Opis **zajęć (sylabus)**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nazwa zajęć: | | Programowanie zaawansowane | | | | | | | **ECTS** | | **2** |
| Nazwa zajęć w j. angielskim: | | Advanced Programming | | | | | | | | | |
| Zajęcia dla kierunku studiów: | | **Informatyka i Ekonometria** | | | | | | | | | |
|  | |  | | | | | | | | | |
| Język wykładowy: | | polski | | | | Poziom studiów: | | studia I stopnia | | | |
| Forma studiów: |  stacjonarne   niestacjonarne | Status zajęć: |  podstawowe   kierunkowe |  obowiązkowe   do wyboru | | Numer semestru: ……1….. | |  semestr zimowy  semestr letni | | | |
|  |  | Rok akademicki, od którego obowiązuje opis (rocznik): | | | | 2020/2021 | Numer katalogowy: | **ZIM-IE-2Z-01Z-8** | | | |
|  | | | | | | | | | | | |
| Koordynator zajęć: | | **dr Marcin Ziółkowski** | | | | | | | | | |
| Prowadzący zajęcia: | | **dr Marcin Ziółkowski** | | | | | | | | | |
| Jednostka realizująca: | | **Instytut Informatyki Technicznej, Katedra Systemów Informatycznych** | | | | | | | | | |
| Jednostka zlecająca: | | **Wydział Zastosowań Informatyki i Matematyki** | | | | | | | | | |
| Założenia, cele i opis zajęć: | | Celem kursu jest podniesienie umiejętności programowania w wybranym języku. Studenci będą ćwiczyli umiejętności programistyczne w paradygmatach: proceduralnym, obiektowym oraz funkcyjnym. Treści obejmują m.in. instrukcje warunkowe, pętle iteracyjne, sekwencyjne struktury danych, funkcje, programy z argumentami pozycyjnymi, obsługę wyjątków, budowanie własnych klas oraz inne zaawansowane techniki obiektowe m.in.: indeksatory, przeciążanie operatorów, delegaty, interfejsy czy klasy abstrakcyjne, a także wykorzystanie specjalistycznych modułów wybranego języka programowania. Ilustracja omawianych zagadnień oparta będzie o język programowania Python.  Tematyka ćwiczeń laboratoryjnych:   * Środowisko uruchomieniowe języka Python. Praca w trybie interaktywnym oraz skryptowym w różnych systemach operacyjnych  Wprowadzanie i wyprowadzanie danych, instrukcja warunkowa, instrukcja iteracyjna WHILE, instrukcje iteracyjne FOR, typy wartościowe i typy referencyjneSekwencje – łańcuchy znaków, krotki, listy, zbiory i słownikiFunkcje i ich wykorzystanie do tworzenia własnych modułów i pakietów, programy z argumentami pozycyjnymiObsługa wyjątków, operacje na plikachPodstawy programowania obiektowego: klasy, metody, obiekty i pola, konstruktory i atrybuty klas, polimorfizm i dziedziczenie, modyfikatory dostępu do pól oraz metodTworzenie interfejsów graficznych GUI jako przykład wykorzystania paradygmatu programowania obiektowego. Wykorzystanie elementów graficznych i multimedialnychZaawansowane techniki obiektowe m.in.: indeksatory, przeciążanie operatorów, delegaty, interfejsy czy klasy abstrakcyjneElementy programowania funkcyjnego (iteratory, generatory)Wykorzystanie specjalistycznych modułów wybranego języka programowania (np. scipy oraz numpy) do obliczeń w ekonometrii (m.in. operacje na macierzach i wektorach, analiza statystyczna dużych zbiorów danych) | | | | | | | | | |
| Formy dydaktyczne, liczba godzin: | | 1. ćwiczenia w laboratorium komputerowym; liczba godzin ...30...; | | | | | | | | | |
| Metody dydaktyczne: | | dyskusja i rozwiązywanie problemów, konsultacje | | | | | | | | | |
| Wymagania formalne  i założenia wstępne: | | Wymagana jest elementarna umiejętność programowania w jednym z języków wysokiego poziomu oraz elementarna umiejętność logicznej analizy problemów. | | | | | | | | | |
| Efekty uczenia się: | | Wiedza:  Student:  1 – zna sposoby programowania strukturalnego, obiektowego oraz funkcyjnego  2 – rozumie podstawowe zasady tworzenia programów w oparciu o obiektowy paradygmat programowania | | | Umiejętności:  1 - potrafi pisać proste programy w oparciu o strukturalny paradygmat programowania oraz tworzyć i wykorzystywać abstrakcyjne typy danych (sekwencje) oraz funkcje  2 - potrafi tworzyć proste ciągi klas dziedziczących po sobie, wykorzystując również zaawansowane techniki obiektowe  3 - potrafi tworzyć publiczne oraz prywatne metody i pola  4 - potrafi wychwytywać i obsługiwać wyjątki oraz wykonywać operacje na plikach  5 – potrafi napisać złożoną aplikację wykorzystującą obiektowy paradygmat programowania, interfejs GUI oraz moduły specjalistyczne | | | | | Kompetencje:  ……………………..  …………………….. | |
| Sposób weryfikacji efektów uczenia się: | | Zaliczenie z przedmiotu odbywa się na podstawie pozytywnego zaliczenia (na co najmniej 50%) następujących rygorów:  - test wiedzy ze znajomości podstaw teoretycznych - waga 0,25  - kolokwium praktyczne (polegające na samodzielnym napisaniu 4-5 programów, których tematy zostaną zaprezentowane przez prowadzącego) - waga 0,5  - projekt grupowy polegający na stworzeniu aplikacji uwzględniającej paradygmat programowania obiektowego z wykorzystaniem interfejsu graficznego GUI - waga 0,25 | | | | | | | | | |
| Forma dokumentacji osiągniętych efektów uczenia się: | | Test wiedzy (archiwum z portalu e-learningowego) Pliki kolokwium praktycznego (archiwum z portalu e-learningowego)  Pliki projektu grupowego (archiwum z portalu e-learningowego) | | | | | | | | | |
| Elementy i wagi mające wpływ  na ocenę końcową: | | Ocena końcowa jest obliczana w oparciu o następujące składowe:  - test wiedzy (25%)  - kolokwium praktyczne (50 %)  - projekt grupowy (25%) | | | | | | | | | |
| Miejsce realizacji zajęć: | | Ćwiczenia laboratoryjne – laboratorium komputerowe | | | | | | | | | |
| Literatura podstawowa i uzupełniająca:  Literatura podstawowa:  1) Michael Dawson: Python dla każdego. Podstawy programowania. Od zera do bohatera, Helion 2014.  2) Dive into Python – internetowy podręcznik programowania w Pythonie .  Literatura uzupełniająca:  1) Andy Ju An Wang, Kai Qian: Component-Oriented Programming, John Wiley & Sons, 2005.  2) Ian Sommerville: Inżynieria oprogramowania, WNT 2003. | | | | | | | | | | | |
| UWAGI  Minimalna liczba punktów konieczna do zaliczenia każdego rygoru: 50% | | | | | | | | | | | |

