KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Nazwa przedmiotu	Kod ECTS
Języki programowania (OA)	11.3.2099

Nazwa jednostki prowadzącej przedmiot

Instytut Informatyki

Studia

wydział	kierunek		
Wydział Matematyki, Fizyki i Informatyki	Informatyka	poziom	pierwszego stopnia
		forma	stacjonarne
		moduł specjalnościowy	wszystkie
		specjalizacja	wszystkie

Nazwisko osoby prowadzącej (osób prowadzących)

dr Tomasz Borzyszkowski; mgr Michał Zakrzewski; prof. UG, dr Piotr Arłukowicz

Formy zajęć, sposób ich realizacji i przypisana im liczba godzin

Formy zajęć

Wykład, Ćw. laboratoryjne

Sposób realizacji zajęć

zajęcia on-line, zajęcia w sali dydaktycznej

Liczba godzin

Wykład: 30 godz., Ćw. laboratoryjne: 30 godz.

Liczba punktów ECTS

7
Przedmiot w wymiarze 30h wykładu, 30h lab. + praca własna studenta, 115h
Razem 175h

Termin realizacji przedmiotu

2023/2024 letni

Status przedmiotu

obowiązkowy

Język wykładowy

polski

Metody dydaktyczne

- Analiza zdarzeń krytycznych (przypadków)
- Programowanie na żywo (life coding)
- Rozwiązywanie zadań
- Wykonywanie doświadczeń
- Wykład z prezentacją multimedialną

Forma i sposób zaliczenia oraz podstawowe kryteria oceny lub wymagania egzaminacyjne

Sposób zaliczenia

Egzamin

Formy zaliczenia

- egzamin pisemny z pytaniami (zadaniami) otwartymi
- ustalenie oceny zaliczeniowej na podstawie ocen cząstkowych otrzymywanych w trakcie trwania semestru

Podstawowe kryteria oceny

	próg zaliczenia	składnik oceny końcowej
projekty	50%	50%
aktywność na zajęciach	brak	10%
egzamin	50%	40%

Sposób weryfikacji założonych efektów uczenia się

zakładany efekt kształcenia	egzamin	kolokwium	projekt	referat	raport	aktywność w dyskusji	obserwacja postawy studenta
	Wiedza						
K_W05	x	x					
P_W01	x	x					
P_W02	x	x					
P_W03	x	x					
	Umiejętności						
K_U06			x	x	x		x
K_U09				x	x	x	x
P_U01		x		x			
P_U02		x		x			
P_U03		x		x			
	Kompetencje						
K_K01						x	x
K_K03						x	x
P_K01						x	x
P_K02						x	x

Określenie przedmiotów wprowadzających wraz z wymogami wstępnymi

A. Wymagania formalne

Ukończony przedmiot: Wstęp do programowania.

B. Wymagania wstępne

Rozumienie podstawowych struktur i konstrukcji programistycznych, takich jak

- instrukcje warunkowe, pętle, funkcje, rekurencja
- podstawowe typy danych, tablice, struktury, wskaźniki.

Znajomość podstawowych zasad poprawnego programowania strukturalnego i dobrych praktyk programistycznych.

Cele kształcenia

Celem przedmiotu jest zapoznanie słuchaczy z zasadami programowania strukturalnego i obiektowego w połączeniu z dobrymi praktykami programistycznymi na bazie wybranego języka programowania.

Treści programowe

1. **Podstawowe koncepcje**:
   - kompilatory i interpretery
   - proste i strukturalne typy danych oraz operacje na danych.
2. **Funkcje**:
   - definicja funkcji i jej znaczenie w konstrukcji programu komputerowego
   - sposoby przekazywania parametrów do funkcji oraz zwracania wyniku
   - lambda-funkcje, ich podstawy matematyczne oraz wykorzystanie w programie komputerowym
   - dokumentowanie funkcji.
3. **Moduły i pakiety**:
   - koncepcja przestrzeni nazw
   - definicja modułu oraz sposoby jego importowania
   - tworzenie pakietów/bibliotek rozwiązań.
4. **Klasy i obiekty:**
   - definicja klasy i tworzenie obiektów
   - dziedziczenie i atrybuty
   - wykorzystanie funkcji polimorficznych w programowaniu obiektowym.
5. **Testy jednostkowe:**
   - koncepje programowania sterowanego testami
   - tworzenie testów jednostkowych
   - automatyzacja testów
   - tworzenie testów behawioralnych
6. **Pliki i wyjątki**:

