

KARTA KURSU

Nazwa	Podstawy programowania w języku Python
Nazwa w j. ang.	Python Programming Fundamentals

Koordynator	dr Roman Czapla	Zespół dydaktyczny
		dr Iryna Artyshchuk dr Roman Czapla mgr inż. Katarzyna Marczak mgr Patryk Mazurek dr Anna Wojciechowska
Punktacja ECTS*	4	

Opis kursu (cele kształcenia)

Celem przedmiotu jest zapoznanie studentów z podstawami programowania w języku Python, z uwzględnieniem ich wcześniejszych umiejętności programistycznych w języku C. Kurs ma na celu przedstawienie specyfiki języka Python, jego składni oraz najlepszych praktyk w tym języku. Studenci rozwiną umiejętność rozwiązywania problemów programistycznych przy użyciu Pythona oraz wykorzystania jego bogatego zestawu bibliotek.

Warunki wstępne

Wiedza	Student zna podstawowe pojęcia dot. programowania w języku C.
Umiejętności	Student posiada podstawowe umiejętność programowania i implementacji prostych algorytmów w języku C.
Kursy	Wstępne kursy nie są wymagane.

Efekty uczenia się

	Efekt uczenia się	Odniesienie do efektów kierunkowych
Wiedza	Po zakończeniu kursu student:	
	W01: zna różnice między językiem Python a językiem C, w tym zasadnicze różnice w zarządzaniu pamięcią oraz w sposobie działania i rozumie podstawy funkcjonowania środowiska programistycznego języka Python.	K_W07
	W02: rozumie podstawowe typy danych w Pythonie, potrafi wyjaśnić zasady ich zarządzania w Pythonie w porównaniu do języka C, a także rozumie znaczenie operatorów w kontekście Pythona.	K_W07
	W03: zna instrukcje warunkowe, pętle i instrukcję dopasowań wzorców w języku Python oraz widzi podobieństwa i różnice w kontekście porównania z analogicznymi instrukcjami języka C.	K_W07
	W04: rozumie zasady definiowania funkcji w Pythonie i zna różnice w tworzeniu funkcji w Pythonie i języku C.	K_W07
	W05: posiada wiedzę na temat zaawansowanych struktur danych i technik w Pythonie, takich jak struktury składane, segmentowanie danych. Zna również funkcje map(), filter(), reduce() oraz rozumie działanie generatorów i wyrażeń generatorów.	K_W07
	W06: rozumie operacje na plikach w Pythonie i zna metody pracy z plikami tekstowymi i binarnymi, a także rozumie zasady obsługi wyjątków w Pythonie.	K_W07

	Efekt uczenia się	Odniesienie do efektów kierunkowych
Umiejętności	Po zakończeniu kursu student:	
	U01: potrafi tworzyć i modyfikować programy w Pythonie, wykorzystując różne typy danych i operatory do rozwiązywania praktycznych problemów.	K_U05
	U02: potrafi zaimplementować instrukcje warunkowe i pętle w Pythonie, tworzyć programy, które wykorzystują pętle for i while do rozwiązywania problemów iteracyjnych, oraz potrafi również wykorzystać instrukcję match do dopasowywania wzorców w bardziej złożonych przypadkach.	K_U05
	U03: jest w stanie tworzyć funkcje w Pythonie, implementować funkcje rekurencyjne oraz lambda do rozwiązywania zadań programistycznych.	K_U05
	U04: potrafi stosować zaawansowane techniki programowania w Pythonie, takie jak listy składane, funkcje map(), filter() i reduce(), a także wykorzystywać generatory i wyrażenia generatorów do tworzenia efektywnych rozwiązań programistycznych.	K_U05
	U05: umie pracować z danymi w Pythonie, w tym operować na plikach tekstowych i binarnych oraz potrafi również efektywnie obsługiwać wyjątki i błędy w swoich programach.	K_U05
	U06: potrafi analizować i oceniać wydajność struktur danych w Pythonie, wybierać odpowiednie struktury (listy, zbiory, słowniki) w zależności od wymagań zadania i optymalizować kod w celu zwiększenia jego wydajności.	K_U05
	U07: potrafi tworzyć i importować moduły w Pythonie, używać wbudowanych modułów (np. os, sys, random, math, collections) oraz tworzyć własne moduły, które umożliwiają lepszą organizację kodu w większych projektach.	K_U05

