



Technical University of Lodz
Institute of Electronics

Algorytmy i struktury danych

3. Elementy programu i typy danych

Łódź 2013





Ćwiczenie – Body Mass Index

- Edytuj kod programu
- Zapisz go w pliku **bmi.py**
- Uruchom skrypt

```
1  # BMI.py
2
3  YourName = input( "Enter your name: " )
4  var = input ( "Enter your height [in cm]: " )
5  YourHeight = var / 100.0
6  YourMass = input ( "Enter the mass of your body [in kg]: " )
7  BMI = YourMass / ( YourHeight * YourHeight)
8  print ("\n")
9  print ("Hello " + YourName + "!")
10 print "Your Body Mass Index is %4.1f" % ( BMI )
11
```

>>> Enter your name: "Jacek"

>>> Enter your height [w cm]: 185

>>> Enter the mass of your body [w kg]: 87



Zmienne

- **Zmienne** są nazwami i odnoszą się do danych zapisanych w pamięci.
- Zmienne są także nazywane w Pythonie **identyfikatorami**.

```
1  # BMI.py
2
3  YourName = input( "Enter your name: " )
4  var = input ( "Enter your height [in cm]: " )
5  YourHeight = var / 100.0
6  YourMass = input ( "Enter the mass of your body [in kg]: " )
7  BMI = YourMass / ( YourHeight * YourHeight)
8  print ("\n")
9  print ("Hello " + YourName + "!")
10 print "Your Body Mass Index is %4.1f" % ( BMI )
11
```

- W programie *BMI.py* znajduje się 5 zmiennych:
YourName, var, YourHeight, YourMass, BMI



Nazwy zmiennych

Nazwy zmiennych

- mogą się składać z małych i wielkich liter alfabetu łacińskiego; podkreślnika _ oraz cyfr 0, 1, ..., 9;
- nie mogą zaczynać się od cyfry;
- nie mogą być żadnym ze słów kluczowych (**keywords**) Pythona

and	del	from	not	while
as	elif	global	or	with
assert	else	if	pass	yield
break	except	import	print	
class	exec	in	raise	
continue	finally	is	return	
def	for	lambda	try	

Aby łatwiej śledzić treść programu, definiuj nazwy tak, by miały znaczenie.



Instrukcje programu

- Program jest ciągiem *instrukcji*.
- W programie *BMI.py* występują dwa rodzaje instrukcji:
instrukcja **podstawienia** i instrukcja **print()**.

Instrukcja **podstawienia**: `<zmienna> = <wyrażenie>`

Instrukcje podstawienia czyta się od prawej do lewej:

- 1) Oblicz wartość wyrażenia po prawej stronie.
- 2) Przypisz zmienną o nazwie z lewej strony do wartości ze strony prawej.

```
>>> #Oblicza wartość wyrażenia po prawej stronie (-2) i przypisuje x do -2  
>>> x = 10 - 17 + 5
```



Znak "=" w instrukcji przypisania nie jest znakiem równości.
Pełni on raczej funkcję separatora zmiennej od wyrażenia.



Instrukcje programu

Instrukcje **print**:

```
print <wyrażenie1>, <wyrażenie2>, ...
```

są używane do wyświetlania wyników obliczeń za pomocą programu.

>>> # Wyrażenia w cudzysłowach (tzw. łańcuchy) są wyświetlane dosłownie.

>>> print "Ala ma kota."

>>> print 'Ala ma kota.'

>>> # Print wyprowadza wartości wyrażeń bez cudzysłowów.

>>> n = 5

>>> print 'Ala ma kota oraz', n-3, ' psy.'

- Wyrażenia są oddzielone pojedynczymi spacjami.
- Każda instrukcja print wyświetla wynik w oddzielnym wierszu.



Instrukcje programu

- Edytuj, zapisz i uruchom następujący skrypt

```
1  # tmp.py
2
3  from math import pi
4  r = 12
5  area = pi * r ** 2
6  print "The area of a circle with radius", r, "is", area
7
```

- Zmień instrukcję **print()** do postaci

>>> print "The area of a circle with radius r is area"

Objasnij rezultat tej zmiany.

- Technicznie, **print()** jest wbudowaną **funkcją**, która jest wywoływana w celu umieszczenia instrukcji print w programie.



