Politechnika Wrocławska

Wstęp do programowania

INP003203L

rok akademicki 2020/21

semestr zimowy

Laboratorium 4

Karol Tarnowski

karol.tarnowski@pwr.edu.pl

L-1 p. 220

Politechnika Wrocławska

Plan prezentacji (1)

- Funkcja print() – ciąg dalszy
  - argumenty nazwane
  - konkatencja napisów
  - formatowanie liczb
- Stałe nazwane

Na podstawie: T. Gaddis, *Python dla zupełnie początkujących*, Helion, 2019

Politechnika Wrocławska

Plan prezentacji (2)

- Funkcje
  - argumenty nazwane
  - funkcje zwracające wartość
  - tabele IPO (input, processing, output)
  - zwracanie ciągu tekstowego
  - zwracanie kilku wartości

Na podstawie: T. Gaddis, *Python dla zupełnie początkujących*, Helion, 2019

Politechnika Wrocławska

Funkcja print()

print03.py

File Edit Format Run Options Window Help

```
#Program pokazuje wypisywanie kilku wierszy przy użyciu funkcji print()
def main():
    print('Raz') #po wypisaniu napisu kursor jest przenoszony
    print('Dwa') #do nowego wiersza
    print('Trzy')

main()
```

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/karol/Desktop/201920/python-examples/print/print03.py =
=
Raz
Dwa
Trzy
>>>
```

Funkcja print()

print04.py

File Edit Format Run Options Window Help

```
#Program pokazuje jak zmienić domyślne zachowanie funkcji print()
#Napisy będą wyświetlone w tym samym wierszu, rozdzielone spacjami
def main():
    print('Raz', end=' ') #po wypisaniu napisu kursor nie jest przenoszony
    print('Dwa', end=' ') #do nowego wiersza, jest wstawiana spacja
    print('Trzy')

main()
```

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/karol/Desktop/201920/python-examples/print/print04.py =
=
Raz Dwa Trzy
>>>
```

Funkcja print()

print05.py

File Edit Format Run Options Window Help

```
#Program pokazuje jak zmienić domyślne zachowanie funkcji print()
#Napisy będą wyświetlone w tym samym wierszu.
def main():
    print('Raz', end='') #po wypisaniu napisu kursor pozostaje w miejscu
    print('Dwa', end='') #kolejne wywołanie wypisuje następny napis
    print('Trzy')

main()
```

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/karol/Desktop/201920/python-examples/print/print05.py =
=
RazDwaTrzy
>>>
```

Politechnika Wrocławska

Funkcja print()

print06.py

File Edit Format Run Options Window Help

```
#Program pokazuje domyślne zachowanie funkcji print()
# wywoływanej z kilkoma argumentami.
def main():
    print('Raz','Dwa','Trzy') #napisy są rozdzielone spacjami

main()
```

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/karol/Desktop/201920/python-examples/print/print06.py =
=
Raz Dwa Trzy
>>>
```

Funkcja print()

print07.py

File Edit Format Run Options Window Help

```
#Program pokazuje jak zmienić domyślne zachowanie funkcji print()
# wywoływanej z kilkoma argumentami.
def main():
    print('Raz','Dwa','Trzy',sep='***') #napisy są rozdzielone gwiazdkami

main()
```

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/karol/Desktop/201920/python-examples/print/print07.py =
=
Raz***Dwa***Trzy
>>>
```

Funkcja print()

Znaki sterujące

print08.py

File Edit Format Run Options Window Help

```
#Program pokazuje wykorzystanie znaków sterujących \n, \t, \', \", \\
def main():
    print('To jest napis') #Funkcja print() wyświetli napis

    #Znak sterujący \n powoduje przeniesienie kursora do nowej linii
    print('To\njest\nnapis')

    #Znak sterujący \t powoduje przejście do następnego położenia tabulatora
    print('To\tjest\tnapis')
    #Może to być pomocne przy formatowaniu nagłówka tabeli
    print('pn\twt\tśr\tcz\tpt')

    #Znaki specjalne \' oraz \" pozwalają uzyskać apostrof i cudzysłów
    print('Apostrof: \' ')
    print('Cudzysłów: \" ')

    #Skoro znaki sterujące wykorzystują ukośnik
    #To jak uzyskać ukośnik?
    print('Ukośnik: \\')

main()
```

Funkcja print()

Znaki sterujące

print08.py

File Edit Format Run Options Window Help

```
#Program pokazuje wykorzystanie znaków sterujących \n, \t, \', \", \\
def main():
    print('To jest napis') #Funkcja print() wyświetli napis

    #Znak sterujący \n powoduje przeniesienie kursora do nowej linii
    print('To\njest\nnapis')

