KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu

Podstawy programowania - Python

Przedmiot

Kierunek studiów
Informatyka

Rok/semestr
1/1

Studia w zakresie (specjalność)

Profil studiów
ogólnoakademicki

Poziom studiów
pierwszego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obieralny

Liczba godzin

Wykład	Laboratoria	Inne (np. online)
24	30	
Ćwiczenia	Projekty/seminaria	

Liczba punktów ECTS

4

Wykładowcy

Odpowiedzialny za przedmiot/wykładowca:
mgr inż. Jan Badura
email: jan.badura@cs.put.poznan.pl
wydział: Wydział Informatyki I Telekomunikacji
adres: ul. Piotrowo 3 60-965 Poznań

Odpowiedzialny za przedmiot/wykładowca:

Wymagania wstępne

Zgodnie z podstawą programową kształcenia ogólnego dostępną na stronie: http://cke.gov.pl zakłada się, że rozpoczynając przedmiot student ma podstawowe umiejętności z:

- matematyki: IV etap edukacyjny, zakres podstawowy poszerzony o rachunek różniczkowy (z zakresu rozszerzonego);

- informatyki: IV etap edukacyjny, zakres podstawowy.

W zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu

Celem przedmiotu jest zapoznanie studentów z podstawami programowania komputerów oraz nauczenie programowania w języku Python3. W szczególności obejmuje to:
- przekazanie studentom podstawowych informacji o rozwoju języków programowania, programowaniu strukturalnym, zasadach programowania zorientowanego obiektowo oraz konstrukcji programów tekstowych i okienkowych,
- rozwijanie u studentów umiejętności algorytmizacji problemów i ich oprogramowaniu, w tym w postaci funkcji,
- nauczenie studentów biegłego posługiwania się zintegrowanym systemem programowania,
- opanowanie przez studentów techniki programowania zorientowanego obiektowo, w tym tworzenia różnych jednostek programowych oraz dostępu do danych i kodów w nich zawartych,
- nauczenie studentów tworzenia i obsługi modułów i pakietów oraz wykorzystania ich w programach,
- nabycie przez studentów umiejętności programowego zabezpieczania kodów przed błędami wykonywania programów.

Przedmiotowe efekty uczenia się

Wiedza

1. Ma rozszerzoną i pogłębioną wiedzę z matematyki przydatną do formułowania i rozwiązywania złożonych zadań informatycznych dotyczących programowania w języku Python3, formalnej specyfikacji i weryfikacji oprogramowania (K1st_W1)
2. Ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną w zakresie kluczowych zagadnień informatyki, którymi są techniki programistyczne oraz wiedzę szczegółową w zakresie wybranych zagadnień tej dyscypliny (K1st_W4)
3. Zna podstawowe techniki, metody oraz narzędzia wykorzystywane w procesie rozwiązywania zadań informatycznych (tutaj programistycznych), głównie o charakterze inżynierskim (K1st_W7)

Umiejętności

1. Potrafi, formułując i rozwiązując zadania programistyczne, zastosować odpowiednio dobrane metody, w tym metody analityczne, symulacyjne lub eksperymentalne implementowane w języku Python3 (K1st_U4)
2. Ma umiejętności formułowania algorytmów i ich implementacji z użyciem przynajmniej jednego z popularnych narzędzi, którym jest język Python3 (K1st_U11)
3. Potrafi organizować, współdziałać i pracować w grupie, przyjmując w niej różne role oraz potrafi odpowiednio określić priorytety służące realizacji określonego przez siebie lub innych zadania w celu opracowania jak najbardziej wydajnego algorytmu sztucznej inteligencji grającego w gry komputerowe (K1st_U18)

Kompetencje społeczne

1. Na podstawie historii rozwoju języka Python3 rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe (K1st_K1)
2. Ma świadomość znaczenia wiedzy w rozwiązywaniu problemów inżynierskich, szczególnie związanych z programowaniem na przykładzie języka Python3 oraz zna przykłady i rozumie przyczyny wadliwie działających systemów informatycznych (K1st_K2)

