SYLABUS

DOTYCZY CYKLU KSZTAŁCENIA 2026/2027 - 2029/2030

Rok akademicki 2026/2027

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	*programowanie obiektowe*
Kod przedmiotu*	
Nazwa jednostki prowadzącej kierunek	*Instytut Informatyki, Wydział Nauk Ścisłych i Technicznych*
Nazwa jednostki realizującej przedmiot	*Instytut Informatyki, Wydział Nauk Ścisłych i Technicznych*
Kierunek studiów	*sztuczna inteligencja*
Poziom studiów	*studia I stopnia*
Profil	*ogólnoakademicki*
Forma studiów	*stacjonarne*
Rok i semestr/y studiów	*rok I, semestr 2*
Rodzaj przedmiotu	*przedmiot kierunkowy*
Język wykładowy	*polski*
Koordynator	*dr inż. Wojciech Kozioł*
Imię i nazwisko osoby prowadzącej / osób prowadzących	*dr inż. Wojciech Kozioł*

***** - *opcjonalnie, zgodnie z ustaleniami w Jednostce*

1.1.Formy zajęć dydaktycznych, wymiar godzin i punktów ECTS

Semestr (nr)	Wykł.	Ćw.	Konw.	Lab.	Sem.	ZP	Prakt.	Inne (jakie?)	**Liczba pkt. ECTS**
2	45			45					6

1.2. Sposób realizacji zajęć

zajęcia w formie tradycyjnej

1.3 Forma zaliczenia przedmiotu (z toku)

wykład – egzamin, laboratoria – zaliczenie z oceną

2.WYMAGANIA WSTĘPNE

znajomość podstaw programowania w zakresie szkoły średniej

3. CELE, EFEKTY UCZENIA SIĘ, TREŚCI PROGRAMOWE I STOSOWANE METODY DYDAKTYCZNE

3.1 Cele przedmiotu

C1	Zapoznanie studentów z zagadnieniami dotyczącymi paradygmatu programowania zorientowanego obiektowo.
C2	Nauczenie studentów myślenia, projektowania i rozwiązywania problemów przy użyciu obiektów i relacji występujących pomiędzy obiektami. Nauczenie studentów tworzenia programów w języku Java.
C3	Zapoznanie studentów z językiem Java i środowiskiem do tworzenia aplikacji w języku Java.
C4	Zapoznanie studentów z zagadnieniami dotyczącymi tworzenia graficznych interfejsów użytkownika i dostępu do relacyjnej bazy danych w języku Java.
C5	Nabycie przez studenta umiejętności tworzenia aplikacji z graficznym interfejsem użytkownika w języku Java.
C6	Nabycie umiejętności tworzenia aplikacji umożliwiających dostęp do relacyjnych baz danych w języku Java.

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu	Odniesienie do efektów kierunkowych [1]
EK_01	Zna dobrze konstrukcje programistyczne i struktury danych występujące w języku Java oraz środowisko NetBeans lub IntelliJ IDEA.	K_W03, K_W07
EK_02	Ma wiedzę dotyczącą obiektowego paradygmatu programowania i jego zastosowania; rozumie takie pojęcia jak: klasa, klasa abstrakcyjna, interfejs, obiekt, hermetyzacja, dziedziczenie, polimorfizm.	K_W03, K_W07
EK_03	Ma wiedzę na temat tworzenia graficznych interfejsów użytkownika oraz interfejsów do łączenia się z relacyjnymi bazami danych i użycia ich w języku Java.	K_W03, K_W07
EK_04	Umie właściwie zaprojektować i tworzyć zarówno proste jak i bardziej złożone aplikacje w języku Java wykorzystując przy tym poznane standardowe biblioteki dostępne w środowisku Java. Potrafi dokonać ich analizy i ocenić poprawność działania utworzonej aplikacji.	K_U03, K_U09
EK_05	Umie zaprojektować i tworzyć w języku Java aplikacje z graficznym interfejsem użytkownika, które umożliwiają łączenie się z relacyjną bazą danych i wykorzystanie tej bazy do przechowywania danych dla realizowanej	K_U03, K_U09

[1] W przypadku ścieżki kształcenia prowadzącej do uzyskania kwalifikacji nauczycielskich uwzględnić również efekty uczenia się ze standardów kształcenia przygotowującego do wykonywania zawodu nauczyciela.

	aplikacji. Wykorzystuje w tym celu poznane biblioteki i narzędzia informatyczne. Potrafi utworzyć dokumentację dla utworzonej aplikacji.	

