Politechnika Krakowska im. Tadeusza Kościuszki

Karta przedmiotu

obowiązuje studentów rozpoczynających studia w roku akademickim 2021/2022

Wydział Informatyki i Telekomunikacji

Kierunek studiów: Informatyka

Profil: Ogólnoakademicki

Forma sudiów: niestacjonarne

Kod kierunku: I

Stopień studiów: I

Specjalności: Brak specjalności

1 Informacje o przedmiocie

Nazwa przedmiotu	Języki symboliczne
Nazwa przedmiotu w języku angielskim	Symbolic languages
Kod przedmiotu	WIiT I oIN D4 21/22
Kategoria przedmiotu	Przedmioty specjalnościowe
Liczba punktów ECTS	4.00
Semestry	5

2 Rodzaj zajęć, liczba godzin w planie studiów

Semestr	Wykład	Ćwiczenia	Laboratorium	Laboratorium komputerowe	Seminarium	Projekt
5	18	0	18	0	0	0

3 Cele przedmiotu

Cel 1 Poznanie składni języka Python. Poznanie podstawowych typów danych i instrukcji sterujących używanych w języku Python.

Cel 2 Nabycie umiejętności projektowania i implementacji programów w języku Python. Umiejętność wykorzystania zaawansowanych typów danych języka Python. Umiejętność wykorzystania języka Python w zakresie programowania w modelu obiektowym.

Cel 3 Poznanie paradygmatu programowania imperatywnego i jego różnicy w stosunku do deklaratywnego podejścia matematycznego. Poznanie składni języka LISP jako przykładu języka symbolicznego pozwalającego na implementowania przetwarzania symbolicznego. Język ten jest zarazem językiem ogólnego przeznaczenia umożliwiającym programowanie w stylu proceduralnym, funkcyjnym i obiektowym.

4 Wymagania wstępne w zakresie wiedzy, umiejętności i innych kompetencji

1 Zaliczone przedmioty: Wstęp do programowania oraz Algorytmy i struktury danych.

5 Efekty kształcenia

EK1 Wiedza Student zna składnię języka Python oraz typowych dla niego typów danych. Umie wykorzystać zaawansowane możliwości języka Python w zakresie programowania w modelu proceduralnym i obiektowym.

EK2 Wiedza Student zna podstawy składni języka LISP. Rozumie na czym polega zastosowanie mechanizmów typowych dla języka w celu budowy abstrakcji na poziomie procesu obliczeniowego, złożonych procedur, danych.

EK3 Umiejętności Student potra pisać programy użytkowe w języku Python oraz odpowiednio dobrać model programowania i struktury danych. Umie skorzystać z dokumentacji języka Python.

EK4 Umiejętności Student umie skongurować środowisko programistyczne w dialekcie Scheme oraz potra napisać i uruchomić własny program w języku LISP rozwiązujący postawiony przed nim problem.

EK5 Kompetencje społeczne Student rozumie potrzebę tworzenia czytelnych i wydajnych programów.

6 Treści programowe

Laboratorium		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
L1	Podstawy składni języka Python. Podstawowe struktury danych. Podstawowe operacje na łańcuchach znakowych, listach i słownikach.	3
L2	Definicje funkcji i argumentów, podstawowe funkcje wbudowane, wyrażenia warunkowe, wyrażenia lambda, pętle. Iterowanie po indeksach sekwencji. Domyślne argumenty funkcji. Tworzenie własnych modułów i pakietów. Zasady zasięgu zmiennych. Implementacja rozbudowanych struktur danych. Listy zagnieżdżone, krotki, zbiory, słowniki.	3
L3	Operacje na plikach. Formatowanie napisów. Tworzenie i wykorzystanie iteratorów i generatorów. Obsługa błędów i wyjątków. Wyjątki definiowane przez użytkownika. Predefiniowane akcje typu clean up.	3
L4	Programowanie obiektowe, hierarchia klas. Obiekty klas i metod. Zmienne obiektów (instancji) i zmienne klas. Atrybuty klas i obiektów. Dziedziczenie klas. Zmienne prywatne. Wyjątki jako klasy.	3