	Efekt uczenia się	Odniesienie do efektów kierunkowych
Kompetencje społeczne	Po zakończeniu kursu student:	
	K01: potrafi korzystać z różnych źródeł informacji (zasobów sieci Internet) do poszerzania własnej wiedzy i zdobywania nowych umiejętności.	K_K02
	K02: potrafi przekazać wiedzę informatyczną w sposób zrozumiały dla innych.	K_K01

Studia stacjonarne

Organizacja											
Forma zajęć	Wykład (W)	Ćwiczenia w grupach									
		A		K		L		S		P	E
Liczba godzin	10					30					

Studia niestacjonarne

Organizacja											
Forma zajęć	Wykład (W)	Ćwiczenia w grupach									
		A		K		L		S		P	
Liczba godzin	6					20					

Opis metod prowadzenia zajęć

Wykłady teoretyczne: omówienie materiału z naciskiem na różnice między Pythonem a językiem C oraz na charakterystyczne cechy języka Python.

Ćwiczenia praktyczne: studenci implementują programy w Pythonie, ucząc się jego składni, struktur danych, list składanych i innych technik.

Na zakończenie kursu uczestnicy przystąpią do testu oceniającego poziom opanowania materiału. Test będzie obejmował zarówno pytania teoretyczne, jak i praktyczne zadania programistyczne, pozwalające zweryfikować zdobytą wiedzę i umiejętności w zakresie podstaw programowania w języku Python.

Formy sprawdzania efektów uczenia się

	E – learning	Gry dydaktyczne	Ćwiczenia w szkole	Zajęcia terenowe	Praca laboratoryjna	Projekt indywidualny	Projekt grupowy	Udział w dyskusji	Referat	Praca pisemna (esej)	Egzamin ustny	Egzamin pisemny	Inne
W01					x	x	x	x					
W02					x	x	x	x					
W03					x	x	x	x					
W04					x	x	x	x					
W05					x	x	x	x					
W06					x	x	x	x					
U01					x	x	x	x					
U02					x	x	x	x					
U02					x	x	x	x					
U03					x	x	x	x					
U04					x	x	x	x					
U05					x	x	x	x					
U06					x	x	x	x					
U07					x	x	x	x					
K01								x					
K02								x					

Osiągnięcie efektów kształcenia podanych powyżej uprawnia studentów do uzyskania oceny nie wyższej niż dostateczna – warunkiem jej przyznania jest zdobycie odpowiedniej liczby punktów na teście końcowym.

Kryteria oceny

Ocena dobra (4.0) może zostać przyznana studentowi, który wykazuje się dobrą znajomością języka Python, potrafi pisać efektywny i czytelny kod. Używa zaawansowanych funkcji i struktur danych, jednak w bardziej skomplikowanych zadaniach mogą mu się zdarzać drobne błędy. Potwierdzeniem tego poziomu opanowania materiału jest uzyskanie odpowiedniej liczby punktów na teście zaliczeniowym.

Ocena bardzo dobra (5.0) przysługuje studentowi, który posiada pełną kontrolę nad językiem Python, pisze optymalny i dobrze zorganizowany kod. Efektywnie stosuje zaawansowane techniki programistyczne i potrafi rozwiązywać skomplikowane problemy programistyczne. Potwierdzeniem tego poziomu wiedzy i umiejętności jest zdobycie wysokiej liczby punktów na teście zaliczeniowym.

Uwagi

Treści merytoryczne (wykaz tematów)

