KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Programowanie obiektowe w Python [S1DSwB1>POwP]

Przedmiot

Kierunek studiów
Data Science w biznesie

Rok/Semestr
1/2

Studia w zakresie (specjalność)
–

Profil studiów
ogólnoakademicki

Poziom studiów
pierwszego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
30

Laboratorium
0

Inne
0

Ćwiczenia
30

Projekty/seminaria
0

Liczba punktów ECTS

5,00

Koordynatorzy

dr Grzegorz Nowak
grzegorz.nowak@put.poznan.pl

dr inż. Marcin Nowak
marcin.nowak@put.poznan.pl

Wykładowcy

Wymagania wstępne

Studenci powinni znać podstawy programowania w Pythonie, w tym składnię języka, typy danych, struktury danych, instrukcje sterujące, funkcje i programowanie modularne. Wymagana jest umiejętność pracy z plikami, podstawowymi bibliotekami oraz podstawy debugowania i testowania kodu.

Cel przedmiotu

Celem przedmiotu jest zapoznanie studentów z paradygmatem programowania obiektowego w języku Python. Studenci nauczą się tworzyć i wykorzystywać klasy oraz obiekty, stosować zasady enkapsulacji, dziedziczenia i polimorfizmu, a także projektować i implementować struktury zgodne z podejściem obiektowym. Kurs rozwija umiejętność organizacji kodu w większe moduły, pracy z zaawansowanymi mechanizmami Pythona oraz korzystania z wzorców projektowych, co przygotowuje do efektywnego tworzenia i utrzymania aplikacji.

Przedmiotowe efekty uczenia się

Wiedza:
1. Wyjaśnia zasady programowania obiektowego (OOP) w Pythonie, w tym koncepcję klas, obiektów, metod oraz różnice względem programowania proceduralnego [DSB1_W02].
2. Charakteryzuje mechanizmy hermetyzacji, dziedziczenia, polimorfizmu i abstrakcji oraz ich zastosowanie w projektowaniu aplikacji [DSB1_W02].
3. Opisuje metody, techniki i narzędzia stosowane w programowaniu obiektowym, w tym wzorce projektowe oraz metaprogramowanie [DSB1_W07].

Umiejętności:
1. Tworzy klasy i obiekty w Pythonie, wykorzystując metody instancyjne, klasowe i statyczne do organizacji kodu [DSB1_U02].
2. Implementuje zasady hermetyzacji, dziedziczenia i polimorfizmu w celu budowy elastycznych i skalowalnych aplikacji obiektowych [DSB1_U08].
3. Stosuje metaprogramowanie i dekoratory do automatyzacji kodu oraz dynamicznego modyfikowania zachowania programów [DSB1_U09].
4. Projektuje, implementuje i testuje obiektowe aplikacje w Pythonie, wykorzystując wzorce projektowe oraz narzędzia do testowania jednostkowego [DSB1_U08].
5. Analizuje kod pod kątem poprawności i efektywności, stosując testy jednostkowe oraz debugowanie [DSB1_U11].
6. Rozwija swoje kompetencje w zakresie programowania obiektowego, korzystając z dokumentacji i literatury naukowej [DSB1_U15].

Kompetencje społeczne:
1. Współpracuje w zespołach programistycznych nad projektowaniem aplikacji obiektowych, stosując najlepsze praktyki programowania w Pythonie [DSB1_K02].
2. Przestrzega zasad jakości kodu i testowania, uwzględniając czytelność, reużywalność i zgodność z zasadami OOP [DSB1_K05].

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Wykład: Dwa kolokwia, za które studenci otrzymują oceny formułujące w postaci punktów - po 50 punktów za każde kolokwium. Ocena końcowa stanowi sumę punktów z dwóch ocen formułujących. Pierwsze kolokwium odbywa się w połowie kursu, a drugie na jego zakończenie. Próg zaliczeniowy to 50 punktów łącznie z obu kolokwiów.
Laboratoria: Dwa kolokwia, za które studenci otrzymują oceny formułujące w postaci punktów - po 50 punktów za każde kolokwium. Ocena końcowa stanowi sumę punktów z dwóch ocen formułujących. Pierwsze kolokwium odbywa się w połowie kursu, a drugie na jego zakończenie. Próg zaliczeniowy to 50 punktów łącznie z obu kolokwiów.

Treści programowe

Kurs obejmuje podstawy programowania obiektowego (OOP) w Pythonie, w tym porównanie z programowaniem proceduralnym. Studenci poznają koncepcję klas i obiektów, metody instancyjne, klasowe i statyczne, a także mechanizmy hermetyzacji, abstrakcji, dziedziczenia i polimorfizmu. Omówione zostaną techniki metaprogramowania, dekoratory oraz wzorce projektowe, takie jak Singleton, Factory, Observer i Adapter. Dodatkowo kurs obejmuje testowanie jednostkowe i debugowanie kodu obiektowego, przygotowując studentów do efektywnego projektowania i implementacji aplikacji zgodnie z zasadami OOP.

Tematyka zajęć

Wprowadzenie do OOP i porównanie z programowaniem proceduralnym
Przestrzenie nazw i zakresy
Tworzenie klas i obiektów w Pythonie
Metody instancyjne, klasowe i statyczne
Hermetyzacja i kontrola dostępu do danych
Abstrakcja i klasy abstrakcyjne
Dziedziczenie - jak unikać powtarzania kodu?
Polimorfizm i przeciążanie operatorów

Metaprogramowanie - dynamiczne tworzenie kodu
Dekoratory klas i metaprogramowanie
Wprowadzenie do wzorców projektowych w Pythonie
Singleton i Factory Pattern
Wzorzec projektowy Observer i Adapter
Testowanie jednostkowe klas i obiektów
Debugowanie kodu obiektowego w Pythonie

Metody dydaktyczne

Wykłady: wykład problemowy, prezentacja case studies
Laboratoria: zadania problemowe, studium przypadku, praca w grupach

Literatura

Podstawowa:
Dusty, P., Lott, S. (2023). Programowanie zorientowane obiektowo w Pythonie. Tworzenie solidnych i łatwych w utrzymaniu aplikacji i bibliotek, Helion
Vasiliev, Y. (2024). Python w data science. Praktyczne wprowadzenie, Helion, Gliwice

Uzupełniająca:
Kalb, I. (2022). Python zorientowany obiektowo. Programowanie gier i graficznych interfejsów użytkownika, Helion

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	125	5,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	60	2,50
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	65	2,50