystaniem środowiska j. Matlab/j. Python. Do laboratorium udostępnione zostaną instrukcje laboratoryjne, a do wykładów print screeny prezentacji.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „Wymień podstawowe podsystemy systemu kognitywnego. Omów architekturę systemu kognitywnego. Scharakteryzuj możliwości modelowania procesu myślowego człowieka".
W_02	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „Wymień podstawowe modele wnioskowania stosowane w systemach rozmytych. Jakie rodzaje sztucznych sieci neuronowych opracowano na ich podstawie. Omów warstwy neuronów w jednej z nich".
W_03	Efekt realizowany na egzaminie w formie pisemnej. Przykładowe pytanie: „Wymień elementy występujące w Sztucznym Systemie Skojarzeniowym. Gdzie znajdują się skojarzenia i wiedza w tym systemie. Jaki jest praktyczny wpływ promienia sąsiedztwa na tworzenie łańcuchów wiedzy".
U_01	Efekt realizowany w trakcie zajęć laboratoryjnych na bloku 1 zajęć laboratoryjnych (Lab 1-3).

	Z wykorzystaniem systemów rozmytych może zostać zaprojektowany i zaimplementowany rozmyty system kognitywny dotyczący następujących zadań, np. analiza nastrojów społecznych sprowadzona do: analizy opinii klientów lub monitorowania mediów społecznościowych, która może zostać wykorzystana do zdefiniowania systemu rozmytego z przypisaniem wartości danych takich jak m.in.: są pozytywne, są negatywne czy są neutralne, itp. Oprócz skupiania się na polaryzacji tekstu, analiza może także wykryć określone uczucia i emocje, takie jak: złość, radość, zawiedzenie się, smutek, żal, zadowolenie, itp. Przykład zadania do realizacji na kolejnych trzech zajęcia 2-lekcyjnych: **Lab 1**. Samodzielne przygotowanie przez studentów danych do eksperymentu badawczego w uzgodnieniu z prowadzącym zajęcia laboratoryjne oraz poznanie projektowania systemów kognitywnych z wykorzystaniem obu języków programowania. **Lab. 2**. Projektowanie prostej Rozmytej Sztucznej Sieci Neuronowej z krótką i długą pamięcią jako modelu do wnioskowania w j. Matlab z wykorzystaniem bibliotek: Deep Learning Toolbox-a, Fuzzy Logic Toolbox-a, itp. Na tych zajęciach studenci wykonują swoje indywidualne zadania w konsultacji z prowadzącym zajęcia laboratoryjne i na koniec je zaliczają oraz opracowują raport z przebiegu laboratorium. **Lab. 3**. Projektowanie prostej Rozmytej Sztucznej Sieci Neuronowej z krótką i długą pamięcią jako modelu do wnioskowania w j. Python/j. Matlab z wykorzystaniem bibliotek: NumPy, Theano, TensorFlow, Keras, PyTorch, Matplotlib, itp. Na tych zajęciach studenci wykonują swoje indywidualne zadania samodzielnie i na koniec je zaliczają i opracowują raport z przebiegu laboratorium.
U_02	Efekt realizowany w trakcie zajęć laboratoryjnych na bloku 2 oraz na bloku 3 zajęć laboratoryjnych (Lab 4-6) oraz (Lab 7-9). **Na Lab 4-6** zadanie dotyczy indywidualnego projektowania i implementacji funkcjonowania systemu kognitywnego z wykorzystaniem Rekurencyjnej SSN jako pamięci autoasocjacyjnej. Przykład zadania do realizacji na kolejnych trzech zajęcia 2-lekcyjnych: **Lab 4.** Samodzielne przygotowanie przez studentów danych do eksperymentu badawczego w uzgodnieniu z prowadzącym zajęcia laboratoryjne oraz poznanie projektowania Asocjacyjnych Sztucznych Sieci Neuronowych w środowisku MATLAB-a oraz/lub w środowisku PYTHONA z wykorzystaniem odpowiedniego języka programowania, tj. j. Matlab/j. Python. **Lab. 5.** Projektowanie prostej Autoasocjacyjnej Sztucznej Sieci Neuronowej z krótką i długą pamięcią jako modelu wnioskowania w j. Matlab z wykorzystaniem bibliotek: Deep Learning Toolbox-a oraz Statistics and Machnine Learning Toolbox-a, itp. Na tych zajęciach studenci wykonują swoje indywidualne zadania w konsultacji z prowadzącym zajęcia laboratoryjne i na koniec je zaliczają oraz opracowują raport z przebiegu laboratorium. **Lab. 6**. Projektowanie prostej Autoasocjacyjnej Sztucznej Sieci Neuronowej z krótką i długą pamięcią jako modelu do wnioskowania w j. Python z wykorzystaniem bibliotek: NumPy, Theano, TensorFlow, Keras, PyTorch, Matplotlib, itp. Na tych zajęciach studenci wykonują swoje indywidualne zadania samodzielnie i na koniec je zaliczają i opracowują raport z przebiegu laboratorium. **Na Lab 7-9** realizowane będą zadania indywidualne przez studentów dotyczące projektowania i implementacji systemów kognitywnych w środowisku j. Python/j. Matlab z wykorzystaniem możliwości analizy skupień w zakresie klasyfikacji, regresji i klasteryzacji zwanej grupowaniem. Przykład zadania do realizacji na kolejnych trzech zajęcia 2-lekcyjnych: **Lab 7**. Samodzielne przygotowanie przez studentów danych do eksperymentu badawczego w uzgodnieniu z prowadzącym zajęcia laboratoryjne oraz poznanie projektowania systemów kognitywnych w obu językach programowania, tj. z