1. Wprowadzenie do języka Python i jego różnice w stosunku do języka C:
 - wstęp do Pythona: różnice i podobieństwa w porównaniu do języka C;
 - środowisko programistyczne;
 - zasady działania Pythona: interpretowany vs kompilowany język, zarządzanie pamięcią (brak wskaźników, automatyczne zarządzanie pamięcią).
2. Typy danych, operatory i struktury w języku Python:
 - podstawowe typy danych: liczby, napisy,
 - operatory: arytmetyczne, logiczne, porównania;
 - różnice w zarządzaniu typami danych w Pythonie a C;
 - podstawowe struktury danych w Pythonie: listy, krotki, słowniki, zbiory.
3. Instrukcje sterujące i pętle:
 - instrukcje warunkowe i operator warunkowy – porównanie z językiem C;
 - pętle for, while, różnice w pętlach w językach C i Pythonie;
 - instrukcja match case.
4. Funkcje i moduły:
 - tworzenie funkcji, argumenty i zwracanie wartości;
 - typy parametrów: domyślne, opcjonalne i *args, **kwargs;
 - funkcje lambda i ich zastosowanie (np. funkcje sort(), min(), max(), itp.);
 - rekurencja w języku Python;
 - importowanie i tworzenie modułów;
 - praca z wbudowanymi modułami (os, sys, random, math, collections, itp.);
 - wybrane pakiety języka Python spoza biblioteki standardowej (omawiane w ramach pracy laboratoryjnej).
5. Struktury danych i techniki Pythona:
 - listy składane (list comprehensions) i ich zastosowanie;
 - słowniki i zbiory składanie;
 - segmentowanie w Pythonie (wycinanie elementów z list, krotek, napisów);
 - funkcje map(), filter(), reduce() i ich zastosowanie w programowaniu funkcyjnym;
 - funkcje generatorów i wyrażanie genartora w Pythonie;
 - wydajność struktur danych w Pythonie (listy vs zbiory vs słowniki).
6. Praca z plikami, zarządzanie danymi i obsługa wyjątków:
 - operacje na plikach: odczyt, zapis, modyfikacja;
 - obsługa plików tekstowych i binarnych;
 - korzystanie z kontekstu with przy pracy z plikami;
 - serializacja i deserializacja danych;
 - obsługa błędów i wyjątków.

Wykaz literatury podstawowej

Wskazane przez prowadzącego rozdziały:

1. E. Matthes, *Python. Instrukcje dla programisty. Wydanie III*, Helion, Gliwice 2023;
2. A. Martelli, A. Martelli Ravenscroft, S. Holden, P. McGuire, *Python w pigułce. Podręczny przewodnik po wersjach 3.10 i 3.11*, Promise, Warszawa 2023;
3. D. Beazley, *Python. Zwięzłe kompendium dla programisty*, Helion, Gliwice 2023;
4. A. Sweigart, *Wielka księga małych projektów w Pythonie. 81 łatwych praktycznych programów*, Helion, Gliwice 2022;
5. Ch. Mayer, *Kod Pythona w jednym wierszu. Jak profesjonaliści piszą programy doskonałe*, Helion Gliwice 2021;

Wykaz literatury uzupełniającej

1. A. Sweigart, *Rekurencyjna książka o rekurencji. Zostań mistrzem rozmów kwalifikacyjnych poświęconych językom Python i JavaScript*, Helion, Gliwice 2023.
2. P. Wróblewski, *Python dla testera*, Helion, Gliwice 2021
3. L. Vaughan, *Python mniej poważnie*, Wydawnictwo Naukowe PWN, Warszawa 2020;
4. P. J. Deitel, H. Deitel, *Python dla programistów. Big Data i AI. Studia przypadków*, Helion, 2020;
5. D. Kopec, *Klasyczne problemy informatyki w Pythonie*, Wydawnictwo Naukowe PWN, Warszawa 2020;
6. M. Lutz, *Python. Leksykon kieszonkowy. Wydanie V*, Helion Gliwice 2014;
7. M. Lutz, *Python. Wprowadzenie. Wydanie IV*, Helion Gliwice 2010.

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) - studia stacjonarne

Liczba godzin w kontakcie z prowadzącymi	Wykład	10
	Konwersatorium (ćwiczenia, laboratorium itd.)	30
	Pozostałe godziny kontaktu studenta z prowadzącym	5
Liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	55
	Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu	0
	Przygotowanie projektu lub prezentacji na podany temat (praca w grupie)	0
	Przygotowanie do egzaminu	0
Ogółem bilans czasu pracy		100
Liczba punktów ECTS w zależności od przyjętego przelicznika		4

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) - studia niestacjonarne

Liczba godzin w kontakcie z prowadzącymi	Wykład	6
	Konwersatorium (ćwiczenia, laboratorium itd.)	20
	Pozostałe godziny kontaktu studenta z prowadzącym	4
Liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	70
	Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu	0
	Przygotowanie projektu lub prezentacji na podany temat (praca w grupie)	0
	Przygotowanie do egzaminu	0
Ogółem bilans czasu pracy		100
Liczba punktów ECTS w zależności od przyjętego przelicznika		4