3.3 Treści programowe

A. Problematyka wykładu

Geneza języka Java. Zasada działania technologii Java.
Ogólna postać programu w języku Java. Praca w środowisku NetBeans.
Identyfikatory, typy, zmienne, wyrażenia, operacje wejścia-wyjścia i komentarze.
Przepływ sterowania programem i sposoby jego modyfikacji. Iteracja i rekurencja w języku Java.
Zmienne tekstowe i operacje na łańcuchach w języku Java.
Obiekty, klasy, pola i metody, konstruktory w języku Java.
Hermetyzacja składowych (w oparciu o język Java).
Dziedziczenie i kompozycja (w oparciu o język Java).
Polimorfizm (w oparciu o język Java).
Klasy abstrakcyjne i interfejsy (w oparciu o język Java).
Składowe statyczne klasy: pola statyczne, metody statyczne, inicjalizatory statyczne (w oparciu o język Java).
Wyjątki w języku Java.
Typy surowe i generyczne w języku Java.
Kolekcje surowe i generyczne w języku Java.
Strumienie i pliki w języku Java.
Graficzny interfejs użytkownika – biblioteka SWING.
Graficzny interfejs użytkownika – biblioteka JavaFX.
Łączenie się z relacyjnymi bazami danych poprzez JDBC.
Mapowanie obiektowo-relacyjne w języku Java przy użyciu Hibernate.
Wprowadzenie do JSP

B. Problematyka laboratoriów

Wprowadzenie do środowiska NetBeans/IntelliJ IDEA.
Tworzenie i uruchamianie prostych programów w środowisku NetBeans/IntelliJ IDEA.
Przepływ sterowania programem w języku Java – pętle, rekurencje. Wyrażenia warunkowe.
Operacje na zmiennych łańcuchowych w Javie.
Tworzenie klas i obiektów w języku Java.
Kapsułkowanie w języku Java.
Dziedziczenie i kompozycja w języku Java.
Polimorfizm w języku Java.
Klasy i metody abstrakcyjne oraz interfejsy w języku Java.
Składowe statyczne klasy w języku Java.
Rzucanie i obsługa wyjątków w języku Java.
Typy uogólnione w języku Java.
Wykorzystanie kolekcji w języku Java.
Obsługa strumieni w języku Java.
Tworzenie GUI w języku Java przy użyciu biblioteki Swing.

Tworzenie GUI w języku Java przy użyciu biblioteki JavaFX.
Obsługa relacyjnych baz danych w języku Java przy użyciu interfejsu JDBC.
Obsługa relacyjnych baz danych przy użyciu frameworka Hibernate.
Tworzenie prostych aplikacji przy użyciu JSP.

3.4 Metody dydaktyczne

Wykład: wykład z prezentacją multimedialną.

Laboratorium: tworzenie programów komputerowych w oparciu o treści zadań zawartych w konspektach laboratoryjnych, projekt praktyczny w postaci oprogramowania realizowanego głównie jako praca domowa w dużej mierze samodzielna, ale konsultowana.

4. METODY I KRYTERIA OCENY

4.1 Sposoby weryfikacji efektów uczenia się

Symbol efektu	Metody oceny efektów uczenia się (np.: kolokwium, egzamin ustny, egzamin pisemny, projekt, sprawozdanie, obserwacja w trakcie zajęć)	Forma zajęć dydaktycznych (w, ćw, ...)
EK_01	egzamin pisemny	w
EK_02	egzamin pisemny	w
EK_03	egzamin pisemny	w
EK_04	kolokwium	lab
EK_05	projekt	lab

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

LABORATORIUM (weryfikowane efekty: EK_04, EK_05)
Zaliczenie z laboratorium wystawiane jest na podstawie średniej z ocen wystawianych kolokwium praktycznego (weryfikowany efekt EK_04) oraz projektu (weryfikowany efekt EK_05).