Laboratorium		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
L5	Konfiguracja środowiska programistycznego dla dialektu języka LISP Schame. Obliczanie wartości prostych wyrażeń. Kombinacje wyrażeń. Operatory i argumenty, notacja prefiksowa. Zagnieżdżanie kombinacji. Formatowanie kodu źródłowego. Zmienne i ich wartości. Nazwy i środowisko. Definiowanie procedur, formy specjalne. Parametry formalne w definicji procedury. Podstawieniowy model stosowania procedur. Stosowana a normalna kolejność obliczeń. Wyrażenia warunkowe i predykaty, analiza przypadków. Predykaty, klauzule i następniki. Wyrażenia cond, if, else, relacje pierwotne oraz operacje logiczne takie jak: and, or, not. Implementacja pierwiastkowania metodą Newtona. Obliczanie pierwiastków sześciennych. Implementacje programów z wykorzystaniem zmiennych lokalnych. Parametry formalne jako zmienne związane. Zakres zmiennych związanych deklarowanych jako parametry formalne procedury. Wiązanie składniowe.	3
L6	Obliczanie n! i przykłady rekursji. Implementacja n! jako liniowy proces rekurencyjny oraz liniowy proces iteracyjny. Rekursja ogonowa. Obliczanie liczb Fibonacciego jako przykład rekursji rozgałęziającej się (drzewiastej). Implementacja operacji potęgowania, poszukiwanie największych wspólnych dzielników oraz licz pierwszych. Procedury operujące na procedurach, procedury wyższego rzędu. Użycie procedur jako argumentów na przykładzie implementacji sumowania szeregów. Lambda-abstrakcje. Tworzenie zmiennych lokalnych za pomocą let. Implementacja zastosowania procedur jako metod ogólnych poprzez znajdowanie pierwiastków równań przez bisekcję. Tworzenie procedur których wyniki same są procedurami. Abstrakcja danych łączenie obiektów danych w dane złożone. Implementacja przykładowych operacji arytmetycznych na liczbach wymiernych. Pary jako - przykład struktur złożonych.	3

Wykład		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
W1	Informacje o przedmiocie. Wprowadzenie do języka Python. Wersje języka. Opis interpretera. Tryby pracy interpretera. Raportowanie błędów. Automatyczne uruchamianie skryptów języka Python. Kodowanie znaków, Konwencje dotyczące komentarzy. Podstawowe typy danych języka Python. Podstawowe operacje na łańcuchach znakowych, listach i słownikach.	3
W2	Definicje funkcji i argumentów, wyrażenia warunkowe, wyrażenia lambda, pętle. Iterowanie po indeksach sekwencji. Domyślne argumenty funkcji. Napisy dokumentujące. Konwencje kodowania i stylu. Rozbudowane struktury danych stos, kolejki. Listy zagnieżdżone, krotki, zbiory, słowniki. Mutowalne i niemutowalne typy danych. Moduły i pakiety oraz ich import. Kompilacja w języku Python. Odwołania relatywne i absolutne.	3
W3	Operacje wejścia i wyjścia języka Python. Konwersja danych do stringów. Formatowanie napisów. Czytanie i pisanie do plików. Tworzenia obiektów typu: persistent. Zgłaszanie i obsługa błędów i wyjątków. Wyjątki definiowane przez użytkownika. Predefiniowane akcje typu clean up.	3

Wykład		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
W4	Przestrzenie nazw. Instrukcje nonlocal i global. Zasięg zmiennych. Programowanie obiektowe w Pythonie. Definicja klas. Obiekty klas i metod. Zmienne obiektów (instancji) i zmienne klas. Atrybuty klas i obiektów. Dziedziczenie klas. Zmienne prywatne. Wyjątki jako klasy. Iteratory i generatory. Biblioteka standardowa. Operacje na wzorcach. Obliczenia matematyczne. Kontrola jakości pisanego oprogramowania, testy.	3
W5	Podstawowe informacje o języku Lisp, jego historii i dialektach. Przedstawienie idei programowania imperatywnego w stosunku do podejścia deklaratywnego. Elementy programowania. Nazwy i środowisko programistyczne. Proste wyrażenia i operacje na prostych typach danych. Podstawy składni. Procedury złożone. Podstawieniowy model stosowania procedur. Stosowana a normalna kolejność obliczania. Wyrażenia warunkowe i predykaty. Procedury jako abstrakcyjne, czarne skrzynki. Formułowanie abstrakcji za pomocą procedur wyższych rzędów. Tworzenie procedur za pomocą lambda-abstrakcji. Tworzenie zmiennych lokalnych za pomocą let. Procedury jako metody ogólne. Procedury jako wyniki.	3
W6	Budowanie abstrakcji za pomocą danych. Bariery abstrakcji. Pojęcie danych. Dane hierarchiczne i własności domknięcia. Reprezentowanie ciągów, list, struktur listowych. Operacje na listach, odwzorowywanie list. Wyrażenia symboliczne i ich wartościowanie. Hierarchiczne struktury danych, odwzorowywanie drzew. Ciągi jako interfejsy konwencjonalne. Operacje na ciągach. Odwzorowania zagnieżdżone. Zbiory jako listy nieuporządkowane i uporządkowane. Zbiory jako drzewa binarne. Wyszukiwanie informacji w zbiorach. Systemy z operacjami ogólnymi. Ogólne operacje arytmetyczne. Łączenie danych różnych typów. Algebra symboliczna.	3

