Programowanie w Pythonie

dr Agnieszka Zbrzezny

27 września 2022

1 Argumenty funkcji – quiz

1. Co wyświetla poniższy kod i dlaczego?

```
def func(a, b = 4, c = 5):
    print(a, b, c)

func(1, 2)
```

2. Co wyświetla poniższy kod i dlaczego?

```
def func(a, b, c = 5):
    print(a, b, c)

func(1, c = 3, b = 2)
```

3. Co wyświetla poniższy kod i dlaczego?

```
def func(a, *args):
    print(a, args)

func(1, 2, 3)
```

4. Co wyświetla poniższy kod i dlaczego?

```
def func(a, **kwargs):
    print(a, kwargs)

func(a = 1, c = 3, b = 2)
```

5. Jaki będzie wynik poniższego kodu i dlaczego?

```
def func(a, b, c = 3, d = 4):
    print(a, b, c, d)

func(1, *(5, 6))
```

2 Argumenty funkcji – zadania programistyczne

1. Napisz funkcję `mult`, która przyjmuje jeden argument tylko pozycyjny. Załóż, że ten argument jest niepustym obiektem iterowalnym typu krotka, lista, zbiór lub zakres. Jako wynik funkcja zwraca iloczyn elementów swojego argumentu. W funkcji `main` przetestuj działanie funkcji `mult`.

   Przykładowo, oba poniższe wywołania

   ```
   mult([3, 5, 7]
   ```

   ```
   mult(range(2, 8, 2)
   ```

   powinny zwrócić liczbę `105`.

2. Napisz funkcję `mult_ints`, która przyjmuje jeden argument tylko pozycyjny. Załóż, że ten argument jest niepustym obiektem iterowalnym typu krotka, lista, zbiór lub zakres. Jako wynik zwraca iloczyn tych argumentów, które są typu całkowitego. W funkcji `main` przetestuj działanie funkcji `mult_ints`.

   Przykładowo, wywołanie

   `mult_ints(3, 3.14, 5, "abc", 7)` powinno zwrócić liczbę `105`.

3. Napisz funkcję `multiply`, która przyjmuje dowolną liczbę argumentów nienazwanych, a jako wynik zwraca iloczyn tych argumentów. W funkcji `main` przetestuj działanie funkcji `multiply`.

   Przykładowo, wywołanie

   `multiply(3, 5, 7)` powinno zwrócić liczbę `105`.

4. Napisz funkcję `multiply_ints`, która przyjmuje dowolną liczbę argumentów nienazwanych, a jako wynik zwraca iloczyn tych argumentów, które są typu całkowitego. W funkcji `main` przetestuj działanie funkcji `multiply_ints`.

   Przykładowo, wywołanie

   `multiply_ints(3, 3.14, 5, "abc", 7)` powinno zwrócić liczbę `105`.

5. Napisz funkcję `make_car`, która przyjmuje dwa obligatoryjne argumenty: `firma` i `model`, oraz dowolną liczbę argumentów nazwanych. Funkcja `make_car` jako swój wynik zwraca słownik opisujący konkretny samochód. Przykładowo:

   ```
   {"firma": "Kia", "model": "Picanto", "kolor": "cafe
   mocca", "poj_silnika": 900}
   ```

   W funcji `main` utwórz listę kilku samochodów opisanych przez różne możliwe cechy.

3 Tablice wielwymiarowe i macierze

3.1 Tablice dwuwymiarowe

1. Utwórz dwuwymiarową tablicę liczb całkowitych o trzech wierszach. W pierwszym wierszu tablicy umieść liczby od 1 do 10, w drugim kwadraty tych liczb, a w trzecim sześciany liczb z pierwszego wiersza. Napisz program tworzący i wyświetlający tę tablicę w konsoli.

2. W tablicy dwuwymiarowej nie wszystkie wiersze muszą mieć ten sam rozmiar. Napisz program, który utworzy tablicę liczb całkowitych o dziesięciu wierszach. Wypełnij tablicę kolejnymi liczbami naturalnymi, zaczynając od liczby 1. W pierwszym wierszu umieść jedną liczbę, w drugim dwie liczby, w trzecim trzy itd. – w dziesiątym dziesięć liczb. Oblicz sumy liczb w kolejnych wierszach i sumę wszystkich liczb zapisanych w tablicy. Wyświetl w konsoli tablicę liczb oraz obliczone sumy.

Macierz jest uporządkowaną prostokątną tablicą liczb, dla której zdefiniowane są działania algebraiczne dodawania (odejmowania) i mnożenia: Dodawanie (odejmowanie) dwóch macierzy jest możliwe tylko dla macierzy o jednakowych liczbach kolumn (n) i wierszy (m). Suma (różnica) macierzy A i B jest macierzą C taką, że $c_{ij} = a_{ij} + b_{ij}$ ($c_{ij} = a_{ij} - b_{ij}$).

Mnożenie macierzy jest możliwe, gdy liczba kolumn pierwszej macierzy (m) jest równa liczbie wierszy drugiej macierzy. Iloczyn macierzy A i B jest macierzą C taką, że

$$c_{ij} = \sum_{k=1}^{m} a_{ik} * b_{kj}.$$

Mnożenie macierzy nie jest przemienne.

Zawsze określone jest mnożenie macierzy przez liczbę λ, polegające na pomnożeniu każdego elementu macierzy przez tę liczbę ($c_{ij} = \lambda \cdot a_{ij}$).

Macierze możemy przedstawiać w programach jako listy dwuwymiarowe. Należy pamiętać, że listy są indeksowane od zera, a indeksy macierzy rozpoczynamy od jedynki.

3.2 Macierze

1. Napisz funkcję `sum(a, b)` dodającą dwie macierze. Funkcja powinna zwracać jako swój wynik macierz będącą sumą macierzy. Napisz program demonstrujący działanie funkcji `sum(a, b)`.

2. Napisz funkcję `product(a, b)` obliczającą iloczyn dwóch macierzy. Funkcja powinna zwracać jako swój wynik macierz będącą iloczynem macierzy. Napisz program demonstrujący działanie funkcji `product()`.

3. Napisz funkcję `mult(a, x)` obliczającą iloczyn macierzy przez liczbę. Funkcja powinna zwracać jako swój wynik macierz będącą iloczynem macierzy przez liczbę. Napisz program demonstrujący działanie funkcji `mult(a, x)`.

4. Napisz funkcję `transp(a)` tworzącą macierz transponowaną z macierzy podanej jako argument funkcji. Funkcja powinna zwracać jako swój wynik macierz transponowaną. Napisz program demonstrujący działanie metody `transp()`.

   Uwaga: Macierz transponowana (przestawiona) powstaje z danej macierzy poprzez zamianę jej wierszy na kolumny a kolumn na wiersze.