CYFROWA SZKOŁA
Wielkopolsk@ 2020

Zmienne i typy danych

Wprowadzenie do języka Python (II)

Pierwszą rzeczą, którą należy wiedzieć, jest to, że wszystkie pliki Pythona kończą się rozszerzeniem .py. Każdy język programowania musi mieć zdolność do przyjmowania, przechowywania i nazywania zmiennych

Musisz mieć możliwość odczytania danych z klawiatury lub z innych części programu i przypisania nazwy do tych danych. Te dane są albo pojedynczą wartością, albo wieloma wartościami, które są oznaczone nazwę. Dane, którym przypisano nazwę i które zawierają dane, nazywane są zmienną.

Python ma wiele różnych sposobów przechowywania wielu rodzajów danych, które zostaną omówione później.

Przypisywanie nazw do danych jest proste. Jeśli chcę przechowywać swój wiek w Pythonie, wpiszę my_age = 43. Gdybym chciał zapisać moje imię, wpisałbym my_name = „Jacek”.

python

CYFROWA SZKOŁA
Wielkopolsk@ 2020

Zasady nazewnictwa zmiennych

Twoje zmienne mogą zaczynać się od litery lub _ (podkreślenie)

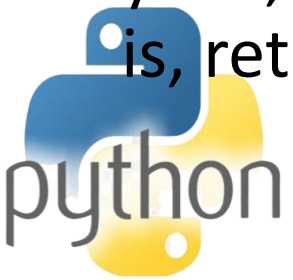
Po pierwszej literze możesz użyć cyfr, takich jak num_1

Nie możesz umieszczać spacji w nazwach zmiennych moj_wiek jest ok, ale moj wiek nie.

Dobrą praktyką jest oddzielanie słów podkreśleniem (moj_wiek vs. mojwiek)

Słowa kluczowe, których nie możesz nam dla nazw zmiennych

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



Spróbuj uruchomić ten kod, w którym przypisujesz swoje imię do zmiennej, a następnie wyświetlasz komunikat.

KOD

```
moje_imie = "Jacek"
print("Cześć", moje_imie)
```

„Hello” to ciąg znaków. Funkcja print() wypisuje na ekranie wartości pomiędzy cudzysłowami w nawiasach. Jeśli masz wiele wartości, oddziel je przecinkami.

Dane są przechowywane zasadniczo w komórkach w pamięci komputera. Rozmiar przypisanego obszaru pamięci jest określany jako typ danych. Jeśli chcesz przechowywać wartości z miejscami dziesiętnymi, przechowujesz te dane w typie danych zmiennoprzecinkowych. Jeśli chcesz przechowywać ciąg znaków, cyfr itp., przechowuj w ciągu tekstowym danych.

Strings

String to typ danych, który zaczyna się i kończy znakiem „ ” lub ” i zawiera litery, cyfry i inne elementy. Jeśli stwierdzisz, że chcesz użyć podwójnego cudzysłowu w ciągu, wykonaj to z takim odwrotnym ukośnikiem.

KOD

```
print("\"Nigdy tak naprawdę nie dorastamy, uczymy się tylko
zachowywać się publicznie\" - Bryan White")
```

CYFROWA SZKOŁA
Wielkopolsk@ 2020

Znaki ucieczki

\” to jedna z wielu Sekwencji Ucieczki. Oto inne popularne sekwencje ucieczki:

Nowa linia: \n

Ukośnik wsteczny : \\

Pojedynczy cytat : \”

Backspace : \b

Tabulator : \t

Numeryczny Typ Danych

W Pythonie istnieją 3 główne typy liczb. Liczby całkowite, zmiennoprzecinkowe i liczby zespolone. Liczby zespolone zostaną omówione przy innej okazji.

Liczby całkowite to wartości, które nie mają wartości dziesiętnych. 3, 8, 100000 to liczby całkowite. Liczby stałoprzecinkowe (float) zawierają wartości dziesiętne. Na przykład Pi to liczba typu float.