	wykorzystaniem analizy w środowisku MATLAB i Simulink z wykorzystaniem Statistics and Machine Learning Toolbox-a (SML) lub alternatywnie środowiska j. Python. **Lab. 8**. Przygotowanie eksperymentu badawczego i przeprowadzenie analizy skupień stosując metodę Warda oraz metodę k-means w środowisku MATLAB-a i Simulink-a z wykorzystaniem bibliotek: Statistic and Machine Learning Toolbox-a oraz ewentualnie Deep Learning Toolbox-a oraz alternatywnie środowiska j. Python. Na tych zajęciach studenci wykonują swoje indywidualne zadania w konsultacji z prowadzącym zajęcia laboratoryjne i na koniec je zaliczają w sposób praktyczny, a następnie po zaliczeniu wysyłają ustalony z prowadzącym laboratoria raport z uzyskanych końcowych wyników badań. **Lab. 9**. Przeprowadzanie analizy uzyskanych wyników badań, ich przetestowanie i poprawienie oraz opracowanie końcowego sprawozdania z badań laboratoryjnych przeprowadzonych w odpowiednim środowisku programistycznym z wykorzystaniem j. Python lub alternatywnie j. Matlab Na tych zajęciach studenci zaliczają teoretycznie ten blok zajęć laboratoryjnych wykazując się znajomością przynajmniej wiedzy zwartej w instrukcji i prezentowanej na wykładach z Wprowadzenia do Systemów Kognitywnych. Należy zwrócić szczególną uwagę na interpretację uzyskanych wyników badań.
U_03	Efekt będzie realizowany na zajęciach L10-L12 i będzie dotyczył projektu indywidualnego każdego studenta. Przykład zadania do realizacji na kolejnych trzech zajęcia 2-lekcyjnych: **Lab 10**. Każdy student po uzyskaniu tematu na laboratorium w tym bloku tematycznym (Lab 10-12) samodzielne przygotowuje koncepcję eksperymentu oraz dane niezbędne mu do wykonania eksperymentów badawczych w uzgodnieniu z prowadzącym zajęcia laboratoryjne oraz poznaje projektowanie systemów kognitywnych w wybranym języku programowania, tj. z wykorzystaniem środowisku MATLAB i Simulink z wykorzystaniem odpowiedniego toolbox-a lub alternatywnie środowiska j. Python z dostępnymi jego bibliotekami. **Lab. 11**. Każdy student indywidualnie przygotowuje i projektuje wykonanie eksperymentu badawczego z wykorzystaniem odpowiedniej metody i metodologii postępowania oraz odpowiedniego środowiska wybranego do implementacji na bazie koncepcji zaakceptowanej na Lab 10. Na tych zajęciach studenci wykonują swoje indywidualne zadania w konsultacji z prowadzącym zajęcia laboratoryjne i na koniec je zaliczają w sposób praktyczny, a następnie po zaliczeniu wysyłają ustalony z prowadzącym laboratoria raport z uzyskanych końcowych wyników badań. **Lab. 12**. Na tych zajęciach studenci przeprowadzają badania testujące na swoim systemie kognitywnym lub odpowiednim jego podsystemie. Można w tym celu wykorzystać w środowisku j. Matlab bibliotekę Simulink, a w j. Python odpowiednie możliwości implementacyjne tego środowiska obliczeniowego. Uzyskana aplikacja ma zostać przetestowana i poprawiona. Końcowym elementem badań jest opracowanie końcowego sprawozdania z badań laboratoryjnych przeprowadzonych w odpowiednim środowisku programistycznym z wykorzystaniem j. Python lub alternatywnie j. Matlab. Na tych zajęciach studenci zaliczają także teoretycznie ten blok zajęć laboratoryjnych wykazując się znajomością wiedzy wynikającej z przypisanego mu środowiska obliczeniowego. Należy zwrócić szczególną uwagę na konieczność interpretacji uzyskanych wyników badań.
K_01	Obserwacja postawy studenta na zajęciach laboratoryjnych
Forma i warunki zaliczenia:	
Moduł kończy się egzaminem. Ocena końcowa jest wystawiana na podstawie ocen z pięciu zadań laboratoryjnych wykonanych samodzielnie przez studentów na zajęciach laboratoryjnych (dopuszcza się realizację zadanie 4, tj. projektu w zespołach dwuosobowych). Z całości zajęć (wykłady oraz laboratoria) przewidywany jest egzamin pisemny sprawdzający wiedzę teoretyczną i praktyczną w postaci sprawdzianu lub alternatywnie zadań problemowych. Na zaliczenie laboratorium składają się oceny	

cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim, w tym projektu wykonanego samodzielnie (zadanie 4), według schematu:

●regularne zajęcia (34 tematy x 25 pkt) – 100 pkt, przy czym każde zadanie oceniane jest w zakresie: wiedzy praktycznej (wykonanie ćwiczenia laboratoryjnego) – do 15 pkt., obrony wykonanego zadania i związanej z nim wiedzy teoretycznej – do 5 pkt. oraz sprawozdania z wykonanego zadania (zadania 1-3) oraz z przygotowanej instrukcji korzystania z odpowiedniego toolbox-a (zadanie 4) – do 10 pkt.,

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania ze wszystkich form aktywności studenta na zajęciach laboratoryjnych co najmniej 51 pkt. Na tej formie zajęć student może maksymalnie uzyskać do 100 pkt.

Za egzamin pisemny w postaci sprawdzianu lub zadań problemowych można uzyskać do 100 pkt. Zaliczenie egzaminu jest możliwe po uzyskaniu co najmniej 51 pkt. Każdy student może uzyskać dodatkowe punkty m.in. za aktywność na zajęciach laboratoryjnych i na wykładach (w części podsumowującej wykłady). Ocena końcowa z modułu (po zaliczeniu wszystkich części składowych), jest średnią oceną z laboratorium oraz z egzaminu, przy czym zarówno z ćwiczeń laboratoryjnych jak też z egzaminu, w zależności od sumy uzyskanych punktów (suma ze względu na dodatkową ocenę aktywności studentów może przekroczyć 100 pkt.) jest następująca (w nawiasach ocena wg skali ECTS):

- 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

Poprawy: Istnieje możliwość jednorazowej poprawy egzaminu (przeprowadzonego w postaci testu sprawdzającego wiedzę teoretyczną oraz wiedzę praktyczną lub w postaci zadań problemowych) w trakcie trwania sesji egzaminacyjnej. Ćwiczenia laboratoryjne praktyczne w każdym bloku tematycznym można jednokrotnie dodatkowo zaliczać w trybie poprawkowym na konsultacjach w trakcie semestru.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Przygotowanie się do egzaminu	16 godz.
Udział w egzaminie	2 godz.
Studia indywidualne z tematów realizowanych na wykładach	10
Przygotowanie się do zajęć laboratoryjnych i wykonanie sprawozdania z laboratoriów	25 godz.
Udział w konsultacjach z przedmiotu	2 godz.
Sumaryczne obciążenie pracą studenta	100 godz.

Punkty ECTS za przedmiot	**4**
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Przygotowanie się do egzaminu	16 godz.
Udział w egzaminie	2 godz.
Studia indywidualne z tematów realizowanych na wykładach	25 godz.
Przygotowanie się do zajęć laboratoryjnych i wykonanie sprawozdania	25 godz.
Udział w konsultacjach z przedmiotu	2 godz.
Sumaryczne obciążenie pracą studenta	100 godz.
Punkty ECTS za przedmiot	**4**

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Środowiska programowania aplikacji wirtualnych i multimedialnych**	
Nazwa w języku angielskim:	Virtual and Multimedia Application Programming Environment	
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:	trzeci	
Semestr:	piąty	
Liczba punktów ECTS:	**4**	
Imię i nazwisko koordynatora przedmiotu:	dr Grzegorz Terlikowski	
Imię i nazwisko prowadzących zajęcia:	dr Grzegorz Terlikowski	
Założenia i cele przedmiotu:	Założeniem tego przedmiotu jest znajomość przez słuchaczy zagadnień związanych z programowaniem obiektowym i stron WWW. Celem przedmiotu jest zapoznanie się zagadnieniami związanymi z aplikacjami wirtualnymi i multimedialnymi.	