Typy danych

- Zmienna w Pythonie może się odnosić do danych dowolnego typu.
Trzy przykłady typów danych to **łańcuch (string)**, liczba całkowita (**integer**) i liczba zmiennoprzecinkowa (**float**).

Łańcuchy są ciągami znaków wewnątrz pojedynczych lub podwójnych cudzysłówów.

```
>>> Motto = "Programming is fun!"  
>>> aQuestion = 'Does she like studying at Harvard?'
```

Liczby całkowite to, na przykład:

```
>>> Large_Score = 857  
>>> NegativeResult = -39
```

Liczby zmiennoprzecinkowe posiadają część ułamkową:

```
>>> DistanceInMeters = 857.21  
>>> SmallQuantity = -0.000138
```



Wyrażenia

Wyrażenie

`input(prompt)`

służy do wprowadzania danych. Wyświetla znak zachęty (**prompt**) i zwraca wyrażenie wprowadzone przez użytkownika. Znak zachęty jest wyświetlany by przypomnieć użytkownikowi, że program oczekuje na wprowadzenie danych.

```
>>> age = input ("Enter your age: ")  
>>> print "When I was", age, "strange thing happened!"
```



Wyrażenia

Wyrażenia numeryczne zawierają operatory arytmetyczne, ,

+, -, *, /, //, **, %

odpowiednio dodawania, odejmowania, mnożenia, dzielenia, dzielenia całkowitego, podnoszenia do potęgi i operacji modulo (reszta z dzielenia liczb całkowitych). Wyrażenia te są wykonywane w następującej kolejności:

- 1) potęgowanie,
- 2) mnożenie, dzielenie i modulo (od lewej do prawej),
- 3) dodawanie i odejmowanie (od lewej do prawej).

Do zmiany kolejności obliczania wartości wyrażeń są używane nawiasy.

- Wynik dzielenia całkowitego jest zaokrąglony do najbliższej mniej liczby całkowitej

>>> 7//2

>>> -1//3

- Dzielenie całkowite liczb typu **float** daje wynik o wartości całkowitej, ale typu float.

- Przed użyciem w wyrażeniu, zmiennym muszą być przypisane wartości.



Komentarze

Komentarze rozpoczynają się znakiem „hash” #. Ich tekst jest ignorowany przez interpreter.

Tekst objaśniający treść programu innym (lub programiście)

Komentarze mogą być umieszczone w dowolnym miejscu programu w języku Python. Objaśniają „zrozumiały” dla komputera kod ludziom.

Interpreter języka Python tłumaczy program na postać, która może być wykonana przez układy komputera. Po uruchomieniu konsoli Pythona lub programu PyLab, interpreter wyświetla znak zachęty (np. >>>). Oznacza to, że interpreter jest gotowy do tłumaczenia kodu, który zostanie wpisany po tym znaku

```
>>> print „Witaj Ziomal!”
```



Ćwiczenia

3.1. Podaj trzy nazwy identyfikatorów, które są legalne i trzy nielegalne, w sensie reguł języka Python.

3.2. Znajdź zmienne w poniższym programie. Określ typ danych, które reprezentują i wartości, do których się odnoszą. Zidentyfikuj wyrażenia i instrukcje przypisania.

```
1  # tmp.py
2
3  from math import pi
4  r = 12
5  area = pi * r ** 2
6  print "The area of a circle with radius", r, "is", area
7
```