    #Znak sterujący \t powoduje przejście do następnego położenia tabulatora
    print('To\tjest\tnapis')
    #Może to być pomocne przy formatowaniu nagłówka tabeli
    print('pn\twt\tśr\tcz\tpt')

    #Znaki specjalne \' oraz \" pozw
    print('Apostrof: \' ')
    print('Cudzysłów: \" ')

    #Skoro znaki sterujące wykorzyst
    #To jak uzyskać ukośnik?
    print('Ukośnik: \\')

main()
```

```
=
To jest napis
To
jest
napis
To      jest    napis
pn      wt      śr      cz      pt
Apostrof: '
Cudzysłów: "
Ukośnik: \
>>>
```

Funkcja print()

Konkatencja napisów

print09.py

File Edit Format Run Options Window Help

```
#Program pokazuje konkatencję ciągów tekstowych.

def main():
    #Dwa napisy zostaną połączone w jeden.
    print('To jest ' + 'jeden ciąg tekstowy.')

    #Trzy napisy zostaną połączone, mimo tego,
    #że znajdują się w kilku liniach.
    print('To jest bardzo długi napis, ' +
          'który trudno byłoby zmieścić ' +
          'w jednej linii.')

main()
```

```
To jest jeden ciąg tekstowy.
To jest bardzo długi napis, który trudno byłoby zmieścić w jednej
linii.
>>>
```

Funkcja print()

Formatowanie liczb

print10.py

File Edit Format Run Options Window Help

```
#Program pokazuje domyślne formatowanie
#liczby zmiennoprzecinkowej.

def main():
    kredyt = 5000.
    rata_miesieczna = kredyt / 12
    print('Miesięczna rata wynosi', rata_miesieczna, 'zł.')

main()
```

```
Miesięczna rata wynosi 416.6666666666667 zł.
>>>
```

Politechnika Wrocławska

Funkcja print()

Formatowanie liczb

```
>>> print(format(12345.6789, '.2f'))
12345.68
>>> print(format(12345.6789, '.1f'))
12345.7
>>> print('Liczba to',format(1.234567, '.2f'))
Liczba to 1.23
>>>
```

Funkcja print()

Formatowanie liczb

print11.py

File Edit Format Run Options Window Help

```
#Program pokazuje formatowanie liczby zmiennoprzecinkowej
#przy użyciu funkcji format().

def main():
    kredyt = 5000.
    rata_miesieczna = kredyt / 12
    #liczba zmiennoprzecinkowa (f) jest zaokrąglona
    #do dwóch (.2f) cyfr po przecinku
    print('Miesięczna rata wynosi',
          format(rata_miesieczna, '.2f'),
          'zł.')

main()
```

```
Miesięczna rata wynosi 416.67 zł.
>>>
```

Politechnika Wrocławska

Funkcja print()

Notacja naukowa

```
>>> print(format(12345.6789,'e'))
1.234568e+04
>>> print(format(12345.6789,'.2e'))
1.23e+04
>>>
```

Politechnika Wrocławska

Funkcja print()

Ustalenie szerokości pola

```
>>> print('Liczba to', format(12345.6789, '12.2f'))
Liczba to     12345.68
>>>
```

Funkcja print()

Funkcja format() - przykłady

print12.py

File Edit Format Run Options Window Help

```
#Program pokazuje efekt wyświetlenia kilku liczb
#zmiennoprzecinkowych w kolumnie według wspólnego formatu.
#Liczby są wyrównane względem przecinka dziesiętnego.

def main():
    num1 = 127.899
    num2 = 3465.148
    num3 = 3.776
    num4 = 264.821
    num5 = 88.081
    num6 = 799.999

    form = '7.2f'

    print(format(num1,form))
    print(format(num2,form))
    print(format(num3,form))
    print(format(num4,form))
    print(format(num5,form))
    print(format(num6,form))

main()
```

```
 127.90
3465.15
   3.78
 264.82
  88.08
 800.00
>>>
```

Politechnika Wrocławska

Funkcja print()

Formatowanie wartości procentowej

```
>>> print(format(0.5,'%'))
50.000000%
>>> print(format(0.5,'.0%'))
50%
>>>
```

Politechnika Wrocławska

Funkcja print()

Formatowanie liczb całkowitych

```
>>> print(format(123456,'d'))
123456
>>> print(format(123456,'10d'))
    123456
>>>
```

Politechnika Wrocławska

Stałe nazwane

const.py

File Edit Format Run Options Window Help

```
#Program pokazuje przykład sytuacji,
#w której przydatne jest użycie stałej.

def main():
    kredyt = 5000.
    odsetki = kredyt * 0.054
    print('Odsetki wynoszą ',odsetki)

main()
```

const01.py - C:\Users\karol\Desktop\201920\python-examples\stałe\const01.py (3.7.4)

File Edit Format Run Options Window Help

```
#Program pokazuje użycie stałej.