Symbol efektu	Efekty uczenia się	Symbol efektu kierunkowego
	WIEDZA **Student zna i rozumie:**	
W_01	na czym polega wirtualizacja oraz zna założenia dotyczące programowania aplikacji wirtualnych i multimedialnych.	**K_W02, K_W10**
W_02	różne atrybuty multimediów i wie jak je wykorzystać w tworzonych przez siebie aplikacjach.	**K_W02, K_W05, K_W06, K_W10**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	w praktyce wykorzystać narzędzia umożliwiające tworzenie aplikacji wirtualnych i multimedialnych. Potrafi wesprzeć się adekwatnymi materiałami dydaktycznymi oraz posługiwać się pojęciami związanymi z multimediami i wirtualizacją.	**K_U01, K_U10, K_U11**
U_02	implementować aplikacje multimedialne w technologiach desktopowych takich jak WPF, MAUI lub ElectronJS.	**K_U10, K_U11, K_U12**

	KOMPETENCJE SPOŁECZNE **Student jest gotów do:**	
K_01	podejmowania decyzji i krytycznej oceny własnych rozwiązań w rozwiązywaniu zadań projektowych z zakresu implementacji aplikacji multimedialnych	**K_K01, K_K03**
K_02	uznania znaczenia wiedzy w rozwiązywaniu zadań projektowych z zakresu aplikacji wirtualnych i multimedialnych.	**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:

Umiejętność wykorzystania języków: HTML(5), Java Script, CSS, Java.

Treści modułu kształcenia:

Wykłady:

1. **Wprowadzenie do aplikacji multimedialnych.** Atrybuty multimediów, usługi multimedialne. Przegląd rozwiązań umożliwiających tworzenie bogatych, multimedialnych aplikacji internetowych.
2. **Wprowadzenie do WPF**. Historia, architektura i cechy platformy. Cechy charakterystyczne WPF. Kontenery i pozycjonowanie. Język XAML i jego cechy. Paradygmat MVVM.
3. **Zaawansowane cechy WPF**. Mechanizm pędzli. Animacje i ich scenariusze. Grafika 3D (układ współrzędnych 3D, rodzaje kamer, transformacje 3D).
4. **Środowisko MAUI.** Historia, architektura. Przegląd możliwości. Obsługa podstawowych zdarzeń. Porównanie z WPF.
5. **Środowisko Java FX**. Historia, architektura i cechy platformy. Język FXML. Efekty, transformacje, animacje i inne multimedia.
6. **Środowisko ElectronJS**. Tworzenie aplikacji hybrydowych na pulpit. Architektura i właściwości frameworka ElectronJS. Przegląd możliwości: menu, zapis/odczyt pliku.
7. **Środowisko QT.** Przegląd możliwości. Język QML Tworzenie interfejsów z użyciem Qt Widgets i QtQuick. Komunikacja między obiektami Qt z wykorzystaniem mechanizmu sygnałów i slotów.
8. **Wprowadzenie do wirtualizacji**. Co to jest wirtualizacja? Podstawowe definicje. Przegląd podstawowych typów wirtualizacji - zalety, wady i zastosowanie każdego z nich.
9. **Wprowadzenie do progresywnych aplikacji webowych (PWA)**. Czym są aplikacje PWA? Zasady tworzenia aplikacji PWA. Plik manifestu, Service Workery, Cache API i praca w trybie offline.
10. **Wprowadzenie do JHipster.** JHipster jako przykład platformy umożliwiającej szybkie tworzenie nowoczesnych multimedialnych aplikacji webowych w architekturze monolitycznej lub mikroserwisów.

Laboratoria:

1. **Wprowadzenie do Windows Presentation Fundation** . Tworzenie pierwszej aplikacji WPF w Visual Studio IDE, wprowadzenie do języka XAML, obsługa podstawowych zdarzeń, pozycjonowanie kontrolek, rozszerzenia znaczników (ang. markup extensions) w XAML.
2. **Wiązanie, konwertowanie i walidacja danych w WPF.** Wiązanie danych do pojedynczej wartości, kierunki wiązania danych, konwertowanie danych, walidacja danych,
3. **Wiązanie danych c.d. i zewnętrzne katalogi zasobów**. Kontrolka DataGrid i wiązanie danych do kolekcji, tworzenie i wykorzystywanie szablonów.

4. **Paradygmat MVVM, polecenia i obsługa baz danych SQLite .** Paradygmant MVVM, • Polecenia, • Obsługa baz danych (SQLite) i wstrzykiwanie zależność.
5. **Animacje w WPF.** Animacje proste i animacje z klatkami kluczowymi.
6. **Szablony kontrolek i wyzwalacze**. Szablony kontrolek (ControlTemplate), wyzwalacze właściwości, zdarzeń i danych.
7. **Tworzenie aplikacji hybrydowych w WPF**. Kontrolka WebView2, komunikacja z częścią webową, tworzenie menu, obsługa plików.
8. **Elementy 3D w aplikacjach WPF**. Kontener Viewport3D i umieszczane w nim elementy wizualne, nakładanie tekstur na elementy 3D, rozpoznawanie kliknięte obszaru, interaktywne elementy UI na bryłach 3D.
9. **Wprowadzenie do .NET MAUI**. Tworzenie pierwszej aplikacji w .NET MAUI w Visual Studio IDE, obsługa podstawowych zdarzeń, pozycjonowanie kontrolek, rozszerzenia znaczników i paradygmat MVVM, animacje.
10. **Wprowadzenie do ElectronJS**. Tworzenie szkieletu aplikacji, obsługa własnego menu. wykorzystanie dodatkowych skryptów w procesach renderujących.
11. **Komunikacja między-procesowa w ElectronJS**. Ograniczenia w dostępie do API, Komunikacja między-procesowa, Obsługa plików
12. **Obrona projektów indywidualnych**. Testowanie i weryfikacja poprawności realizacji projektu indywidualnego. Możliwe niewielkie modyfikacje oprogramowania, w celu weryfikacji samodzielności jego wykonania

Literatura podstawowa:

1. A. Kempa, Wprowadzenie do WPF. Tworzenie aplikacji w WPF przy użyciu XAML i C#. Helion 2016.
2. D. Vuika. Build over 9 cross-platform desktop applications from scratch. ISBN: 978-1838552206, 2019.
3. R. McKendrick, S. Gallagher. Docker. Programowanie aplikacji dla zaawansowanych. Wydanie II. Wydawnictwo: Helion, 2018.

Literatura dodatkowa:

1. A. Nathan. WPF 4.5. Księga eksperta (ebook). Helion 2015.
2. U. Piechota, J. Piechota. JavaFX. Tworzenie graficznych interfejsów użytkownika. Helion 2021.
3. Miell, A. H. Sayers. Docker w praktyce (ebook). Wydawnictwo Naukowe PWN. 2020.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, ćwiczenia laboratoryjne wspomagane technikami multimedialnymi. Zamieszczanie na stronach internetowych problemów i zadań laboratoryjnych

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Sprawdzane podczas pisemnego zaliczenia wykładu na ocenę. Przykładowe pytanie: ● Na czym polega wirtualizacja aplikacji. Scharakteryzuj sposoby wirtualizacji aplikacji.
W_02	Sprawdzane podczas pisemnego zaliczenia wykładu na ocenę. Przykładowe pytanie: ● Co to jest interaktywność, adaptacyjność i dostępność.

