Nazwa implementacji: Nauka języka Python – funkcje

Autor: Piotr Fiorek

Opis implementacji: Nauka tworzenia oraz używania funkcji w języku C

Do tej pory korzystaliśmy tylko z gotowych funkcji, których Python dostarcza całe mnóstwo. Czas jednak wreszcie zacząć tworzyć własne funkcje. Jest to bardzo proste i jednocześnie bardzo przyśpiesza tworzenie nowych programów, ponieważ raz napisany kod można bardzo łatwo i szybko wykorzystać ponownie.

Nową funkcję zaczynamy od zadeklarowania jej nazwy oraz parametrów, jakie funkcja będzie pobierać, jeśli w ogóle jakieś ma pobierać:

```
>>> def nowa_funkcja():

... pass
```

Tak wygląda najprostsza definicja nowej funkcji. Słówko „def” oznacza właśnie, że chcemy zdefiniować funkcję, „nowa_funkcja” jest nazwą naszej funkcji, a w nawiasy po nazwie wpisuje się oddzielone przecinkami nazwy parametrów, które nasza funkcja będzie pobierać. W tym przykładzie pojawiło się jeszcze nowe słówko kluczowe „pass” oznaczające „nic nie rób”. Zatem tak zdefiniowana funkcja jest całkowicie poprawną funkcją, ale nic nie robi, więc na nic nam się nie przyda. Zdefiniujmy zatem funkcję, która będzie pobierała dwie zmienne jako parametry i będzie zwracała sumę tych zmiennych:

```
>>> def suma(x, y):

... z = x + y

... return z

...
```

W ten sposób stworzyliśmy funkcję o nazwie „**suma**”, która pobiera zmienne o nazwach „

x

” i „y” (nazwy parametrów, które podajemy podczas definiowania funkcji, obowiązują tylko wewnątrz funkcji). Kod funkcji dodaje do siebie te dwie zmienne, wynik zapisując w zmiennej „z” i następnie zwraca jej zawartość, abyśmy mogli go wykorzystać dalej w naszym kodzie, który wywoła tą funkcję. A wywołanie, czyli użycie tej funkcji może wyglądać tak:

```
>>> a = 5

>>> b = 2

>>> c = suma(a, b)

>>> print(c)

7

>>> suma(a, 3)

8

>>> suma(6, a)
1
```


Projekt “Strategia Wolnych i Otwartych Implementacji jako innowacyjny model zainteresowania kierunkami informatyczno-technicznymi oraz wspierania uczniów i uczennic w kształtowaniu kompetencji kluczowych” współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

```
11

>>> suma(c, b)

9
```

Jak widać naszą funkcję, wywołujemy tak samo jak każdą inną, czyli używając jej nazwy, a w nawiasy wpisując parametry. Parametrami funkcji mogą być zarówno zmienne, jak i stałe, czyli w tym przypadku po prostu cyfry.

Treść funkcji może być dowolna i może wykorzystywać dowolne konstrukcje językowe. Zwracać funkcja też może dowolne typy danych. Może to być zarówno pojedyncza zmienna dowolnego typu, jak i zmienne bardziej złożonych typów danych jak krotki czy listy, które w praktyce używane są bardzo często, jeśli funkcja ma zwrócić więcej niż jedną wartość. Spróbujmy więc przepisać nasz kalkulator tak aby używał funkcji:

```
def dodawanie(x, y):

return x + y

def odejmowanie(x, y):

return x - y

def mnozenie(x, y):

return x * y

def dzielenie(x, y):

return x / y

def potegowanie(x, y):

return x ** y

def pobierz_zmienne():

zmienna1 = float(input('Podaj 1 liczne: '))

zmienna2 = float(input('Podaj 2 liczne: '))

return (zmienna1, zmienna2)

def kalkulator():

print('Jaką operację chcesz wykonać?\n', '1) dodawanie\n', '2) odejmowanie\n', '3) mnożenie\n', '4) dzielenie\n', '5) potęgowa

wybor = int(input('Wpisz cyfrę operacji: '))

a, b = pobierz_zmienne()

if wybor == 1:

2
```

```
wynik = dodawanie(a, b)

elif wybor == 2:

wynik = odejmowanie(a, b)

elif wybor == 3:

wynik = mnozenie(a, b)

elif wybor == 4:

if b == 0:

print('\nDzielenie przez zero jest zabronione!\n')

exit()

else:

wynik = dzielenie(a, b)

elif wybor == 5:

wynik = potegowanie(a, b)

else:

print('\nWybrałeś nieistniejącą opcję!\n')

exit()

print('Wynik: ', wynik)

def main():

while True:

kalkulator()

pytanie = input('Wyjść z programu? (t/n): ')

if pytanie == 't':

break

elif pytanie == 'n':

continue

print('Podałeś błędną literę!')

main()
```


KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
ORE
FWIOO
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt “Strategia Wolnych i Otwartych Implementacji jako innowacyjny model zainteresowania kierunkami informatyczno-technicznymi oraz wspierania uczniów i uczennic w kształtowaniu kompetencji kluczowych” współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Efekt może wyglądać np. tak, jak powyżej gdzie cały kod został podzielony na funkcje, łącznie z głównym kodem programu, który został umieszczony w osobnej funkcji. W takiej sytuacji cały kod wczytywany jest jako funkcje i wywoływany dopiero w momencie, kiedy interpreter dojdzie do ostatniej linii, czyli wywołania funkcji „main”. Wewnątrz funkcji „main” wywoływany jest kod naszego kalkulatora, który z kolei wywołuje kolejne funkcje do realizacji poszczególnych zadań.

Zadania:

Napisz funkcję ,która przyjmuje dwa parametry i zwraca większy z nich.

Napisz funkcję, która jako parametr przyjmuje tablicę i zwraca największy element z tablicy.

- **Napisz funkcję, która przyjmuje dwie tablice i zwraca sklejenie tych tablic.**

BONUS (domknięcia)– Napisz funkcję, która przyjmuje parametr, stosuje go do funkcji zadeklarowanej wewnątrz tej funkcji i zwraca tak powstałą funkcję.

KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
ORE
FWIOO
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt “Strategia Wolnych i Otwartych Implementacji jako innowacyjny model zainteresowania kierunkami informatyczno-technicznymi oraz wspierania uczniów i uczennic w kształtowaniu kompetencji kluczowych” współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.