Wskaźniki ilościowe charakteryzujące moduł/przedmiot:

|  |  |
| --- | --- |
| Szacunkowa sumaryczna liczba godzin pracy studenta (kontaktowych i pracy własnej) niezbędna dla osiągnięcia zakładanych dla zajęć efektów uczenia się - na tej podstawie należy wypełnić pole ECTS: | **60 h** |
| Łączna liczba punktów ECTS, którą student uzyskuje na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich lub innych osób prowadzących zajęcia: | **1 ECTS** |

Tabela zgodności kierunkowych efektów uczenia się z efektami przedmiotu:

|  |  |  |  |
| --- | --- | --- | --- |
| kategoria efektu | Efekty uczenia się dla zajęć: | Odniesienie do efektów dla programu studiów dla kierunku | Oddziaływanie zajęć na efekt kierunkowy\*) |
| Wiedza 1 | zna sposoby programowania strukturalnego, obiektowego oraz funkcyjnego | K\_W06 / P7S\_WG  K\_W12 / P7S\_WG | 3  2 |
| Wiedza 2 | rozumie podstawowe zasady tworzenia programów w oparciu o obiektowy paradygmat programowania | K\_W08 / P7S\_WG | 3 |
| Umiejętności 1 | potrafi pisać proste programy w oparciu o strukturalny paradygmat programowania oraz tworzyć i wykorzystywać abstrakcyjne typy danych (sekwencje) oraz funkcje | K\_U13 / P7S\_UW | 1 |
| Umiejętności 2 | potrafi tworzyć proste ciągi klas dziedziczących po sobie, wykorzystując również zaawansowane techniki obiektowe | K\_U13 / P7S\_UW | 1 |
| Umiejętności 3 | potrafi tworzyć publiczne oraz prywatne metody i pola | K\_U13 / P7S\_UW | 1 |
| Umiejętności 4 | potrafi wychwytywać i obsługiwać wyjątki oraz wykonywać operacje na plikach | K\_U13 / P7S\_UW | 1 |
| Umiejętności 5 | potrafi napisać złożoną aplikację wykorzystującą obiektowy paradygmat programowania, interfejs GUI oraz moduły specjalistyczne | K\_U13 / P7S\_UW | 1 |
| Kompetencje - |  |  |  |
| Kompetencje - |  |  |  |

\*)

3 – zaawansowany i szczegółowy,

2 – znaczący,

1 – podstawowy,