U_01	Systematycznie sprawdzane na zajęciach laboratoryjnych. Materiały na następne laboratorium muszą być dostępne co najmniej tydzień przed zajęciami. Student, na podstawie podanej literatury, musi się do nich samodzielnie, lub korzystając z konsultacji, przygotować. Przykładowe zadanie: ● Zapoznaj się z dokumentacją kontrolki WebView2 w WPF a następnie zaimplementuj aplikację hybrydową, wykorzystując mechanizmy komunikacji pomiędzy częścią natywną (WPF) a webową (HTML/JS).
U_02	Systematycznie sprawdzane na zajęciach laboratoryjnych. Przykładowe zadania: ● Dodaj animacje i efekty do przycisków w swojej aplikacji WPF. Animacje mają być inne dla różnych zdarzeń (naciśnięcie przycisku, najechanie, zjechanie z przycisku). ● Wykorzystując klasę WebView2 zaimplementuj prostą aplikację hybrydową w WPF – formularz rejestracyjny. Użyj animacji CSS3 w celu podkreślenia multimedialnego charakteru aplikacji.
K_01	Weryfikowany, w oparciu o posiadaną wiedzę i umiejętności w czasie zajęć laboratoryjnych, podczas zaliczania zadania indywidualnego, a także będą sprawdzane na egzaminie.
K_02	Weryfikowany, w oparciu o posiadaną wiedzę i umiejętności w czasie zajęć laboratoryjnych, podczas zaliczania zadania indywidualnego.

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem z oceną. Ocena końcowa jest wystawiana na podstawie zajęć laboratoryjnych i jednego kolokwium pisemnego przeprowadzonego na ostatnim wykładzie. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim oraz z samodzielnie wykonanego zadania indywidualnego według schematu:

- Regularne zajęcia – 39 pkt.,
- Obrona zadania indywidualnego – 21 pkt.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej połowy punktów z poszczególnych form aktywności studenta: regularne zajęcia – co najmniej 20 pkt., obrona indywidualnego zadania – co najmniej 11 pkt. Na tej formie zajęć student może maksymalnie uzyskać 60 pkt.

Za pisemne kolokwium można uzyskać do 40 pkt. Zaliczenie kolokwium jest możliwe po uzyskaniu co najmniej 21 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).

Poprawy:

1. Poprawa dwóch ćwiczeń w trakcie zajęć w semestrze.
2. Jednorazowa poprawa projektu indywidualnego przed sesją egzaminacyjną.

Jednorazowa poprawa kolokwium przed sesją egzaminacyjną.

Bilans punktów ECTS:	
Studia stacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	18 godz.
Praca nad projektem indywidualnym	15 godz.
Udział w konsultacjach godz. z przedmiotu	2 godz.
Przygotowanie się do kolokwium	20 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	4 ECTS
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	25 godz.
Praca nad projektem indywidualnym	18 godz.
Udział w konsultacjach godz. z przedmiotu	2 godz.
Przygotowanie się do kolokwium	25 godz.
Sumaryczne obciążenie pracą studenta	**100 godz.**
Punkty ECTS za przedmiot	4 ECTS

Sylabus przedmiotu / modułu kształcenia		
Nazwa przedmiotu/modułu kształcenia:	**Symulacja komputerowa**	
Nazwa w języku angielskim:	Computer simulation	
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:	trzeci	
Semestr:	piąty	
Liczba punktów ECTS:	3	
Imię i nazwisko koordynatora przedmiotu:	dr Ewa Szczepanik	
Imię i nazwisko prowadzących zajęcia:	dr Ewa Szczepanik	
Założenia i cele przedmiotu:	Założeniem tego przedmiotu jest znajomość przez słuchaczy zagadnień związanych z podstawami fizyki i analizą matematyczną oraz zakłada się, że student posiada umiejętność programowania. Celem zajęć jest zapoznanie studentów z wybranymi zagadnieniami związanymi z modelowaniem i symulacjami komputerowymi.	

Symbol efektu	**Efekty uczenia się**	**Symbol efektu kierunkowego**
	WIEDZA **Student zna i rozumie:**	
W_01	etapy modelowania systemów i procesu tworzenia modelu symulacyjnego	**K_W06**
W_02	Orientuje się w obecnym stanie oraz najnowszych trendach rozwojowych z zakresu metod modelowania i symulacji komputerowej,	**K_W07, K_W09**
	UMIEJĘTNOŚCI **Student potrafi:**	
U_01	przeanalizować wybrany proces lub zjawisko oraz utworzyć jego model matematyczny.	**K_U01, K_U07**
U_02	ułożyć algorytm i przeprowadzić symulację dla zadanego problemu i zaimplementować go w języku MATLAB lub innym wybranym języku programowania.	**K_U07, K_U12, K_U22**
U_03	utworzyć pełny model symulacyjny wybranego zjawiska/procesu oraz przeprowadzić symulację komputerową, eksperymentować z doborem parametrów.	**K_U09,K_U20**

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:

Warunkiem uczestnictwa w zajęciach jest znajomość zagadnień z przedmiotu „Matematyka I", „Fizyka dla Informatyków" lub znajomość literatury obowiązującej w tych przedmiotach.

Treści modułu kształcenia:

Treści dotyczące wykładu:

1. Idea modelowania. Pojęcie modelu. Klasyfikacja modeli. Symulacja systemów naturalnych.
2. Etapy modelowania systemów. Model cybernetyczny i jego modyfikacje.
3. Model fizyczny. Przykłady modeli fizycznych wybranych systemów.
4. Model matematyczny. Ogólna postać modelu matematycznego.
5. Klasyfikacja modeli matematycznych. Przykłady modeli matematycznych wybranych systemów.
6. Model symulacyjny. Proces budowy modelu symulacyjnego. Reguły upraszczania modeli. Techniczne aspekty badań symulacyjnych.
7. Metoda Monte Carlo. Próbkowanie ważone. Procesy Markowa. Równowaga szczegółowa. Algorytm Metropolisa. Symulacje Monte Carlo w różnych zespołach statystycznych.
8. Wprowadzenie do modelowania systemów masowej obsługi (teoria kolejek). Procesy stochastyczne. Procesy Markowa. Zastosowanie teorii kolejek w badaniach wydajności systemów komputerowych.
9. Symulacja przykładowych modeli matematycznych dynamicznych systemów ciągłych za pomocą oprogramowania Matlab i Simulink (systemy mechaniczne, elektryczne, ekonomiczne). Narzędzia programistyczne. Grafika w środowiskach programistycznych. Wizualizacja wyników symulacji komputerowej przy użyciu podstawowych narzędzi programistycznych.
10. Symulacje przykładowych procesów mechanicznych i elektrycznych.
11. Kolokwium pisemne

Treści dotyczące laboratorium:

