Informatyka II

Funkcje i klasy w języku Python

Cel

Celem ćwiczenia jest zaznajomienie się z pojęciami i użyciem funkcji i klas w języku Python

Zadanie 1

Uruchom przeglądarkę internetową. Otwórz stronę znajdującą się pod adresem https://repl.it/languages/python3. Zapoznaj się z środowiskiem on-line.

Możesz również skorzystać z własnego komputera lub ze środowisk obecnych na komputerach w laboratorium. Upewnij się, że Twoje środowisko obsługuje język Python w wersji 3.6.1 lub nowszej.

Zadanie 2

Uruchom następujący program napisany w języku Python:

```
def wypisz(slowo1, slowo2):
    print(slowo1, slowo2)

wypisz("Argumenty", "po kolei")
wypisz(slowo2="nazwane wprost", slowo1="Lub")
```

Powyższy listing prezentuje sposób deklaracji i wywołania funkcji w języku Python. Definicję rozpoczyna słowo kluczowe ***def***. Następnie należy zdefiniować nazwę funkcji i argumenty. Nagłówek kończymy znakiem dwukropka.

Każda linia kodu, która nastąpi po wcięciu, należy do ciała funkcji. W powyższym wypadku do ciała funkcji należy jedynie wywołanie funkcji ***print***.

Funkcję stworzoną przez użytkownika wywołujemy jak dowolną inną funkcję (***print***, ***input*** itp.). Należy podać jej nazwę i – w nawiasach, oddzielone przecinkiem – wartości argumentów. Python pozwala przekazać wartości argumentów w dowolnej kolejności, o ile jasno zostanie określone których parametrów dane wartości dotyczą (zobacz: drugie wywołanie funkcji wypisz).

Zadanie 3

Uruchom następujący program napisany w języku Python:

```
def silnia(n):
  if n==1:
    return 1
  else:
    return n*silnia(n-1)

n = int(input("Podaj liczbę: "))
print("Silnia",n,"to",silnia(n))
```

Każda funkcja w języku Python może zwracać wartość dowolnego typu. Aby zwrócić wartość z funkcji, należy zastosować słowo kluczowe ***return***. Warto tutaj wspomnieć, że można zwrócić także całą listę wartości.

Zadanie 4

Uruchom następujący program napisany w języku Python:

```
class Osoba:
  imie = ""
  nazwisko = ""

poeta = Osoba()
poeta.imie = "Adam"
poeta.nazwisko = "Mickiewicz"

print(poeta.imie, poeta.nazwisko)
```

Python jest językiem wieloparadygmatowym. Jednym z jego atutów jest wsparcie dla programowania obiektowego. Klasy w języku Python możesz przyrównać do struktur z języka C. Tak jak i struktury w C, tak i klasy w Pythonie grupują pewne właściwości w jedno ciało (obiekt).

Deklarację obiektu zaczynamy od słowa kluczowego ***class***. Po nim należy umieścić nazwę klasy, listę klas-rodziców (zostało to ominięte w powyższym przykładzie i zostanie omówione w kolejnych zadaniach). Deklarację należy zakończyć znakiem dwukropka.

Instancję klasy nazywamy obiektem. Obiekt można utworzyć poprzez podanie nazwy klasy oraz nawiasów okrągłych (jak w przykładzie). Uzyskać dostęp do zmiennych obiektu można poprzez podanie nazwy obiektu (zmiennej), kropkę i nazwę pola.

Zadanie 5

Uruchom następujący program napisany w języku Python:

```
class Osoba:
  imie = ""
  nazwisko = ""

  def __init__(self,i,n):
    self.imie = i
    self.nazwisko = n

  def wypisz(self):
    print("Wielki poeta!",self.imie, self.nazwisko)

poeta = Osoba("Juliusz","Słowacki")
poeta.wypisz()
```

W przeciwieństwie do struktur języka C, elementami klasy mogą być także funkcje (funkcje składowe klasy nazywamy metodami). Klasa z przykładu implementuje dwie metody: ***__init__*** oraz ***wypisz***. Funkcję składową klasy definiujemy podobnie jak zwykłą funkcję, z tą różnicą, że pierwszym argumentem zawsze będzie obiekt, na rzecz którego metoda będzie wywoływana. Domyślnie parametr ten określa się nazwą ***self***, ale nazwa ta może być dowolna.

Specjalną funkcją jest funkcja ***__init__***. Funkcja ta jest automatycznie wywoływana przez interpreter w momencie tworzenia obiektu.