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:
Ocena formująca:
- na podstawie odpowiedzi udzielanych odnośnie realizacji zadań w ramach laboratoriów;
Ocena podsumowująca:
Sprawdzanie założonych efektów kształcenia realizowane jest przez:
- ocenę umiejętności związanych z realizacją zadań laboratoryjnych (aby uzyskać ocenę 3.0, w ciągu semestru każdy student ma do rozwiązania określone przez prowadzącego na początku semestru proste programy rozwiązujące zadania dostępne w systemie HackerRank. Dodatkowo student może zwiększyć swoją ocenę o 0.5 stopnia za każde 10 zadań rozwiązanych ze zbioru zadań dodatkowych – także wskazanego przez prowadzącego na początku semestru,
- ocenę wiedzy i umiejętności wykazanych na pisemnym (na komputerze) kolokwium zaliczeniowym o charakterze problemowym i praktycznym (kolokwium składa się z zadań programistycznych sprawdzanych w sposób automatyczny na komputerze, które trzeba rozwiązać aby uzyskać zaliczenie przedmiotu na ocenę wynikającą z ilości rozwiązanych zadań laboratoryjnych.

Treści programowe

Program przedmiotu obejmuje następujące zagadnienia:
- podstawowe pojęcia związane z programowaniem (programowanie, algorytm, program, język programowania, język ukierunkowany maszynowo, rozkaz, język wyższego rzędu, język uniwersalny, język specjalizowany),
- przegląd języków programowania (Ada, Algol, asemblery, Basic, C, C++, Cobol, Fortran, HTML, Java, Lisp, Logo, Pascal, PHP, PL/1, Prolog),
-sieci działań (schematy blokowe) i symbole stosowane w nich,
- ogólne zasady programowania zorientowanego obiektowo (dziedziczność, hermetyczność i polimorfizm),
- ogólna charakterystyka pakietu PyCharm,
- podstawowe pojęcia związane z konstruowaniem programów w zintegrowanym systemie programowania PyCharm (projekt, pakiet, moduł),
- posługiwanie się zintegrowanym pakietem programowania PyCharm,
- przegląd konstrukcji języka Python3 (program, moduł, pakiet, funkcje, klasy i obiekty, typy danych, zmienne, instrukcje),
- struktura programu i pakietu,
- podstawowe elementy języka (symbole podstawowe, słowa kluczowe i dyrektywy języka, identyfikatory, liczby, łańcuchy, w tym łańcuchy znaków Unicode, literały logiczne, komentarze i separatory),
- typy danych i ich opis (definiowanie typów, typy proste, łańcuchowe, opisujące obiekty, zgodność typów, dekoratory),
- zmienne (deklaracje zmiennych, zmienne indeksowane, obiektowe, dynamiczne, funkcyjne, z początkową wartością, nakładanie zmiennych, literały stałe i zmienne),

- wyrażenia (rodzaje operatorów i ich priorytet, składnia wyrażenia, wyrażenie stałe),
- instrukcje (proste, strukturalne, wyrażenia lambda),
- funkcje (definicje, rodzaje parametrów, przeciążanie, wywoływanie),
- przetwarzanie obiektów (konstruktory i destruktory, metody statyczne, wirtualne, dynamiczne i abstrakcyjne, obsługa wiadomości, własności),
- publikowanie pakietów w Python Package Index,
- przetwarzanie plików, serializacja,
- wielowątkowość (synchronizacja wątków, priorytety, oczekiwanie na zakończenie).

Na zajęciach laboratoryjnych studenci, po zapoznaniu się ze zintegrowanym środowiskiem programowania PyCharm, piszą programy wykorzystujące poznane elementy języka.

Metody dydaktyczne

1. Wykład: prezentacja multimedialna oraz prezentacja pisania i wykonywania wybranych programów bezpośrednio w pakiecie PyCharm lub Interactive.
2. Ćwiczenia laboratoryjne: ćwiczenia praktyczne dotyczące elementów języka Python3, pisanie programów w tym języku.

Literatura

Podstawowa

1. Python 3: kompletne wprowadzenie do programowania, Mark Summerfield, Helion, 2010

Uzupełniająca

1. Python. Wprowadzenie. Wydanie IV, Mark Lutz, Helion 2010

2. How to Think Like a Computer Scientist: Interactive Edition (https://runestone.academy/ns/books/published/thinkcspy/index.html)

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	100	4,0
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	54	2,0
Praca własna studenta (studia literaturowe, przygotowanie do ćwiczeń, przygotowanie do kolokwium, wykonanie pracy kontrolnej)[1]	46	2,0

[1] niepotrzebne skreślić lub dopisać inne czynności