- podstawowe operacje na plikach
- serializacja i deserializacja danych
- tworzenie, podnoszenie i obsługa wyjątków.

7. **Wyrażenia regularne:**
   - przegląd podstawowych konstrukcji
   - grupy i podgrupy wyrażeń regularnych
   - wykorzystanie wyrażeń regularnych do wybranych zadań programistycznych.
8. **Czysty kod**:
   - przegląd podstawowych paradygmatów programistycznych oraz ich praktyczne zastosowanie na wybranych przykładach
   - wykorzystanie wybranych wzorców projektowych
   - pojęcie długu technologicznego.

Wykaz literatury

1. Metodologia programowania:
   - Robert C. Martin, Czysty kod. Podręcznik dobrego programisty. Helion 2023.
   - Beck Kent, TDD. Sztuka tworzenia dobrego kodu Wydawnictwo Helion, 2020.
   - Harry J.W. Percival. TDD w praktyce. Niezawodny kod w języku Python. Wydawnictwo Helion, 2020.
   - Eric Freeman, Elisabeth Freeman, Kathy Sierra, Bert Bates, Head First Design Patterns. Edycja polska (Rusz głową!). Helion 2020.
2. Wybrany język programowania:
   - Guido van Rossum, Python Tutorial, http://docs.python.org/tut/.
   - Mark Pilgrim, Dive into Python. http://diveintopython.org/.
   - Bruce Eckel, Thinking in Python, http://www.mindview.net/Books/TIPython.
   - Python's official documentation, http://docs.python.org/.

Kierunkowe efekty uczenia się

K_W05: ma ogólną wiedzę na temat różnych paradygmatów programowania i języków programowania; szczegółowo zna metody i wzorce projektowania i programowania obiektowego
K_U06: potrafi projektować, tworzyć, uruchamiać i testować programy przy wykorzystaniu dedykowanych narzędzi oraz adekwatnych wzorców
K_U09: potrafi oceniać przydatność paradygmatów i narzędzi programistycznych do rozwiązywania problemów różnego typu
K_K01: zna ograniczenia własnej wiedzy i rozumie potrzebę dalszego uczenia się
K_K03: potrafi i jest gotów formułować opinie na temat podstawowych zagadnień informatycznych

Wiedza

P_W01: student wyjaśnia działanie podstawowych konstrukcji programistycznych takich jak instrukcje wyboru, pętle, bloki, struktury itp.
P_W02: student rozumie na czym polega praca z danymi umieszczonymi w pamięci komputera oraz na dysku.
P_W03: student rozróżnia różne rodzaje języków programowania i potrafi wskazać ich najważniejsze zastosowania.

Umiejętności

P_U01: student potrafi projektować, tworzyć, uruchamiać i testować programy przy wykorzystaniu dedykowanych narzędzi oraz adekwatnych wzorców
P_U02: student potrafi rozwiązań proste problemy programistyczne przy użyciu metod programowania strukturalnego i innych.
P_U03: student potrafi zastosować poznane konstrukcje programistyczne do pisania działających i poprawnych programów.

Kompetencje społeczne (postawy)

P_K01: zna ograniczenia własnej wiedzy w dziedzinie programowania i rozumie potrzebę dalszego uczenia się i dokładniejszego poznawania języków programowania
P_K02: potrafi i jest gotów formułować opinie na temat podstawowych zagadnień informatycznych związanych z poznanymi językami programowania

Kontakt

tomasz.borzyszkowski@ug.edu.pl