1. Idea tworzenia modeli w środowisku MatLab (Simulink) na podstawie przykładów wahadła matematycznego, figury Lissajou itp. Tworzenie czytelnych modeli.
2. Równanie Lotki-Volterry. Modelowanie i symulacja z doborem parametrów
3. Symulacja modelu epidemiologicznego
4. Tworzenie własnych bloków, systemów i funkcji w Simulinku,
5. Symulacje Monte Carlo.
6. Metoda Simpleks w modelowaniu i symulacji
7. Modelowanie systemów masowej obsługi (teoria kolejek). Procesy stochastyczne. Procesy Markowa.
8. Zaawansowane pakiety oprogramowania symulacyjnego (np. MatLab, MODELLUS, AnyLogic, Easy Java Simulations, FlexSlim). Wymiana danych pomiędzy różnymi środowiskami użytkowymi.
9. FlexSim. Wprowadzenie do środowiska. Projektowanie schematu linii produkcyjnej
10. FlexSim. Symulacja schematu linii produkcyjnej dla zadanych parametrów
11. FlexSim. Kontrola jakości
12. FlexSim. Metoda Simpleks

Literatura podstawowa:

1. R. Kotowski, Podstawy modelowania i symulacji komputerowej, PJATK, 2019
2. Gutenbaum J.: Modelowanie matematyczne systemów. Akademicka Oficyna Wydawnicza EXIT. Warszawa 2003.
3. Kotowski Romuald, Tronczyk Piotr, Modelowanie i symulacje komputerowe. Wydaw. Uniw. Kazimierza Wielkiego w Bydgoszczy. Bydgoszcz 2009.

Literatura dodatkowa:

1. Tikhonenko O.: Modele obsługi masowej w systemach informacyjnych. Akademicka Oficyna Wydawnicza EXIT. Warszawa 2003
2. Matyka Maciej, Symulacje komputerowe w fizyce. Helion, Gliwice 2021

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, laboratorium komputerowe wykorzystujące środowisko umożliwiające prowadzenie symulacji komputerowych np. MatLab czy FlexSim. Na stronie internetowej prowadzącego zamieszczane są materiały z problemami i zadaniami laboratoryjnymi.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Efekt realizowany na kolokwium w formie pisemnej. Przykładowe pytanie: - Wymień i scharakteryzuj etapy modelowania systemów - Opisz różnice pomiędzy modelem fizycznym i matematycznym. - W jakich zagadnieniach mają zastosowanie procesy Markowa?
W_02	Efekt realizowany na kolokwium w formie pisemnej. Przykładowe pytanie: - Wymień i krótko scharakteryzuj znane Ci współczesne środowiska do przeprowadzania symulacji komputerowych
U_01	Efekt realizowany w trakcie zajęć laboratoryjnych. Sprawdzane będą umiejętności przeanalizowania oraz zamodelowania np. zjawiska poruszania się wahadła matematycznego
U_02	Efekt realizowany w trakcie zajęć laboratoryjnych. Sprawdzane będą umiejętności przeprowadzenia symulacji zamodelowanych problemów w środowisku Matlab, Excel czy FlexSim np. Przeprowadzenie symulacji modelu gradacji owadów Choristoneura fumiferana
U_03	Efekt realizowany w trakcie zajęć laboratoryjnych. Sprawdzane będą umiejętności przeprowadzenia pełnej symulacji problemów czy zjawisk np. zadanie laboratoryjne Należy zamodelować przebieg epidemii choroby według równań Kermacka-McKendricka oraz przeprowadzić symulację z doborem parametrów

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem na ocenę.

Ocena z przedmiotu składa się z dwóch ocen cząstkowych:

- oceny z zajęć laboratoryjnych,
- oceny z kolokwium w formie pisemnej, przeprowadzonego na ostatnim wykładzie

Na ocenę z zajęć laboratoryjnych składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim, za które można uzyskać sumarycznie 60 pkt. Zaliczenie zajęć laboratoryjnych możliwe jest po uzyskaniu co najmniej 51% liczby punktów z tej formy zaliczenia.

Zaliczenie wykładu w formie kolokwium nastąpi w przypadku uzyskania co najmniej 51% liczby z puli 40 punktów przewidzianej dla tej formy zaliczenia.

Sumarycznie można zdobyć 100 punktów.

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).

Poprawy: Uzyskanie poprawkowego zaliczenia laboratoriów możliwe jest w sesji egzaminacyjnej, odpowiednio przed drugim terminem egzaminu pisemnego.

Bilans punktów ECTS:	
Studia stacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Przygotowanie się do kolokwium	12 godz.
Przygotowanie się do zajęć laboratoryjnych	15 godz.
Udział w konsultacjach z przedmiotu	3 godz.
Sumaryczne obciążenie pracą studenta	75 godzin
Punkty ECTS za przedmiot	**3**
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Przygotowanie się do kolokwium	15 godz.
Przygotowanie się do zajęć laboratoryjnych	27 godz.
Udział w konsultacjach z przedmiotu	3 godz.
Sumaryczne obciążenie pracą studenta	75 godzin
Punkty ECTS za przedmiot	**3**

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Tworzenie interfejsów użytkownika**
Nazwa w języku angielskim:	Creating user interfaces
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:	trzeci
Semestr:	piąty
Liczba punktów ECTS:	3
Imię i nazwisko koordynatora przedmiotu:	dr Marcin Stępniak
Imię i nazwisko prowadzących zajęcia:	dr Marcin Stępniak
Założenia i cele przedmiotu:	Studenci przystępujący do tego przedmiotu powinni posiadać podstawową wiedzę z zakresu grafiki komputerowej i technologii internetowych. Celem przedmiotu jest zapoznanie studentów z historią i ewolucją komunikacji człowiek-komputer, a także ze specyfiką zmysłów (np. wzrok, słuch) wykorzystywanych w tej komunikacji. Studenci poznają zasady i normy projektowania IU oraz sposoby oceny ich jakości. Poza powyższymi studenci poznają także narzędzia i technologie wykorzystywane w projektowaniu IU.

Symbol efektu	Efekty uczenia się **WIEDZA** **Student zna i rozumie:**	Symbol efektu kierunkowego
W_01	zagadnienia związane z podbudowaną teoretycznie wiedzą w zakresie zasad i metodyki projektowania interfejsów użytkownika ze szczególnym uwzględnieniem fazy analizy i modelowania konceptualnego.	**K_W03, K_W06, K_W09**
W_02	zagadnienia dotyczące oceny użyteczności multimedialnego interfejsu użytkownika.	**K_W09**
W_03	technologie i narzędzia wykorzystywane w projektowaniu IU.	**K_W06, K_W12**
W_04	aspekty mające wpływ na trendy rozwojowe interfejsów użytkownika. Zna i rozumie najistotniejsze nowe osiągnięcia w zakresie interfejsów użytkownika.	**K_W03, K_W12**

Symbol efektu	UMIEJĘTNOŚCI Student potrafi:	Symbol efektu kierunkowego
U_01	opracować szczegółową dokumentację projektu interfejsu użytkownika ze szczególnym uwzględnieniem widoków użytkownika.	**K_U08**
U_02	zaproponować ulepszenia istniejących rozwiązań, ocenić przydatność i możliwość wykorzystania nowych osiągnięć w zakresie metod do projektowania i implementacji systemów informatycznych z naciskiem na interfejs użytkownika.	**K_U10, K_U15**
U_03	zastosować odpowiednie techniki i narzędzia do projektowania i budowy interfejsów użytkownika.	**K_U02, K_U10**
U_04	integrować wiedzę, pochodzącą z różnych źródeł, z dziedziny informatyki i innych dyscyplin, stosując podejście systemowe, z uwzględnieniem aspektów pozatechnicznych (w tym psychologii poznania).	**K_U01**
Symbol efektu	**KOMPETENCJE SPOŁECZNE Student jest gotów do:**	**Symbol efektu kierunkowego**
K_01	myślenia i działania w sposób kreatywny. Zna ograniczenia własnej wiedzy i rozumie potrzebę dalszego kształcenia.	**K_K01**
K_02	przekazywania społeczeństwu informacji i opinii dotyczących osiągnięć informatyki.	**K_K02**

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 zakresu grafiki komputerowej.
2. Wiedza i umiejętności z zakresu technologii WWW.

