Załącznik nr 1.5 do Zarządzenia Rektora UR nr 61/2025

SYLABUS

DOTYCZY CYKLU KSZTAŁCENIA *2025-2029*

(skrajne daty)

Rok akademicki 2025/2026

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	Podstawy programowania
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	Informatyka i ekonometria
Poziom studiów	studia pierwszego stopnia
Profil	praktyczny
Forma studiów	studia stacjonarne
Rok i semestr/y studiów	rok I, sem. I
Rodzaj przedmiotu	przedmiot podstawowy
Język wykładowy	polski
Koordynator	dr inż. Michał Kępski
Imię i nazwisko osoby prowadzącej / osób prowadzących	dr inż. Michał Kępski, mgr inż. Marcin Pater, mgr inż. Wojciech Gałka

** -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**
1	15			30					3

1.2. Sposób realizacji zajęć

☒ zajęcia w formie tradycyjnej

☐ zajęcia realizowane z wykorzystaniem metod i technik kształcenia na odległość

1.3 Forma zaliczenia przedmiotu (z toku) (egzamin, zaliczenie z oceną, zaliczenie bez oceny)

Zaliczenie z oceną

2.WYMAGANIA WSTĘPNE

Umiejętność programowania na poziomie szkoły średniej.

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

3.1 Cele przedmiotu

C1	Zapoznanie studentów z językiem programowania wysokiego poziomu Python jako językiem skryptowym o szerokim zastosowaniu i rozbudowanej bibliotece standardowej.
C2	Nabycie przez studentów umiejętności tworzenia praktycznych skryptów z wykorzystaniem języka skryptowego.

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu	Odniesienie do efektów kierunkowych
EK_01	Student zna składnię i struktury danych języka Python oraz przeznaczenie wybranych bibliotek. Rozumie różnice między językami interpretowanymi a kompilowanymi. Zna mechanizmy programowania obiektowego w tym języku.	K_W02, K_W03
EK_02	Potrafi samodzielnie programować w języku Python, tworząc programy złożone z kilku funkcji, wykorzystujące obiekty różnych klas, edytujące pliki. Potrafi definiować własne klasy, korzystać z mechanizmów obiektowości, tworzyć kod obsługujący wystąpienia wyjątków.	K_U01, K_U04,
EK_03	Potrafi wykorzystywać wybrane zaawansowane zagadnienia języka Python. Potrafi wykorzystywać wirtualne środowiska i narzędzia do zarządzania pakietami. Potrafi korzystać z wybranych bibliotek do i obliczeń i prezentacji danych.	K_U06

3.3 Treści programowe

A. Problematyka wykładu

1. Wstęp do języka Python. Interpreter języka Python. Składnia języka. Typy i operatory. Instrukcje sterujące.
2. Składnia języka (ciąg dalszy). Definiowanie funkcji. Zmienna, obiekt, referencja. Przestrzenie nazw.
3. Moduły i pakiety. Obsługa wyjątków, asercje. Logowanie. Definiowanie klas – podstawy.
4. Klasy, obiekty, programowanie obiektowe. Iteratory. Generatory.
5. Środowisko programisty Pythona. Zarządzanie pakietami i ścieżka wyszukiwania modułów. Wirtualne środowiska. Narzędzia do zarządzania instalacjami: conda, pip.
6. Narzędzia wbudowane – biblioteka standardowa języka Python. Typowe zadania w języku Python: manipulacja plikami, katalogami, wczytywanie danych. Prosta komunikacja sieciowa. Działania w internecie.
7. Biblioteka Numpy. Biblioteka Scipy. Matplotlib, Jupyter Notebook.

B. Problematyka ćwiczeń, konwersatoriów, laboratoriów, zajęć praktycznych

1. **Wprowadzenie do Pythona.** Instalacja środowiska, Jupyter Notebook / VS Code, podstawowe typy danych i operatory.
2. **Zmienne i instrukcje sterujące.** Typowanie dynamiczne, instrukcje if/elif/else, operatory porównania.
3. **Pętle i podstawowe algorytmy.** Pętle for/while, funkcje wbudowane (range, enumerate, zip), suma/średnia, minimum/maksimum.
4. **Funkcje (cz. 1).** Definiowanie funkcji, argumenty i wartości zwracane, parametry domyślne, prosta rekurencja.
5. **Funkcje (cz. 2).** Zasięg zmiennych, funkcje anonimowe (lambda), funkcje wyższego rzędu (map, filter, reduce).
6. **Struktury danych.** Listy, krotki, słowniki, zbiory, operacje na kolekcjach, list/dict comprehensions.

