



KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu

Techniki programowania obiektowego [S1MNT1>D-TPO]

Przedmiot

Kierunek studiów

Matematyka nowoczesnych technologii

Rok/Semestr

2/4

Studia w zakresie (specjalność)

–

Profil studiów

ogólnoakademicki

Poziom studiów

pierwszego stopnia

Język oferowanego przedmiotu

polski

Forma studiów

stacjonarne

Wymagalność

obieralny

Liczba godzin

Wykład

15

Laboratorium

30

Inne

0

Ćwiczenia

0

Projekty/seminaria

15

Liczba punktów ECTS

4,00

Koordynatorzy

dr hab. Grzegorz Oleksik

grzegorz.oleksik@put.poznan.pl

Wykładowcy

Wymagania wstępne

Wiedza i umiejętności z kursów Technologie Informacyjne I i II, Wstęp do programowania oraz Programowa- nie z elementami uczenia maszynowego. Umiejętność efektywnego samokształcenia. Znajomość ograniczeń własnej wiedzy oraz zrozumienie potrzeby dalszego kształcenia.

Cel przedmiotu

Przedmiot ma na celu opanowanie przez studenta podstawowej wiedzy i umiejętności z zakresu programowania obiektowego na przykładzie języka Python.

Przedmiotowe efekty uczenia się

Wiedza:

- zna i rozumie przynajmniej jeden język programowania lub środowisko programowania albo pakiet oprogramowania [K_W07(P6S_WG)];
- zna i rozumie zależności między matematyką a nowoczesnymi technologiami [K_W05(P6S_WG)];
- zna i rozumie zaawansowanym stopniu wybranych dziedzin matematyki oraz szczegółową wiedzę dotyczącą zastosowań metod i narzędzi matematycznych w naukach inżyniersko-technicznych [K_W 09(P6S_W G)].

Umiejętności:

- potrafi skonstruować algorytm rozwiązania prostego zadania inżynierskiego oraz zaimplementować i przetestować go w wybranym środowisku programistycznym [K_U04(P6S_UW)];
- potrafi zastosować nowoczesne technologie do rozwiązywania problemów matematycznych oraz inżynierii no-technicznych [K_U05(P6S_UW)];
- potrafi zastosować narzędzia matematyczne do wspomagania i rozwoju nowoczesnych technologii wykorzystywanych w naukach inżynierii no-technicznych [K_U06(P6S_UW)];
- potrafi zgodnie z ogólnymi wymogami i dokumentacją techniczną eksploatować urządzenia, narzędzia itp.; umie stosować zasady bezpieczeństwa i higieny pracy [K_U11(P6S_UW)];
- potrafi pracować indywidualnie i w zespole oraz współdziałać z innymi osobami; umie oszacować czas potrzebny na realizację zleconego zadania; potrafi opracować i zrealizować harmonogram prac zapewniający dotrzymanie terminu. [K_U17(P6S_UU)];
- potrafi samodzielnie planować i realizować samokształcenie w celu podnoszenia i aktualizacji swoich kompetencji [K_U17(P6S_UU)].

Kompetencje społeczne:

- jest gotów do krytycznej oceny poziomu swojej wiedzy w odniesieniu do prowadzonych badań w naukach ścisłych i przyrodniczych oraz naukach inżynierii no-technicznych [K_K01(P6S_KK)];
- jest gotów do pogłębiania i poszerzania wiedzy do rozwiązywania nowopowstałych problemów technicznych [K_K02(P6S_KK)].

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Wykłady: wiedza jest weryfikowana na teście pisemnym;

Laboratoria: wiedza i umiejętności weryfikowane są na podstawie oceny z kolokwium programistycznego;

Projekty/seminaria: wiedza i umiejętności są weryfikowane na podstawie oceny realizacji projektu, jego funkcjonalności i celów;

Skala ocen: % -49% - NDST, 50%-59% - DST, 60%-69% - DST+, 70%-79% - DB, 80%-89% - DB+, 90%-100% - BDB.

Treści programowe

- wstęp do programowania obiektowego, niezależnie od konkretnego języka programowania
- hermetyzacja danych
- dziedziczenie
- polimorfizm
- metody specjalne
- dekoratory
- graficzny interfejs użytkownika
- wzorce projektowe

Tematyka zajęć

Wykłady:

- wstęp do programowania obiektowego w języku Python;
- ochrona danych: hermetyzacja i modyfikatory dostępu;
- wstęp do diagramów UMLa;
- praca z obiektem: tworzenie, inicjalizacja, usuwanie;
- zagadnienie dziedziczenia oraz pojęcie polimorfizmu;
- klasy i metody abstrakcyjne;
- biblioteka tkinter
- obsługa wyjątków;
- metody specjalne w klasach:
 - `__len__`;
 - `__str__`! i `__repr__`;
 - `__add__`!, `__sub__`, `__mul__`, `__truediv__`;
 - `__call__`!;
- wprowadzenie do dekoratorów.

- wzorce projektowe

Laboratorium:

- tworzenie własnych klas z metodami - podstawy
- tworzenie klas bazowych i pochodnych
- tworzenie klas z metodami specjalnymi
- używanie dekoratorów w metodach klas
- tworzenie własnego GUI przy użyciu biblioteki tkinter
- stosowanie wzorców projektowych (MVC, Strategy)

Metody dydaktyczne

Wykłady: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, rozwiązywanie zadań, pokaz multimedialny, demonstracja;

Laboratoria: rozwiązywanie praktycznych problemów, dyskusja, praca indywidualna lub zespołowa;

Projekty/seminaria: prezentacja multimedialna, dyskusja, praca indywidualna lub zespołowa, rozwiązywanie praktycznych problemów.

Literatura

Podstawowa:

- Lutz, M., Python. Wprowadzenie, Helion, najnowsze wydanie;
- Steven F. Lott, Dusty Phillips: Programowanie zorientowane obiektowo w Pythonie. Helion 2023.

Uzupełniająca:

- Brian, J., Beazley, J., Python. Receptury, Helion, najnowsze wydanie;
- Eric, M., Python. Instrukcje dla Programisty, Helion, najnowsze wydanie;
- http://wazniak.mimuw.edu.pl/index.php?title=Programowanie_obiektowe;
- <https://analitik.edu.pl/programowanie-objektowe-w-python/>

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	100	4,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	60	2,50
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwium/egzaminu, wykonanie projektu)	40	1,50