7 Narzędzia dydaktyczne

N1 Wykłady

N2 Ćwiczenia laboratoryjne

8 Obciążenie pracą studenta

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe z nauczycielem akademickim, w tym:	
Godziny wynikające z planu studiów	36
Konsultacje przedmiotowe	20
Egzaminy i zaliczenia w sesji	4
Godziny bez udziału nauczyciela akademickiego wynikające z nakładu pracy studenta, w tym:	
Przygotowanie się do zajęć, w tym studiowanie zalecanej literatury	20
Opracowanie wyników	20
Przygotowanie raportu, projektu, prezentacji, dyskusji	20
Sumaryczna liczba godzin dla przedmiotu wynikająca z całego nakładu pracy studenta	**120**
Sumaryczna liczba punktów ECTS dla przedmiotu	4.00

9 Sposoby oceny

Ocena formująca

F1 Kolokwium (2 razy)

F2 Egzamin

Ocena podsumowująca

P1 Średnia ważona ocen formujących

Warunki zaliczenia przedmiotu

W1 Warunkiem jest zaliczenie wszystkich laboratoriów, kolokwiów i egzaminu na ocenę co najmniej 3.0

Kryteria oceny

Efekt kształcenia 1	
Na ocenę 3.0	Student zna podstawowe konstrukcje języka Python.
Na ocenę 3.5	Student zna natywne typy danych języka Python.
Na ocenę 4.0	Student zna składnię języka Python i umiejętnie wykorzystuje typy danych.
Na ocenę 4.5	Student wie jak konstuować klasy w Pythonie.

Na ocenę 5.0	Student wie jak poprawnie i modułowo zrealizować złożony projekt Pythonie wraz z obsługą błędów, oraz dobrą dokumentacją.
Efekt kształcenia 2	
Na ocenę 3.0	Student wie jak deniować procedury w Lispie.
Na ocenę 3.5	Student rozumie pojęcie programowania imperatywnego. Zna podstawy składni języka LISP. Student wie jak deniować procedury złożone.
Na ocenę 4.0	Student rozumie pojęcie programowania imperatywnego. Zna na dobrym poziomie składnię języka LISP. Student wie jak deniować procedury złożone. Zna składnię wyrażenia lambda
Na ocenę 4.5	Student rozumie pojęcie programowania imperatywnego. Zna na dobrym poziomie składnię języka LISP. Wie jakie konstrukcje językowe są adekwatne do rozważanych problemów.
Na ocenę 5.0	Student rozumie pojęcie programowania imperatywnego. Zna bardzo dobrze składnię języka LISP. Wie jakie konstrukcje językowe są adekwatne do rozważanych problemów i potra projektować systemy na nich bazujące. Student wie jakie mechanizmy abstrakcji są adekwatne do rozważanego problemu oraz potra wskazać mechanizmy i odpowiednie struktury danych do ich realizacji typowe dla języka LISP.
Efekt kształcenia 3	
Na ocenę 3.0	Student umiejętnie programuje w Pythonie z wykorzystaniem jego podstawowej składni.
Na ocenę 3.5	Student umiejętnie wykorzystuje podstawowe typy danych w Pythonie.
Na ocenę 4.0	Student umiejętnie programuje w Pythonie w modelu obiektowym.
Na ocenę 4.5	Student umie zarówno napisać poprawny program w Pythonie jak i zaprojektować testy tego oprogramowania.
Na ocenę 5.0	Student programuje poprawnie w Pythonie, testuje oraz dokumentuje działanie programów.
Efekt kształcenia 4	
Na ocenę 3.0	Student umie deniować procedury złożone.
Na ocenę 3.5	Student umie deniować procedury złożone. Umie przedstawić stosowaną i normalną kolejność obliczania danej procedury.
Na ocenę 4.0	Umie przedstawić stosowaną i normalną kolejność obliczania danej procedury. Potra formułować abstrakcje za pomocą procedur wyższego rzędu.
Na ocenę 4.5	Umie przedstawić stosowaną i normalną kolejność obliczania danej procedury. Potra formułować abstrakcje za pomocą procedur wyższego rzędu z wykorzystaniem lambda-abstrakcji.

