KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Podstawy programowania - Python	**1010514311010500074**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**1 / 1**
Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
-	**polski**	**obieralny**

Stopień studiów:	**Forma studiów** (stacjonarna/niestacjonarna)
I stopień	**niestacjonarna**

Godziny	Liczba punktów
Wykłady: **12** Ćwiczenia: **-** Laboratoria: **20** Projekty/seminaria: **-**	**5**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
kierunkowy	**z danego kierunku**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**
nauki techniczne	**100 5%**
nauki techniczne	**100 5%**

Odpowiedzialny za przedmiot / wykładowca:

dr inż. Szymon Wąsik
email: szymon.wasik@cs.put.poznan.pl
tel. 61 665 3032
Wydział Informatyki
ul. Piotrowo 2, 60-965 Poznań

Odpowiedzialny za przedmiot / wykładowca:

dr inż. Marcin Borowski
email: marcin.borowski@cs.put.poznan.pl
tel. 61 665 3032
Wydział Informatyki
ul. Piotrowo 2, 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Zgodnie z podstawą programową kształcenia ogólnego dostępną na stronie: http://bip.men.gov.pl/men bip/akty_prawne/rozporzadzenie_20081223_zal_4.pdf zakłada się, że rozpoczynając przedmiot student ma podstawową wiedzę: - z matematyki: IV etap edukacyjny, zakres podstawowy poszerzony o rachunek różniczkowy (z zakresu rozszerzonego); - z informatyki: IV etap edukacyjny, zakres podstawowy.
2	**Umiejętności:**	Zgodnie z podstawą programową kształcenia ogólnego dostępną na stronie: http://bip.men.gov.pl/men bip/akty_prawne/rozporzadzenie_20081223_zal_4.pdf zakłada się, że rozpoczynając przedmiot student ma podstawowe umiejętności: - z matematyki: IV etap edukacyjny, zakres podstawowy poszerzony o rachunek różniczkowy (z zakresu rozszerzonego); - z informatyki: IV etap edukacyjny, zakres podstawowy.
3	**Kompetencje społeczne**	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 Python. 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.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma rozszerzoną i pogłębioną wiedzę z matematyki przydatną do formułowania i rozwiązywania złożonych zadań informatycznych dotyczących programowania w języku Python, 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 Python - [K1st_U4]

2. ma umiejętności formułowania algorytmów i ich implementacji z użyciem przynajmniej jednego z popularnych narzędzi, którym jest język Python - [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 Python 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 Python oraz zna przykłady i rozumie przyczyny wadliwie działających systemów informatycznych - [K1st_K2]

Sposoby sprawdzenia efektów kształcenia

Efekty kształcenia 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 (w ciągu semestru każdy student ma do napisania algorytm grający w pewną grę w serwisie CodeInGame, który awansuje do wskazanej dywizji X na ocenę 3.0, X+1 na 4.0, X+2 na 5.0 oraz 64 proste programy rozwiązujące zadania dostępne w systemie CheckIO: 32 na 3.0, 40 na 3.5, itd. co 8 programów ocena o 0.5 wyżej),

- ocenę wiedzy i umiejętności związanych z realizacją ćwiczeń laboratoryjnych poprzez kolokwium w połowie semestru (20 zadań testowych ocenianych w zależności od poziomu trudności na 1-5 punktów, 50% punktów na 3.0, 60% punktów na 3.5, itd. co 10%),

- ocenę wiedzy i umiejętności wykazanych na pisemnym 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 będącą średnią ocen z CodeinGame, CheckIO i kolokwium w połowie semestru).

Studenci, którzy wyróżniająco wykonali programistyczne zadania laboratoryjne (co najmniej 75 zadań w CheckIO) są zwalniani z kolokwium w połowie semestru z oceną 5.0.

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 Python (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 (każdy wykład) oraz prezentacja pisania i wykonywania wybranych programów bezpośrednio w pakiecie PyCharm lub Interactive Python, quizy społecznościowe (Kahoot).
2. Ćwiczenia laboratoryjne: ćwiczenia praktyczne dotyczące elementów języka Python, pisanie programów w tym języku.

Literatura podstawowa:

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

Literatura uzupełniająca:

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

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. Udział w zajęciach laboratoryjnych	20
2. Przygotowanie do ćwiczeń laboratoryjnych	8
3. Udział w konsultacjach związanych z realizacją procesu kształcenia, w szczególności ćwiczeń laboratoryjnych	3
4. Napisanie programów, uruchomienie I weryfikacja (czas poza zajęciami laboratoryjnymi)	20
5. Przygotowanie do sprawdzianów	12
6. Udział w wykładach	12
7. Zapoznanie się ze wskazaną literaturą - 300 stron	30
8. Przygotowanie do zaliczenia wykładów i udział w kolokwium zaliczeniowym	12

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	117	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	35	3
Zajęcia o charakterze praktycznym	48	2