OPROCENTOWANIE = 0.054

def main():
    kredyt = 5000.
    odsetki = kredyt * OPROCENTOWANIE
    print('Odsetki wynoszą ',odsetki)

main()
```

Funkcje

Argumenty nazwane

show_interest_01.py

File Edit Format Run Options Window Help

```
#Ten program pokazuje przykład wywołania funkcji
#z użyciem argumentów nazwanych.

#Funkcja główna wywołuje funkcję wyświetlającą obliczony wynik.
def main():
    #W tym wywołaniu argumenty przyjmują wartości na podstawie pozycji.
    show_interest(1000,0.03,3)

    #W tym wywołaniu argumenty przyjmują wartości zgodnie z nazwą.
    #Kolejność nie ma znaczenia.
    show_interest(rate = 0.03, periods = 3, deposit = 1000)

#Funkcja wyświetlająca wysokość odsetek należnych od zdeponowanej kwoty
#w zależności od czasu trwania depozytu oraz wysokości oprocentowania.
def show_interest(deposit, rate, periods):
    interest = deposit * rate * periods
    print('Wysokość odsetek wynosi',
          format(interest, '.2f')
          )

#Wywołanie funkcji głównej
main()
```

Funkcje

Argumenty nazwane

show_interest_02.py

File Edit Format Run Options Window Help

```
#Ten program pokazuje przykład wywołania funkcji
#z użyciem argumentów nazwanych oraz .

#Funkcja główna wywołuje funkcję wyświetlającą obliczony wynik.
def main():
    #W tym wywołaniu argumenty przyjmują wartości na podstawie pozycji.
    show_interest(1000,0.03,3)

    #W tym wywołaniu argumenty przyjmują wartości zgodnie z nazwą.
    #Kolejność nie ma znaczenia.
    show_interest(deposit = 1000, rate = 0.03, periods = 3)
    show_interest(rate = 0.03, periods = 3, deposit = 1000)

    #W tym wywołaniu pierwszy argument przyjmuje wartość dzięki pozycji.
    #Kolejne na podstawie nazwy.
    show_interest(1000, periods = 3, rate = 0.03)

#Funkcja wyświetlająca wysokość odsetek
#należnych od zdeponowanej kwoty
#w zależności od czasu trwania depozytu
#oraz wysokości oprocentowania.
def show_interest(deposit, rate, periods):
    interest = deposit * rate * periods
    print('Wysokość odsetek wynosi',
          format(interest, '.2f')
          )

#Wywołanie funkcji głównej
main()
```

Politechnika Wrocławska

Funkcje

Funkcje zwracające wartość

- Ogólna postać funkcji zwracającej wartość

```
def nazwa_funkcji():
    polecenie
    polecenie
    polecenie
    itd.
    return wyrażenie
```

Funkcje

Funkcje zwracające wartość

return_result_01.py

File Edit Format Run Options Window Help

```
1  #Program pokazuje definicję i wywołanie funkcji zwracajacej wartość.
2
3  #Program pyta użytkownika o liczbę jabłek i pomarańczy.
4  #Następnie wywołuje funkcję sum() do obliczenia łącznej liczby owoców
5  #i wyświetla stosowny komunikat.
6  def main():
7      apples  = int(input('Ile masz jabłek? '))
8      oranges = int(input('Ile masz pomarańczy? '))
9      print('Liczba wszystkich owoców',sum(apples,oranges))
10
11 #Funkcja sum oblicza sumę dwóch liczb
12 def sum(num1, num2):
13     result = num1 + num2 #zapisanie wyniku w zmiennej pomocniczej
14     return result        #zwrócenie wyniku
15
16 main()
17
```

Funkcje

Funkcje zwracające wartość

return_result_02.py

File Edit Format Run Options Window Help

```
1  #Program pokazuje definicję i wywołanie funkcji zwracajacej wartość.
2
3  #Program pyta użytkownika o liczbę jabłek i pomarańczy.
4  #Następnie wywołuje funkcję sum() do obliczenia łącznej liczby owoców
5  #i wyświetla stosowny komunikat.
6  def main():
7      apples  = int(input('Ile masz jabłek? '))
8      oranges = int(input('Ile masz pomarańczy? '))
9      print('Liczba wszystkich owoców',sum(apples,oranges))
10
11 #Funkcja sum oblicza sumę dwóch liczb
12 def sum(num1, num2):
13     return num1 + num2 #polecenie return może zwracać wartość wyrażenia
14
15 main()
16
```

Funkcje

Funkcje zwracające wartość

return_result_03a.py

File Edit Format Run Options Window Help

