FUNKCJE W JĘZYKU PYTHON

Funkcja w języku Python - wyodrębniona część programu, mająca unikatową nazwę oraz ustalony sposób wymiany danych z innymi częściami programu.

Funkcja, podprogram realizujący określone zadanie może być wielokrotnie wykorzystany w tym samym programie lub w innym. Stosowanie podprogramów zwiększa przejrzystość programu.

Główne funkcje w języku Python: print(), input(). Istnieje również możliwość tworzenia własnych funkcji.

Funkcje dzielimy na zwracające wartość i niezwracające wartości. Aby zastosować funkcję należy ją wcześniej zdefiniować oraz w odpowiednim momencie wywołać aby zadziałała.

W nazwie funkcji można używać tylko liter, znaków podkreślenia i cyfr.

Funkcje powstały po to aby uniknąć powtarzalności kodu czyli zwiększyć jego czytelność.

Funkcję zwracającą wartość stosujemy, jeśli celem funkcji jest obliczenie i zwrócenie pewnej wartości do programu. Funkcja musi zawierać instrukcję return z wartością zwracaną do miejsca wywołania (np. do programu głównego)

Budowa funkcji zwracającej wartość.

```
def nazwa_funkcji(lista parametrów_formalnych):
    lista_instrukcji
    return wartość
```

Nagłówek funkcji

Treść funkcji

Funkcje definiujemy w dowolnym miejscu programu (zazwyczaj na początku), ale przed pierwszym użyciem.

Treść funkcji (lista_instrukcji oraz instrukcja return) musi być wcięta względem nagłówka.

Budowa funkcji niezwracającej wartości. Bez argumentów.

Aby wykonać treść instrukcji należy wykonać funkcję (bez parametrów w nawiasie).

```
def funkcja():
    BLOK INSTRUKCJI
```

Przykład:

```
def przywitanie():
    print("Dzień dobry")

przywitanie()
```

Budowa funkcji niezwracającej wartości. Z jednym lub wieloma argumentami.

Aby wykonać treść instrukcji należy wywołać funkcję (podać nazwę funkcji a w nawiasach wartość parametrów).

```
def funkcja(x, y):
    BLOK INSTRUKCJI
```

W definicji funkcji powyższej x i y to parametry formalne; nie mają one w tym momencie określonych wartości, są miejscami do wypełnienia konkretnymi wartościami, gdy funkcję postanowimy użyć. Liczby 3 i 4 to parametry aktualne.

Użycie czyli wywołanie funkcji to wyrażenie postaci

```
funkcja(x, y)
```

gdzie x i y mają już konkretne wartości:

```
def funkcja(x, y):
    print(x + y)

funkcja(3,4)
```

Ćwiczenie 1.

1. Napisz funkcję wypisującą treść „Moja pierwsza funkcja!”
2. Wywołaj funkcję.
3. Zapisz plik pod nazwą **pierwsza_funkcja**

```
def funkcja_1():
    print("Moja pierwsza funkcja!")

funkcja_1()
```

Ćwiczenie 2.

1. Napisz funkcję z jednym parametrem podnoszącą wybrany parametr do kwadratu.
2. Wywołaj funkcję.
3. Zapisz plik pod nazwą **kwadrat**

```
def kwadrat(a): # a to parametr formalny
    print(a ** 2)

kwadrat(5)
```

Ćwiczenie 3.

1. Napisz funkcję z dwoma parametrami, sumującą te parametry.
2. Wywołaj funkcję.
3. Zapisz plik pod nazwą **dwa_parametry**

```
def funkcja(x, y):
    print(x + y)

funkcja(3,4)
```

Ćwiczenie 4.

1. Napisz funkcję o nazwie „maksymalna”, która szuka w dwóch podanych parametrach szuka wartości maksymalnej i ją wypisuje.
2. Wywołaj funkcję
3. Zapisz plik pod nazwą **maksymalna**

```
def maksymalna(a, b):
    if a > b:
        print(a, 'to maximum')
    elif a == b:
        print(a, 'jest równe', b)
    else:
        print(b, 'to maximum')


#1 Sposób wywołania, wartości są wprowadzone bezpośrednio:
maksymalna(3, 4)

#2 Sposób wywołania, zmienne stają się argumentami:
x = 5
y = 7
maksymalna(x, y)

#3 Sposób wywołania, dowolne zmienne stają się argumentami:
x = int(input("Wpisz pierwszą liczbę: "))
y = int(input("Wpisz drugą liczbę: "))
maksymalna(x, y)
```

Ćwiczenie 5.

1. Napisz funkcję z dwoma parametrami, w efekcie której program zapyta się nas o imię i nazwisko i na tej podstawie wyświetli powitanie np. „Witaj Paweł Nowak”.
2. Zapisz plik pod nazwą **imie_nazwisko**

```
def witaj(imie,nazwisko):
    print("Witaj",imie,nazwisko)

x = input("Podaj swoje imię: ")
y = input("Podaj swoje nazwisko: ")

witaj(x,y)
```

Ćwiczenie 6.

1. Przepisz poniższe dane.
2. Zauważ relacje między danymi formalnymi a aktywnymi.

```
def dodaj(a, b = 2, c = 0):
    print(a + b + c)

dodaj(3, 4)
dodaj(3)
dodaj(3, 4, 5)
```

Parametr aktywny ma pierwszeństwo nad parametrem formalnym.

Ćwiczenie 7

1. Napisz funkcję wypisującą 5 razy „Hallo world!”
2. Wywołaj funkcję
3. Zapisz plik pod nazwą **hallo_5**

```
def powiedz(wiadomosc, ile = 1):
    print(wiadomosc * ile)

powiedz(" Hallo World! ", 5)
```

Ćwiczenie 8

1. Napisz funkcję wykonaną wielokrotnie za pomocą pętli for.
2. Zapisz plik pod nazwą **witaj_for**

```
def witaj():
    print("Hallo world!")
for i in range (10):
    witaj()
```

Ćwiczenie 9

1. Napisz program (na podstawie funkcji) obliczający pole kwadratu po długości boku.
2. Zapisz plik pod nazwą **pole_kwadratu**

```
def pole_kwadratu(dlugosc_boku):
    pole = dlugosc_boku**2
    print(pole)
x = int(input("Podaj długość boku kwadratu: "))
pole_kwadratu(x)
```

Ćwiczenie 10

1. Napisz 2 programy, z parametrem i bez (na podstawie funkcji), wyświetlający 20 gwiazdek (*******)
2. Zapisz program pod nazwą **gwiazdki**

Ćwiczenie 11

1. Zmodyfikuj powyższy program tak aby użytkownik wprowadził z klawiatury ilość gwiazdek
2. Zapisz program pod nazwą **n_gwiazdki**

Ćwiczenie 12

1. Napisz program (na podstawie funkcji) obliczający pole prostokąta
2. Zapisz program pod nazwą **poleprostokąta**

Ćwiczenie 13

1. Napisz program (na podstawie funkcji bez parametrów) wyświetlający wynik jak poniżej (za pomocą pętli).

```
= 1
= 2
= 3
= 4
= 5
= 6
= 7
= 8
= 9
= 10
```

2. Zapisz program pod nazwą **1_10**