Treści modułu kształcenia:

Wykłady:

1. **Wprowadzenie do interfejsów użytkownika**: historia, znaczenie i ewolucja.
2. **Percepcja obrazów i dźwięków**: filozofia widzenia, percepcja barw (wpływ kolorów na zachowanie użytkowników), złudzenia optyczne, rozpoznawanie mowy.
3. **Interakcje człowiek-komputer (HCI)**: modele interakcji, teoria przetwarzania informacji.
4. **Projektowanie interfejsów użytkownika**: projektowanie User Experience (UX design), projektowanie zorientowane na użytkownika - User Centered Design (UCD).
5. **Projektowanie interfejsów dla różnych platform**: rodzaje platform, wzorce projektowe aplikacji i interfejsów użytkownika, standardy opisu interfejsu i automatyczne generowanie interfejsów, integracja interfejsów z systemami backend: komunikacja z serwerem.
6. **Narzędzia i techniki tworzenia interfejsów użytkownika**: UX/UI design tools, prototypowanie.
7. **Projektowanie responsywnych i dostępnych interfejsów użytkownika**: elastyczność i adaptacyjność.
8. **Architektura informacji**: struktura interfejsów, nawigacja, organizacja treści.
9. **Wizualny design interfejsów**: kolor, typografia, ikony, kompozycja.
10. **Testowanie i ocena użyteczności interfejsów użytkownika**: metody testowania, badania użytkowników, pojęcie użyteczności IU, heurystyki Nielsena, kryteria oceny jakości interfejsu graficznego i multimedialnego, rodzaje testów (kwestionariusze, eye-tracking, web-usability).

11. **Trendy w projektowaniu interfejsów użytkownika**: dark mode, voice UI, interfejsy AR/VR.

Ćwiczenia laboratoryjne:

12. Podstawowe interfejsy użytkownika
13. Elementy interfejsu graficznego i złudzenia optyczne z wykorzystaniem HTML i CSS
14. Elementy multimedialne interfejsu z wykorzystaniem HTML i CSS
15. Narzędzia do generowania i wspierania budowy interfejsów użytkownika
16. Budowa modelu szkieletowego interfejsu użytkownika
17. Implementacja aplikacji w architekturach MVC i MVP
18. Wizualny design interfejsów
19. Responsywne strony WWW
20. Budowa dostępnych interfejsów użytkownika
21. Narzędzia do testowania dostępności cyfrowej
22. Badanie użyteczności interfejsu użytkownika
23. Obrona projektu

Literatura podstawowa:

1. J. Tidwell, C. Brewer, A. Valencia Projektowanie interfejsów. Sprawdzone wzorce projektowe, Helion 2012.
2. M. Kasperski, Anna Boguska-Torbicz Projektowanie stron WWW. Użyteczność w praktyce, Helion 2008.
3. J.Nielsen, Projektowanie funkcjonalnych serwisów internetowych, Helion 2003.

Literatura dodatkowa:

1. J. Spolsky, *Projektowanie interfejsu użytkownika: poradnik dla programistów* (Mikom, Warszawa, 2001).
2. Masarek K., Sikorski M. (red), Interfejs użytkownika Kansei w praktyce, Wydawnictwo PJWSTK, Warszawa 2006.
3. The Essential Guide to User Interface Design - An Introduction to GUI Design Principles and Techniques, Wilbert O. Galitz, Wiley Publishing, Inc. 2007.

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, ćwiczenia praktyczne. Zamieszczanie na stronach internetowych materiałów do zajęć laboratoryjnych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Weryfikowane w trakcie zajęć laboratoryjnych i realizacji zadania indywidualnego. Przykładowe pytanie: „Jak psychologia percepcji wpływa na projektowanie interfejsów użytkownika?”.
W_02	Weryfikowane w trakcie zajęć laboratoryjnych i realizacji zadania indywidualnego. Przykładowe pytanie: „Jakie metody testowania użyteczności są najczęściej stosowane i dlaczego?”.
W_03	Weryfikowane w trakcie zajęć laboratoryjnych i realizacji zadania indywidualnego. Przykładowe pytanie: „Jakie narzędzia można wykorzystać do tworzenia makiet interfejsu użytkownika?”.

W_04	Weryfikowane w trakcie zajęć laboratoryjnych i realizacji zadania indywidualnego. Przykładowe pytanie: „Jakie wyzwania niesie projektowanie interfejsów w technologii AR/VR?”.
U_01- U_03	Weryfikowane w trakcie zajęć laboratoryjnych oraz podczas oceny realizacji zadania indywidualnego.
U_04	Weryfikowane w trakcie zajęć laboratoryjnych oraz podczas oceny realizacji zadania indywidualnego. Przykładowe zadanie: „Przeprowadź analizę zgodności projektu z heurystykami Nielsena”.
K_01	Weryfikowane podczas prezentacji zadania indywidualnego przez studenta. W trakcie prezentacji student przedstawi mocne strony i ograniczenia swojego rozwiązania.
K_02	Weryfikowane podczas prezentacji zadania indywidualnego przez studenta. W trakcie prezentacji student przedstawi wykorzystane przez siebie rozwiązania i porówna je do popularnych i nowych rozwiązań.

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem z oceną. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim oraz z samodzielnie wykonanego zadania indywidualnego według schematu:

- Regularne zajęcia – 100 pkt.
- Obrona zadania indywidualnego – 60 pkt.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej 51% sumy punktów. Wymagane jest uzyskanie co najmniej połowy punktów z poszczególnych form aktywności studenta: regularne zajęcia – co najmniej 51 pkt., obrona indywidualnego zadania – co najmniej 31 pkt. Ocena końcowa z modułu (wystawiana po zaliczeniu wszystkich części składowych), w zależności od sumy uzyskanych punktów (maksymalnie 160 pkt.) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 80 punktów: niedostateczna (F),
- 81 – 96 punktów: dostateczna (E),
- 97 – 112 punktów: dostateczna plus (D),
- 113 – 128 punktów: dobra (C),
- 129 – 144 punktów: dobra plus (B),
- 145 – 160 punktów: bardzo dobra (A).

Poprawy:

- Poprawa wybranych laboratoriów w trakcie konsultacji. W jednym tygodniu można poprawić maksymalnie 2 ćwiczenia laboratoryjne.
- Poprawa zadania indywidualnego w sesji egzaminacyjnej przed drugim terminem egzaminu.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.

Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	14 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Wykonanie zadania indywidualnego	13 godz.
Sumaryczne obciążenie pracą studenta	75 godz.
Punkty ECTS za przedmiot	3 ECTS
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do ćwiczeń laboratoryjnych	32 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Wykonanie zadania indywidualnego	10 godz.
Sumaryczne obciążenie pracą studenta	75 godz.
Punkty ECTS za przedmiot	3 ECTS

Sylabus przedmiotu / modułu kształcenia	
Nazwa przedmiotu/modułu kształcenia:	**Rozproszone Bazy Danych**
Nazwa w języku angielskim:	Distributed Database
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:	trzeci
Semestr:	piąty
Liczba punktów ECTS:	4
Imię i nazwisko koordynatora przedmiotu:	dr Artur Niewiadomski
Imię i nazwisko prowadzących zajęcia:	dr Artur Niewiadomski, mgr Wojciech Nabiałek
Założenia i cele przedmiotu:	Studenci przystępujący do tego przedmiotu powinni posiadać podstawową wiedzę z zakresu baz danych, przetwarzania równoległego i rozproszonego, programowania i systemów operacyjnych . Celem kursu jest zapoznanie studentów z teoretycznymi zagadnieniami związanymi z rozproszonymi bazami danych, ich trendami rozwojowymi oraz modelami stosowanymi w przetwarzaniu rozproszonym

Symbol efektu	**Efekty uczenia się** **WIEDZA** **Student zna i rozumie:**	**Symbol efektu kierunkowego**
W_01	podstawowe zagadnienia z zakresu teoretycznych podstaw rozproszonych baz danych, wykorzystywanych modeli oraz ich projektowania	**K_W08**
W_02	podstawowe zagadnienia z zakresu standardów i norm z zakresu baz danych, orientuje się w obecnym stanie oraz najnowszych trendach rozwojowych z zakresu rozproszonych baz danych	**K_W06**
W_03	zagadnienia związane z podstawowymi modelami i problemami przetwarzania rozproszonego i równoległego, zna i rozumie modele systemów rozproszonych i techniki przetwarzania rozproszonego, w tym w szczególności w zakresie rozproszonych baz danych	**K_W07**

Symbol efektu	UMIEJĘTNOŚCI Student potrafi:	Symbol efektu kierunkowego
U_01	pozyskiwać informacje na temat rozproszonych baz danych z literatury i innych źródeł, w tym zwłaszcza internetowych; potrafi analizować, interpretować, porządkować, oceniać przydatność i użyteczność oraz agregować i integrować uzyskane informacje, a także wyciągać wnioski z ich treści i formułować opinie	**K_U01**
U_02	dokonać wstępnej oceny proponowanych rozwiązań i podejmowanych działań inżynierskich przy identyfikowaniu i formułowaniu specyfikacji zadań inżynierskich w dziedzinie baz danych oraz przy ich rozwiązywaniu	**K_U09**
U_03	zaprojektować, zaimplementować oraz przetestować system informatyczny o charakterze rozproszonej bazy danych	**K_U09, K_U10, K_U12**
U_04	ocenić przydatność dostępnych metod i narzędzi służących do rozwiązywania zadań inżynierskich w dziedzinie baz danych i rozwiązywać praktyczne zadania inżynierskie wymagające korzystania ze standardów i norm inżynierskich oraz stosowania technologii informatycznych, wykorzystując doświadczenie zdobyte w środowisku zajmującym się zawodowo działalnością inżynierską	**K_U10, K_U24**
Symbol efektu	**KOMPETENCJE SPOŁECZNE Student jest gotów do:**	**Symbol efektu kierunkowego**
K_01	podejmowania decyzji i krytycznej oceny własnych rozwiązań w rozwiązywaniu zadań projektowych z zakresu rozproszonych baz danych	**K_K01**
K_02	Jest gotów do uznania znaczenia wiedzy w rozwiązywaniu zadań projektowych z zakresu rozproszonych baz danych oraz krytycznie potrafi ocenić swoje działania	**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:

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

1. Bazy danych
2. Podstawy przetwarzania równoległego
3. Systemy operacyjne

lub znajomość literatury obowiązującej w tych przedmiotach.

Treści modułu kształcenia:

Wykłady:

1. Wprowadzenie do systemów rozproszonych. Podstawowe pojęcia związane z rozproszeniem. Rozproszenie zasobów. Główne aspekty rozproszenia baz danych. Przezroczystość rozproszenia. Współdziałanie. Heterogeniczność.

2. Rozproszona baza danych - podstawowe pojęcia, cele i zalety rozproszenia. Komunikacja: aplikacja - baza danych, dostęp do zbioru nazw usług. Lokalny zbiór nazw usług, katalogowa. Baza danych LDAP. Serwer nazw, adresowanie serwera. Zewnętrzny serwis katalogowy. Konfigurowanie lokalnego zbioru nazw usług. Konfigurowanie środowiska klienta.

3. Typy i reguły rozproszonych baz danych. Homogeniczność i heterogeniczność programowania. Stopień lokalnej autonomii. Federacyjny system baz danych i zarządzanie nim. Relacyjno-obiektowe bazy danych. Techniki sterowania współbieżnego. Reguły Date. Fragmentacja pozioma, pionowa i mieszana. Kryteria poprawności fragmentacji.

4. Architektura rozproszonych baz danych. Architektura rozproszonej bazy danych. Specjalizowane oprogramowanie sieciowe. Łącznik bazy danych, perspektywa, synonim, migawka. Nazewnictwo baz danych w sieci, domena i nazwa globalna. Architektura klient - serwer, mechanizm komunikacji między klientem a serwerem. Określenie jednostki komunikacji klient-serwer, funkcje po stronie klienta i po stronie serwera. Architektura klient - broker - serwer, broker - pośrednik w dostępie do odległych zasobów. Architektura odniesienia ANSI.

5. Federacyjne systemy baz danych. Zarządzanie hierarchiami elementów baz danych. Protokół drzewiasty. Sterowanie współbieżnością za pomocą znaczników czasowych. Sterowanie współbieżnością za pomocą walidacji. Tryby integrowania danych.

6. Podejścia do projektowania rozproszonych baz danych. Podejście top-down, bottom-up, ad-hoc. Podział schematu logicznego, metody. Podejście oparte o globalny schemat. Problem alokacji. Problematyka replikacji. Określanie jednostki replikacji, ilości replikowanych danych, momentu i sposobu odświeżania.

7. Komunikacja: aplikacja - baza danych. Szeregi i plany szeregowane. Szeregowalność kolizyjna. Zapewnienie atomowości rozproszonej. Zatwierdzanie dwufazowe. Odtwarzanie transakcji rozproszonych.

8. Przetwarzanie i optymalizacja zapytań rozproszonych. Transakcja rozproszona. Architektura zarządzania transakcjami rozproszonymi. Protokół 2PC, scentralizowany, zdecentralizowany i liniowy. Rodzaje optymalizacji poleceń, wybór optymalizatora i celu optymalizacji. Generowanie statystyk i algorytmy łączenia tabel. Wykonywanie zapytań rozproszonych. Filtrowanie, grupowanie i sortowanie danych z tabeli zdalnej. Łączenie tabel, wykorzystanie wskazówek w łączeniu tabel.

