Funkcje

- Funkcja jest narzędziem grupującym zbiór instrukcji w taki sposób, aby mogły one być wykonane w programie więcej niż jeden raz.
- Funkcje są alternatywą dla programowania polegającego na kopiowaniu i wklejaniu. Zamiast tworzyć kilka zbędnych kopii kodu operacji, możemy wykorzystać jedną funkcję. W taki sposób zmniejszamy również nakład pracy, jaki trzeba by było poświęcić w przyszłości, gdyby operacja musiała zostać zmodyfikowana. Dzięki funkcji kod należy uaktualnić tylko w jednym miejscu, a nie w kilku.
- Funkcje są najważniejszą strukturą programu w Pythonie, służącą do maksymalizowania możliwości ponownego wykorzystania kodu i minimalizowania powtarzalności kodu.
- Funkcje są również narzędziem projektowym pozwalającym na rozbicie złożonych systemów na części, którymi łatwiej jest zarządzać.

Funkcje

- Funkcje pojawiły się już na wcześniejszych slajdach. Na przykład używaliśmy funkcji wbudowanej **len** do obliczenia liczby elementów listy. Inny przykład: funkcja **sqrt** z modułu math.
- Teraz dowiemy się, jak w Pythonie pisze się nowe funkcje. Funkcje pisane przez nas zachowują się w ten sam sposób jak prezentowane już funkcje wbudowane - są wywoływane w wyrażeniach, przekazuje się do nich wartości i zwraca się z nich wyniki.

Funkcje

Definiowanie funkcji

def nazwa_funkcji**(par1, par2, ..., parN):**
    instrukcje_do_wykonania

- Polecenie **def** (skrót od ang. define) pozwala zdefiniować funkcję.
- Następnie pojawia się nazwa funkcji.
- Kolejno piszemy parę nawiasów okrągłych, które mogą zawierać nazwy zmiennych (tzn. parametry przekazywane do funkcji). Na końcu stawiamy dwukropek.
- Poniżej piszemy blok instrukcji, będący treścią funkcji.

Funkcje

Program wypisujący tekst "Witaj w świecie Pythona".

```
print("Witaj w świecie Pythona")
```

Ten sam program z funkcją

```
#definicja funkcji
def hello():
    print("Witaj w świecie Pythona")

#wywołanie funkcji
hello()
```

Funkcja hello nie przyjmuje żadnych parametrów (nawias okrągły jest pusty) i nie zwraca żadnej wartości.

Przykład - bez funkcji

```
a = int(input("Podaj pierwszą liczbę"))
b = int(input("Podaj drugą liczbę"))
s = a + b
print(s)
```

Przykład - z funkcją (wersja 1)

```
a = int(input("Podaj pierwszą liczbę"))
b = int(input("Podaj drugą liczbę"))

#definicja funkcji; funkcja przyjmuje z zewnątrz dwa parametry x i y
def suma(x, y):
    s = x + y
    return s

#wywołanie funkcji i przypisanie wartości przez nią zwracanej do zmiennej wynik
wynik = suma(a, b)

print(wynik)
```

Przykład - z funkcją (wersja 1)

```
a = int(input("Podaj pierwszą liczbę"))
b = int(input("Podaj drugą liczbę"))
def suma(x, y):
    s = x + y
    return s
wynik = suma(a, b)
print(wynik)
```

Przykład - z funkcją (wersja 2: skrócona)

```
a = int(input("Podaj pierwszą liczbę"))
b = int(input("Podaj drugą liczbę"))
def suma(x, y):
    return x + y
print(suma(a, b))
```

Symbol Newtona: $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

Program bez funkcji

```
n = int(input("Podaj n"))
k = int(input("Podaj k"))

sn = 1
for i in range(2, n+1):
    sn = sn * i

sk = 1
for i in range(2, k+1):
    sk = sk * i

snk = 1
for i in range(2, n-k+1):
    nk = snk * i

w = sn/(sk*snk)
print(w)
```

Program z funkcją

```
n = int(input("Podaj n"))
k = int(input("Podaj k"))

def silnia(p):
    s=1
    for i in range (2, p+1):
        s = s * i
    return s

wynik = silnia(n)/(silnia(k)*silnia(n-k))
print(wynik)
```

Zad. 56. Napisz funkcję obliczającą euklidesową odległość pomiędzy dwoma punktami. Współrzędne punktów należy przekazać do funkcji jako parametry.

Zad. 57. Napisz funkcję, która zwraca większą z dwóch przekazanych jako jej parametry liczb.

Zad. 58. Napisz funkcję, która dla zadanych liczb rzeczywistych a, b, c zwraca -1, jeśli nie mogą być one długościami boków trójkąta, lub pole trójkąta o bokach długości a, b, c.

Zad. 59. Napisz funkcję, która ma trzy parametry a, b, c będące liczbami całkowitymi. Wartością funkcji jest *true*, jeśli zadane liczby są liczbami pitagorejskimi oraz *false* w przeciwnym wypadku.

Zad. 60. Napisz funkcję, która zwraca największy wspólny dzielnik dwóch liczb całkowitych przekazanych jako parametry.

Uwaga: Napisane funkcje należy wywołać i sprawdzić poprawność ich działania.

Zad. 61. Napisz funkcję, która sprawdza, czy przekazana w parametrze liczba jest liczbą pierwszą.

Zad. 62. Napisz funkcję, która podaną przez użytkownika kwotę pieniędzy (liczba całkowita) rozmienia na jak najmniejszą ilość monet i banknotów o nominałach $1, 2, 5, 10$ złotych.

Przykład dla kwoty 188 zł:
188 zł rozmieniamy na
18 banknotów 10 zł
1 monetę 5 zł
1 monetę 2 zł
1 monetę 1 zł.

Uwaga: Napisane funkcje należy wywołać i sprawdzić poprawność ich działania.