```
#Przykład pokazuje funkcję, która prosi użytkownika o wprowadzenie
#danych, a następnie zwraca otrzymany wynik.

def get_regular_price():
    price = float(input('Podaj cenę detaliczną produktu: '))
    return price

def main():
    reg_price = get_regular_price()
    print(reg_price)

main()
```

Funkcje

Funkcje zwracające wartość

return_result_03b.py

File Edit Format Run Options Window Help

```
#Przykład pokazuje funkcję, która oblicza zniżkę, na podstawie
#ceny podstawowej (procent znikżki zapisany w stałej).

DISCOUNT = 0.20

def discount(price):
    return price*DISCOUNT

def main():
    price = 100
    print('Zniżka:', discount(price))

main()
```

Funkcje

Funkcje zwracające wartość

return_result_03c.py

File Edit Format Run Options Window Help

```
#Program oblicza promocyjną cenę produktu. Na podstawie ceny podstawowej
#podanej przez użytkownika. Program wykorzystuje dwie funkcje:
# discount(price)
# get_regular_price()

#Stała DISCOUNT zawiera zniżkę.
DISCOUNT = 0.20

#Funkcja main() zawierająca logikę główną programu
def main():
    #Pobranie podstawowej ceny produktu
    reg_price = get_regular_price()
    #Obliczenie ceny promocyjnej
    sale_price = reg_price - discount(reg_price)
    #Wyświetlenie ceny promocyjnej
    print('Cena promocyjna wynosi ',
          format(sale_price,'.2f'), ' zł.',sep = '')

#Funkcja discount() przyjmuje jako argument podstawową
#cenę produktu i zwraca wysokość rabatu obliczoną
#na podstawie zniżki (stała DISCOUNT)
def discount(price):
    return price*DISCOUNT

#Funkcja get_regular_price() prosi użytkownika o podanie
#ceny podstawowej produktu i zwraca wartość
def get_regular_price():
    price = float(input('Podaj cenę detaliczną produktu: '))
    return price


main()
```

Politechnika Wrocławska

Funkcje

Tabele IPO

- Efektywny narzędziem używanym do projektowania programu są tabele IPO (input, processing, output) – dane wejściowe, przetwarzanie danych, dane wyjściowe.

Funkcje

Tabele IPO

funkcja `get_regular_price()`		
wejście	przetwarzanie	wyjście
brak	prosi użytkownika o podanie ceny podstawowej produktu	cena podstawowa produktu

funkcja `discount(price)`		
wejście	przetwarzanie	wyjście
cena podstawowa	oblicza wartość rabatu mnożąc cenę produktu przez stałą DISCOUNT	wartość rabatu

Funkcje

Zwracanie ciągu tekstowego

return_string.py

File Edit Format Run Options Window Help

```
#Przykładowy program wywołujący funkcję zwracającą łańcuch tekstowy

#Funkcja główna programu
def main():
    name = get_name()        #wczytanie imienia
    print('Witaj,',name)     #wyświetlenie powitania

#Funkcja get_name()
#  wejście:       brak
#  przetwarzanie: funkcja prosi użytkownika o podanie imienia
#  wyjście:       imię (jako łańcuch znakowy)
def get_name():
    name = input('Podaj imię: ')
    return name

main()
```

Funkcje

Zwracanie kilku wartości

return_tuple.py

File Edit Format Run Options Window Help

```
#Przykładowy program wywołujący funkcję zwracającą kilka wartości.

#Funkcja główna programu
def main():
    first_name, last_name = get_name()  #wczytanie imienia i nazwiska
    #wyświetlenie powitania (operator + użyty do połączenia łańcuchów)
    print('Witaj,', first_name + ' ' + last_name )

#Funkcja get_name()
#  wejście:       brak
#  przetwarzanie: funkcja prosi użytkownika o podanie imienia
#                 funkcja prosi użytkownika o podanie nazwiska
#  wyjście:       imię i nazwisko (jako dwa łańcuchy znakowe)
def get_name():
    first = input('Podaj imię: ')
    last  = input('Podaj nazwisko: ')
    return first, last

main()
```

Politechnika Wrocławska

Absolutne minimum

- Funkcja print()
  - argumenty nazwane: end oraz sep
  - wykorzystanie funkcji format() do formatowania liczb
- Stałe nazwane
  - unikaj stosowania „magicznych liczb”
- Funkcje
  - dwa sposoby przekazywania argumentów do funkcji (pozycyjnie i według nazwy)
  - funkcje zwracające wartość
  - tabele IPO – opisu przepływ danych przez funkcje