7. **Obsługa plików i wyjątków.** Otwieranie plików (with open), try/except/finally, proste przetwarzanie danych tekstowych/CSV.
8. **Moduły i biblioteka standardowa.** Import własnych modułów, wybrane moduły: math, random, datetime, os, sys.
9. **Podstawy NumPy.** Tablice NumPy, indeksowanie, operacje wektorowe i macierzowe, podstawowe statystyki.
10. **Podstawy matplotlib w Jupyter Notebook.** Wykresy liniowe, słupkowe i punktowe, opisy osi, wykresy wieloserii.
11. **Programowanie obiektowe.** Klasy, obiekty, metody, atrybuty instancji i klasy, dziedziczenie i nadpisywanie metod.
12. **Projekt podsumowujący.** Integracja: funkcje, pliki, NumPy, matplotlib – mini-projekt analizy i wizualizacji danych.

3.4 Metody dydaktyczne

Wykład: wykład z prezentacją multimedialną.
Laboratorium: rozwiązywanie zadań programistycznych.

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	Zaliczenie (test)	wykład
Ek_ 02, Ek_ 03	Kolokwium, obserwacja w trakcie zajęć	laboratorium

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

Wykład – efekt EK_01:

Test zaliczeniowy: Aby otrzymać ocenę pozytywną, student musi odpowiedzieć na co najmniej 50% pytań dotyczących składni, struktur danych, konstrukcji programistycznych, w tym tych realizujących paradygmat programowania obiektowego, języka Python oraz wybranych bibliotek tego języka.

Zajęcia laboratoryjne – efekty EK_02, EK_03:

Zaliczenie z oceną: na podstawie kolokwiów i aktywności podczas zajęć. Prowadzący na podstawie oceny aktywności podczas zajęć może podwyższyć lub obniżyć końcową ocenę o 0.5 stopnia.

Na ocenę dostateczny – student zna składnię języka Python i potrafi się nią posługiwać w stopniu średnim. Zna i potrafi wykorzystać podstawowe typy danych i instrukcje sterujące. Student potrafi napisać prosty skrypt w języku Python i uruchomić go z linii poleceń. Zna i potrafi poprawnie zaimportować elementy biblioteki standardowej. Potrafi definiować funkcje wykonujące zestaw instrukcji i zwracające wynik. Potrafi wywołać zdefiniowane funkcje. Potrafi korzystać z gotowych klas – tworzyć obiekty i wywoływać ich metody.

Na ocenę dobry – jw., a ponadto student zna składnię języka Python w stopniu dobrym, potrafi definiować własne klasy, poprawnie implementować zależności dziedziczenia.

Na ocenę bardzo dobry – jw., a ponadto student wykazuje się umiejętnością płynnej pracy z językiem Python

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	45
Inne z udziałem nauczyciela akademickiego (udział w konsultacjach, egzaminie)	2
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	55
SUMA GODZIN	102
SUMARYCZNA LICZBA PUNKTÓW ECTS	4

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	-
zasady i formy odbywania praktyk	-

7. LITERATURA

Literatura podstawowa:

1. Python 3: kompletne wprowadzenie do programowania / Mark Summerfield ; [tł. z ang. Robert Górczyński]. - Wyd. 2. - Gliwice: Helion, cop. 2010.

Literatura uzupełniająca:

1. Mark Lutz, David Ascher: Python. Wprowadzenie, 2002, Helion
2. Mark Lutz: Python. Wprowadzenie. Wydanie IV, 2010, Helion
3. Oficjalna dokumentacja języka Python: http://www.python.org/doc/

Akceptacja Kierownika Jednostki lub osoby upoważnionej