Weryfikacja efektu EK_04 odbywa się na podstawie kolokwium praktycznego, obejmującego tworzenie programów w języku Java zgodnie z zakresem treści programowych - od podstawowych konstrukcji języka po elementy programowania obiektowego.
Ocena końcowa z laboratorium jest ustalana na podstawie jakości i zakresu poprawnie wykonanego kolokwium praktycznego.
Dopuszcza się stosowanie dwóch równoważnych sposobów oceniania:

- systemu punktowego, w którym wynik procentowy odpowiada ocenie według ustalonej skali,
- lub systemu zadaniowego (sekwencyjnego), w którym każde kolejne poprawnie wykonane zadanie odpowiada wyższemu poziomowi oceny.

Skala ocen – system punktowy:
dostateczny (3.0) – 51–60%, dostateczny plus (3.5) – 61–70%, dobry (4.0) – 71–80%, dobry plus (4.5) – 81–90%, bardzo dobry (5.0) – 91–100%

Skala ocen – system zadaniowy (sekwencyjny).
W przypadku kolokwium zbudowanego z zadań o rosnącym stopniu trudności, ocena ustalana jest według zasady sekwencyjnego osiągania kolejnych poziomów:
Poprawne wykonanie zadania 1: 3.0 (dostateczny)
Poprawne wykonanie całego zadania 1, z częściowym rozpoczęciem zadania 2: 3.5 (dostateczny plus)
Poprawne wykonanie zadań 1 i 2: 4.0 (dobry)
Poprawne wykonanie zadań 1 i 2, z częściową realizacją zadania 3: 4.5 (dobry plus)
Poprawne wykonanie zadań 1, 2 i 3: 5.0 (bardzo dobry).

Weryfikacja efektu EK_05 odbywa się na podstawie zaliczenie projektu. W ramach zaliczenia projektu oceniane są:

- złożoność bazy danych i poprawność schematu bazy danych projektu (maksymalnie: 10 pkt.),
- struktura aplikacji, jej złożoność i podział na pakiety, oraz jakość utworzonego kodu aplikacji (maksymalnie: 20 pkt.),
- użyte technologie – wykorzystanie w projekcie bardziej wymagających technologii tj. JavaFX i Hibernate jest wyżej punktowane niż użycie SWING i JDBC. Punktacja za użyte technologie: JavaFX: 10 pkt., Hibernate: 10 pkt., SWING: 5 pkt., JDBC 5 pkt. (maksymalnie: 20 pkt.),
- organizacja interfejsu użytkownika: układ, złożoność, stylizacja, intuicyjność (maksymalnie: 20 pkt.),
- jakość utworzonej dokumentacji projektu (maksymalnie: 10 pkt.),
- obrona projektu, w ramach której student prezentuje swój projekt, sprawdzająca jak student orientuje się w kodzie utworzonego projektu (maksymalnie: 20 pkt).

Ocena z laboratorium jest oceną uzyskaną z zaliczenia projektu wg punktacji: 0 ÷ 50 pkt. - **niedostateczny**, 51 ÷ 60 pkt. - **dostateczny**, 61 ÷ 70 pkt. - **dostateczny plus**, 71 ÷ 80 pkt. - **dobry**, 81 ÷ 90 pkt. - **dobry plus**, 91 ÷ 100 pkt. - **bardzo dobry**.

<u>WYKŁAD (weryfikowane efekty: EK_01, EK_02, EK_3):</u>

1. Pozytywny wynik egzaminu pisemnego z zakresu materiału prezentowanego na wykładzie, przy czym:

- Student otrzymuje z egzaminu ocenę **dostateczny**, jeśli wykona co najmniej 50% zadań egzaminacyjnych dotyczących efektu EK_01 i co najmniej 50% zadań egzaminacyjnych dotyczących efektu EK_02 oraz co najmniej 50% zadań egzaminacyjnych dotyczących efektu EK_03.
- Student otrzymuje z egzaminu ocenę **dostateczny plus**, jeśli średnia wykonanych przez niego zadań egzaminacyjnych dotyczących efektów EK_01, EK_02, EK_03 wynosi co najmniej 60%, przy czym każdy z efektów musi być zaliczony na co najmniej 50%.
- Student otrzymuje z egzaminu ocenę **dobry**, jeśli średnia wykonanych przez niego zadań egzaminacyjnych dotyczących efektów EK_01, EK_02, EK_03 wynosi co najmniej 70%, przy czym każdy z efektów musi być zaliczony na co najmniej 50%.
- Student otrzymuje z egzaminu ocenę **dobry plus**, jeśli średnia wykonanych przez niego zadań egzaminacyjnych dotyczących efektów EK_01, EK_02, EK_03 wynosi co najmniej 80%, przy czym każdy z efektów musi być zaliczony na co najmniej 50%.
- Student otrzymuje z egzaminu ocenę **bardzo dobrą**, jeśli średnia wykonanych przez

niego zadań egzaminacyjnych dotyczących efektów EK_01, EK_02, EK_03 wynosi co najmniej 90%, przy czym każdy z efektów musi być zaliczony na co najmniej 50%.