9. Replikacje. Odświeżanie replik. Migawka - perspektywa zmaterializowana, typ migawki. Dziennik migawki , definiowanie dziennika. Implementacja dziennika, fizyczne parametry składowania dziennika. Modyfikowanie i usuwanie dziennika. Grupa odświeżania.

10. Przetwarzanie transakcji. Szeregowalność i odtwarzalność. Szeregowalność perspektywiczna. Rozwiązywanie problemu zakleszczeń. Zatwierdzanie rozproszone. Transakcje o długim czasie trwania.

11. Partycjonowanie tabel i indeksów. Algorytmy partycjonowania danych, partycjonowanie tabel. Partycjonowanie bazujące na wartości, haszowe i hybrydowe. Fizyczne parametry składowania tabel partycjonowanych. Zarządzanie tabelami partycjonowanymi. Partycjonowanie indeksów, typy indeksów. Zarządzanie indeksami partycjonowanymi.

Ćwiczenia laboratoryjne:

1. Definiowanie łącznika bazy danych
2. Łącznik współdzielony. Perspektywy i synonimy.
3. Transakcje rozproszone.
4. Protokół zatwierdzania dwufazowego.

5. Awarie i odtwarzanie transakcji rozproszonych.
6. Migawka – perspektywa zmaterializowana.
7. Typy migawek. Dziennik migawki.
8. Grupy odświeżania.
9. Partycjonowanie danych.
10. Optymalizacja zapytań rozproszonych.
11. Obrona projektu
12. Zaliczenie laboratorium

Literatura podstawowa:

1. Bębel B., Wrembel R.; Oracle. Projektowanie rozproszonych baz danych; Wydawnictwo Helion,2003
2. Wrembel R., Bębel B., Oracle : projektowanie rozproszonych baz danych : wiedza niezbędna do projektowania oraz zarządzania rozproszonymi bazami danych. Wydawnictwo Helion, 2005
3. Loney K.: Oracle database 11g : kompendium administratora, Helion 2010

Literatura dodatkowa:

1. Andrzej Barczak, Dariusz Zacharczuk, Damian Pluta., Methods of optimization of distributed databases in oracle – part 1 w Studia Informatica: Systems and Information Technology - 2016, nr 1-2 (20), s. 5-15
2. Andrzej Barczak, Dariusz Zacharczuk, Damian Pluta., Methods of optimization of distributed databases in oracle – part 2 w Studia Informatica: Systems and Information Technology - 2017, nr 1(21),

Planowane formy/działania/metody dydaktyczne:

Wykład tradycyjny wspomagany technikami multimedialnymi, laboratoria – praktyczna praca na komputerze. Zamieszczanie na stronach internetowych materiałów do zajęć laboratoryjnych.

Sposoby weryfikacji efektów uczenia się osiąganych przez studenta:

Symbol efektu	Metody weryfikacji efektów uczenia się
W_01	Weryfikowane na kolokwium. Przykładowe pytanie: Wyjaśnij podstawowe pojęcia: przezroczystość rozproszenia, współdziałanie, heterogeniczność itd. • Omów główne aspekty rozproszenia bazy danych • Wymień i scharakteryzuj typy rozproszonych baz danych • Omów architekturę systemu rozproszonej bazy danych, • Omów wady i zalety rozproszonych baz danych
W_02	Weryfikowane na kolokwium. Przykładowe pytania: Omów główne aspekty rozproszenia bazy danych, Wymień i scharakteryzuj typy rozproszonych baz danych
W_03	Weryfikowane na kolokwium. Przykładowe pytania: Omów architekturę systemu rozproszonej bazy danych, Omów wady i zalety rozproszonych baz danych

U_01- U_04	Weryfikowane w trakcie zajęć laboratoryjnych oraz podczas oceny realizacji zadania indywidualnego. Przykładowe zadania: Przeprowadź optymalizację zapytań w systemie rozproszonej bazy danych. Przygotuj plan wykonania transakcji rozproszonej, Przeprowadź odświeżanie przyrostowe migawki
K_01	Weryfikowane podczas prezentacji zadania indywidualnego przez studenta. W trakcie prezentacji student przedstawi mocne strony i ograniczenia swojego rozwiązania.
K_02	Weryfikowane podczas prezentacji zadania indywidualnego przez studenta. W trakcie prezentacji student przedstawi wykorzystane przez siebie rozwiązania i porówna je do popularnych i nowych rozwiązań.

Forma i warunki zaliczenia:

Moduł kończy się zaliczeniem z oceną. Ocena końcowa jest wystawiana na podstawie zajęć laboratoryjnych i jednego kolokwium pisemnego przeprowadzonego na ostatnim wykładzie. Na zaliczenie laboratorium składają się oceny cząstkowe uzyskane na regularnych zajęciach z nauczycielem akademickim oraz z samodzielnie wykonanego zadania indywidualnego według schematu:

- Regularne zajęcia – 40 pkt.,
- Obrona zadania indywidualnego – 20 pkt.

Zajęcia laboratoryjne będą zaliczone w wypadku uzyskania co najmniej połowy punktów z poszczególnych form aktywności studenta: regularne zajęcia – co najmniej 20 pkt., obrona indywidualnego zadania – co najmniej 10 pkt. Na tej formie zajęć student może maksymalnie uzyskać 60 pkt.

Za pisemne kolokwium można na nim uzyskać do 40 pkt. Zaliczenie kolokwium jest możliwe po uzyskaniu co najmniej 20 pkt. Ocena końcowa z modułu (po zaliczeniu wszystkich części składowych), w zależności od sumy uzyskanych punktów (maksymalnie 100pkt.) jest następująca (w nawiasach ocena wg skali ECTS):

- 0 – 50 - niedostateczna (2,0) (F)
- 51 -60 - dostateczna (3,0) (E)
- 61 -70 - dostateczna plus (3,5) (D)
- 71 -80 - dobra (4,0) (C)
- 81 -90 - dobra plus (4,5) (B)
- 91 -100 - bardzo dobra (5,0).(A).

Poprawy: Dodatkowy termin kolokwium pisemnego oraz drugi termin zaliczenia laboratorium – w toku sesji egzaminacyjnej.

Bilans punktów ECTS:

Studia stacjonarne

Aktywność	Obciążenie studenta
Udział w wykładach	21 godz.
Udział w ćwiczeniach laboratoryjnych	24 godz.
Samodzielne przygotowanie się do zajęć i zaliczenia oraz praca nad zadaniem indywidualnym	52 godz.

Udział w konsultacjach godz. z przedmiotu	3 godz.
Sumaryczne obciążenie pracą studenta	100 godz.
Punkty ECTS za przedmiot	4 ECTS
Studia niestacjonarne	
Aktywność	Obciążenie studenta
Udział w wykładach	15 godz.
Udział w ćwiczeniach laboratoryjnych	15 godz.
Samodzielne przygotowanie się do zajęć i zaliczenia oraz praca nad zadaniem indywidualnym	67 godz.
Udział w konsultacjach godz. z przedmiotu	3 godz.
Sumaryczne obciążenie pracą studenta	100 godz.
Punkty ECTS za przedmiot	4 ECTS