3.3. Wyjaśnij efekt usunięcia ostatniej spacji z instrukcji input().

```
>>> k = input("Enter k: ")
```




Ćwiczenia

3.4. Określ wartość każdego z wyrażeń zapisanych w języku Python:

(a) $1 + 2 * 3 - 4 * 5 + 6$

(b) $1 + 2 ** 3 * 4 - 5$

(c) $1 / 2 - 3 / 4$

(d) $1 // 2 - 3 // 4 + 5 // 6$

(e) $1 + 4 - 2 / 2$

(f) $(1 + 4 - 2) / 2$

3.5. Znajdź wartości wyjściowe zmiennych w poniższych fragmentach kodu

(a) `x = 10`

`y = 15`

`print x, y`

`y = x`

`print x, y`

`x = 5`

`print x, y`

(b) `x = 10`

`y = 15`

`print x, y`

`y = x`

`x = y`

`print x, y`

`x = 5`

`print x, y`

(c) `x = 10`

`y = 15`

`z = 20`

`x = z`

`z = y`

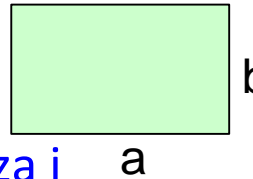
`y = x`

`print x, y, z`



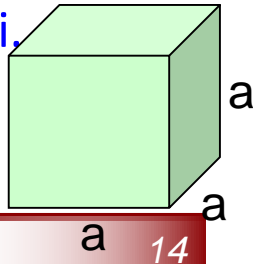
Ćwiczenia

3.6. Napisz program **average.py**, który oblicza i wyświetla średnią trzech liczb typu integer. Wyświetl wyniki w postaci liczb zmiennoprzecinkowych o dwóch cyfrach po separatorze części ułamkowej. Zmień liczbę wyświetlonych cyfr po kropce. Wskazówka: Użyj instrukcji **print** podobnej do zastosowanej w wierszu 10 programu **BMI.py**.



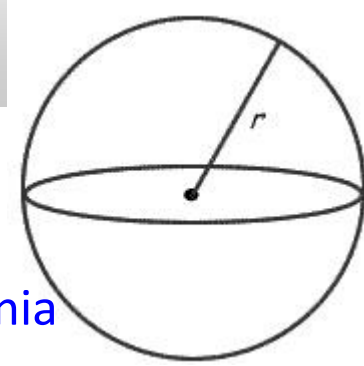
3.7. Zmodyfikuj skrypt **average.py** by napisać program **rect.py**, który oblicza i wyświetla wartości pola powierzchni i długości obwodu prostokąta o podanej szerokości i wysokości. Uruchom program dla kilku wartości wejściowych, aby go przetestować.

3.8. Zmodyfikuj skrypt **average.py** by otrzymać program **cube.py**, który oblicza i drukuje objętość i pole powierzchni sześcianu o podanej długości krawędzi. Uruchom program kilka razy dla różnych długości krawędzi by oszacować długość boku, dla której pole powierzchni jest (liczbowo) równe objętości.





Exercises



3.9. Zmodyfikuj skrypt **average.py** aby otrzymać program **sphere.py**, który oblicza i wyświetla objętość i pole powierzchni kuli o podanym promieniu. Uruchom program kilka razy dla różnych długości promienia by oszacować promień, dla którego pole powierzchni jest (liczbowo) równe objętości bryły.

3.10. Napisz skrypt, który wyświetla „prostokąt” jak niżej.

```
*****
```

```
*           *
```

```
*****
```

3.11. Posiadanie „kalkulatora napiwków” w telefonie komórkowym byłoby przydatne. Załóżmy, że w telefonie zainstalowano interpreter języka Python. Napisz program, by zapytać o cenę posiłku i procent napiwku. Wyświetl wartość napiwku i całkowitą sumę.





Podsumowanie

- 1) Zmienne odnoszą się do ...
- 2) ... poprzez instrukcje przypisania, które czytamy od prawej do lewej.
- 3) Jak dotąd, dane mogą być typu **string**, **integer** lub **float**. Typ danych przypisany zmiennej decyduje o tym, jakie operacje można na niej wykonać.
- 4) **input()** jest wyrażeniem wprowadzania danych do uruchomionego programu (w trybie konsoli języka Python).
- 5) Operatory arytmetyczne są używane w wyrażeniach numerycznych. Są wykonywane w określonej kolejności. Kolejność tę można zmienić za pomocą nawiasów.
- 6) Symbol "=" oznacza przypisanie, a nie matematyczny znak równości.
- 7) Komentarze są ignorowane przez interpreter języka Python. Skrypt programu powinien zawierać wiele komentarzy. (Dlaczego?)



Literatura

Brian Heinold, Introduction to Programming Using Python, Mount St. Mary's University, 2012 (<http://faculty.msmary.edu/heinold/python.html>).

Brad Dayley, Python Phrasebook: Essential Code and Commands, SAMS Publishing, 2007 (dostępne też tłumaczenie: B. Dayley, Python. Rozmówki, Helion, 2007).

Mark J. Johnson, A Concise Introduction to Programming in Python, CRC Press, 2012.