Student przystępujący do egzaminu poprawkowego jest zobowiązany do poprawy tylko tych efektów, których nie zaliczył w terminie podstawowym.

W procedurze potwierdzania efektów uczenia się uzyskanych w procesie uczenia się poza systemem studiów istnieje możliwość zaliczenia wykładu z pierwszej części przedmiotu na podstawie posiadanych certyfikatów.
Sposób uznawania osiągnięcia efektów na podstawie certyfikatów:
Java SE 8 Programmer I – zalicza efekty EK_01 i EK_02 z pierwszego semestru przedmiotu (w odniesieniu do języka Java) na ocenę dostateczną.
Java SE 8 Programmer I + Java SE 8 Programmer II – zalicza efekty EK_01 i EK_02 z pierwszego semestru przedmiotu (w odniesieniu do języka Java) na ocenę dobrą.
Java SE 11 Developer – zalicza efekty EK_01 i EK_02 z pierwszego semestru przedmiotu (w odniesieniu do języka Java) na ocenę bardzo dobrą.
Java SE 17 Developer – zalicza efekty EK_01 i EK_02 z pierwszego semestru przedmiotu (w odniesieniu do języka Java) na ocenę bardzo dobrą.

5. CAŁKOWITY NAKŁAD PRACY STUDENTA POTRZEBNY DO OSIĄGNIĘCIA ZAŁOŻONYCH EFEKTÓW W GODZINACH ORAZ PUNKTACH ECTS

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny z harmonogramu studiów	90
Inne z udziałem nauczyciela akademickiego (udział w konsultacjach, egzaminie)	2
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	70
SUMA GODZIN	162
SUMARYCZNA LICZBA PUNKTÓW ECTS	**6**

** Należy uwzględnić, że 1 pkt ECTS odpowiada 25-30 godzin całkowitego nakładu pracy studenta.*

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	–
zasady i formy odbywania praktyk	–

7. LITERATURA

Literatura podstawowa:
1. B. Eckel: Thinking in Java: edycja polska, Gliwice, Helion, 2006
2. C. S. Horstmann: Java: podstawy, Gliwice, Helion, 2023
3. C. S. Horstmann: Java™: techniki zaawansowane, Gliwice, Helion, 2020
4. M. Suwała : Java: teoria w praktyce, Gliwice, Helion, 2024
5. K. Sierra, B. Bates, T. Gee: Java: przewodnik po praktycznym programowaniu w Javie,

Gliwice, Helion, 2023
6. M. Lis: Java. Ćwiczenia praktyczne. Wyd. 3, Gliwice, Helion, 2011
7. M. Lis, Java: praktyczny kurs, Gliwice, Helion, 2015
8. W. Rychlicki: Programowanie w języku Java: zbiór zadań z (p)odpowiedziami, Gliwice, Helion, 2012
9. S. Perkins : Hibernate search : skuteczne wyszukiwanie, Gliwice, Helion, 2014
10. M. Hall, L. Brown: Java Servlet i JavaServer Pages. T. 1, Gliwice, Helion, 2006

Literatura uzupełniająca:

1. M. Lis: Java: ćwiczenia zaawansowane, Gliwice, Helion, 2012
2. B. J. Evans, D. Flanagan: Java w pigułce, Gliwice, Helion, 2015
3. J. Bloch: Java : efektywne programowanie. Wyd. 3, Gliwice, Helion, 2022
4. H. Schildt: Java™: przewodnik dla początkujących: twórz, kompiluj i uruchamiaj nowoczesne programy w Javie, Gliwice, Helion 2024
5. H. Schild: Java: kompendium programisty, Gliwice, Helion, 2012
6. E. Jendrock: Java EE 6: zaawansowany przewodnik, Gliwice, Helion, 2013