Zadanie 6

Uruchom następujący program napisany w języku Python:

```
class Pojazd:
  def __init__(self,k=4):
    self.ilośćKół = k

class Łódź:
  wyporność = 0
  def __init__(self,w=1):
    self.wyporność = w

class Amfibia(Pojazd,Łódź):
  pass

obiekt = Amfibia()
print("Ilość kół:",obiekt.ilośćKół,"wyporność:",obiekt.wyporność)
```

Zwróć uwagę na definicję klasy ***Amfibia***. W zadaniu 4 wspomniano, że przy definiowaniu klasy, można wymienić listę klas-rodziców (klas nadrzędnych). Amfibia to połączenie pojazdu lądowego i łodzi. Wiedz także, że podczas definiowania klasy, nie musimy definiować wszystkich jej pól. Skoro metoda ***__init__*** jest wywoływana w momencie tworzenia obiektu, zmienne ***ilośćKół*** i ***wyporność*** będą dostępne od momentu jego stworzenia.

Tworząc klasę, która łączy inne, już istniejące klasy, mówimy o ***dziedziczeniu***. Klasy nadrzędne nazywamy rodzicami, a klasę dziedziczącą – potomną lub dzieckiem.

Słowo kluczowe ***pass*** informuje interpreter, że do definicji klasy nie będą dodawane żadne dodatkowe elementy. Jeżeli dodajemy do klasy nową zawartość, słowo kluczowe ***pass*** można ominąć.

Zadanie 7

Uruchom następujący program napisany w języku Python:

```
class Osoba:
  def __init__(self,imie,nazwisko):
    self.imie = imie
    self.nazwisko = nazwisko

class Obywatel(Osoba):
  def __init__(self,imie,nazwisko,PESEL):
    self.PESEL = PESEL
    super().__init__(imie,nazwisko)

test = Obywatel("Adam","Kowalski","01234567890")
print(test.imie, test.nazwisko, test.PESEL)
```

Pisząc kod klasy potomnej możliwe jest dokonanie zmian w zachowaniu istniejącej metody. Mówimy wtedy o ***nadpisywaniu*** metod. W powyższym przykładzie nadpisano metodę ***__init__***. Wbudowana w język Python funkcja ***super*** pozwala nam uzyskać dostęp do klasy-rodzica.

Zadanie 8

Napisz funkcję ***policz***, która przyjmuje dwa argumenty - ciąg znaków i literę oraz zwraca ilość wystąpień litery w ciągu znaków. Podpowiedź: możesz pobrać długość ciągu znaków funkcją wbudowaną ***len(string)***. Możesz również przeglądać zawartość ciągu znaków jak każdej innej listy.

Zadanie 9

Napisz funkcję ***dzielniki***. Jedynym argumentem powinna być liczba całkowita. Funkcja powinna zwracać listę całkowitych dzielników tej liczby. Zapytaj użytkownika o liczbę i wypisz wynik działania funkcji na ekranie. Podpowiedź: w języku Python, tak jak w języku C, istnieje operator % zwracający resztę z dzielenia.

Zadanie 10

Napisz klasę ***Student***. Klasa ta powinna dziedziczyć po przedstawionej powyżej klasie ***Osoba*** i powinna dodatkowo przechowywać numer indeksu i semestr studiów. W funkcji ***init*** wywołaj ***init*** z klasy bazowej i ustaw wartość indeksu na „000000” oraz semestr studiów na 1.

Zadanie 11

Napisz klasę ***Ocena***. Klasa powinna mieć dwa pola: nazwę przedmiotu i wartość liczbową oceny. Napisz funkcję ***init*** która pozwoli nadać wartości polom klasy w momencie tworzenia obiektu. Zadbaj o to, by można było wpisać tylko dozwolone oceny (2, 3, 3.5, 4, 4.5, 5). Rozszerz klasę ***Student*** o listę ***Ocen***.

Zadanie 12

Rozbuduje klasę ***Student*** o metody: ***wpiszOcenę*** i ***średnia***. Metoda ***wpiszOcenę*** powinna pobierać dwa argumenty: nazwę przedmiotu i wartość oceny. Efektem wywołania metody powinno być dodanie do listy ocen nowej oceny. Jeżeli ocena już istnieje, powinna zostać zmieniona jej wartość. Funkcja ***średnia*** powinna zwracać średnią wszystkich ocen danego studenta.