Na ocenę 5.0	Umie przedstawić stosowaną i normalną kolejność obliczania danej procedury. Potra formułować abstrakcje za pomocą procedur wyższego rzędu z wykorzystaniem lambda-abstrakcji. potra dobrać odpowiednie struktury danych do rozważnego problemu.
Efekt kształcenia 5	
Na ocenę 3.0	Student zwraca uwagę na przejrzystość pisanego kodu
Na ocenę 3.5	Student zwraca uwagę na przejrzystość pisanego kodu oraz przestrzega odpowiednio obranych konwencji w programowaniu
Na ocenę 4.0	Student zwraca uwagę zarówno na tworzenie czytelnych jak i wydajnych programów
Na ocenę 4.5	Student odpowiednio dokumentuje kod programów
Na ocenę 5.0	Student zarówno dokumentuje kod programu jak i w sposób czytelny potra przekazać idę swojego podejścia przy rozwiązywaniu danego problemu

10 Macierz realizacji przedmiotu

Efekt kształcenia	Odniesienie danego efektu do szczegółowych efektów zdefiniowanych dla programu	Cele przedmiotu	Treści programowe	Narzędzia dydaktyczne	Sposoby oceny
EK1	I1_W04 I1_W06 I1_W11	Cel 1 Cel 2	W1 W2 W3 W4	N1	F1 F2 P1
EK2	I1_W04 I1_W06 I1_W11	Cel 3	W5 W6	N1	F2 P1
EK3	I1_U07b I1_U10 I1_U17	Cel 1 Cel 2	L1 L2 L3 L4	N2	F1 F2 P1
EK4	I1_U07b I1_U10 I1_U17	Cel 3	L5 L6	N2	F2 P1
EK5	I1_K01 I1_K04	Cel 2 Cel 3	W1 W2 W3 W4 W5 W6	N1	F2 P1

11 Wykaz literatury

Literatura podstawowa

[1] Harold Abelson and Gerald Jay Sussman Sussman — *Struktura i interpretacja programów komputerowych*, Warszawa, 2002, WNT

[2] Mark Lutz — *Learning Python*, Beijing, 2013, OReilly Media, Inc.

Literatura uzupełniająca

[2] Brian Harvey Matthew Wright — *Simply Scheme*, London, 1999, The MIT Press

[3] Daniel P. Friedman, Matthias Felleisen — *The Little Schemer*, London, 1995, The MIT Press

[4] Mark Pilgrim — *Dive into Python*, Miejscowość, 2004, Wydawnictwo

12 Informacje o nauczycielach akademickich

Osoba odpowiedzialna za kartę

dr inż. Daniel Grzonka (kontakt: `daniel.grzonka@pk.edu.pl`)

Osoby prowadzące przedmiot

1 dr inż. Tomasz Gąciarz (kontakt: `tga@pk.edu.pl`)

2 dr Adam Marszałek (kontakt: `adam.marszalek@pk.edu.pl`)

3 dr hab. inz. prof. PK Maciej Jaworski (kontakt: `maciej.jaworski@pk.edu.pl`)

4 mgr inż. Jan Wojtas (kontakt: `jan.wojtas@pk.edu.pl`)

13 Zatwierdzenie karty przedmiotu do realizacji

(miejscowość, data) (odpowiedzialny za przedmiot) (dziekan)

Przyjmuję do realizacji (data i podpisy osób prowadzących przedmiot)

. . . . . .

. . . . . .

. . . . . .

. . . . . .