CYFROWA SZKOŁA
Wielkopolsk@ 2020

Nie ma maksymalnej wartości liczby całkowitej, o ile masz wystarczającą ilość pamięci. Możesz jednak uzyskaj dzięki temu praktyczny maksymalny rozmiar. Pamiętaj, że będziesz musiał zaimportować moduł sys aby ten kod działał. Moduł dostarcza wstępnie napisany kod (w zakresie procedur związanych z obsługą systemu operacyjnego), którego możesz użyć w swoim programie.

KOD

```
import sys
print(sys.maxsize)
```

Możesz uzyskać maksymalny rozmiar typu danych float

KOD

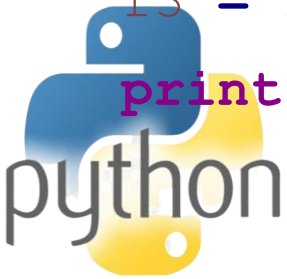
```
import sys

print(sys.float_info.max)
```

Należy jednak pamiętać, że podczas korzystania z typu danych float mogą wystąpić błędy. Dotyczy to wszystkich języków programowania.

Kiedy tworzysz zmienną, rezerwowana jest określoną ilość miejsca. Jeśli utworzysz wartość większa niż pozwala na to miejsce, mogą wkradać się błędy. Na przykład

KOD

```
f1 = 1, 11111111111111111
f2 = 1, 11111111111111111
f3 = f1 + f2
print(f3)
```



CYFROWA SZKOŁA
Wielkopolsk@ 2020

Jak widać, float mają dokładność do 15 cyfr. Później zostaną przedstawione typy danych z większą precyzją.

Oto przykład liczby zespolonej. Zostanie omówiona bardziej szczegółowo później.

KOD

```
Cn1 = 5 + 6j
```

Typ logiczny (Booleans)

Typ danych logicznych może mieć wartość True lub False.

KOD

```
może_głosować = True
```

CYFROWA SZKOŁA
Wielkopolsk@ 2020

Dynamiczne typowanie

Python jest typowany dynamicznie. Oznacza to, że typ danych zmiennych jest określony przez wartość, którą mu przypiszesz. Różni się to od innych języków. Wartością zmiennej może być również zmienione, nawet jeśli czasami może to nie mieć sensu. Na przykład

KOD

```
moj_wiek = 43

moj_wiek = "Pies"
```

Jeśli nie upewnisz się, że używasz prawidłowych wartości, może wystąpić wiele błędów. Czasami zaistnieje potrzeba konwersji z jednego typu na inny.

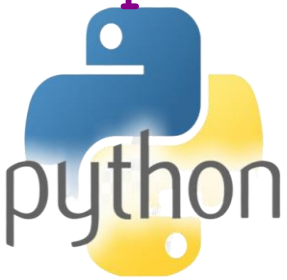
CYFROWA SZKOŁA
Wielkopolsk@ 2020

Rzutowanie (Casting)

Rzutowanie umożliwia konwersję z jednego typu na inny. Oto, jak można rzutować na różne typy danych. Użyję funkcji type(), aby wyświetlić nowy typ danych dla każdej zmiennej.

KOD

```
print("Cast ", type(int(5.4)))   # float to int
print("Cast 2 ", type(str(5.4)))   # float to string
print("Cast 3 ", type(chr(97)))   # znak Unicode do string
print("Cast 4 ", type(ord('a')))   # znak do unicode
print("Cast 5 ", type(float(2)))   # integer do float
```

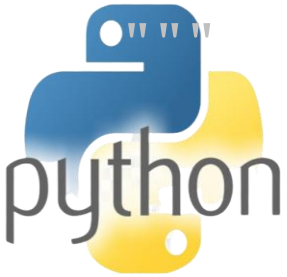


Komentowanie

Być może zauważyłeś symbol # użyty w powyższym kodzie. # jest używane, gdy chcesz skomentować, co robi twój kod. Wszystko, co wpisujesz po #, jest ignorowane. Bardzo ważne jest, aby komentować swój kod, ponieważ to, co dzisiaj rozumiesz o swoim kodzie, możesz zapomnieć za 3 miesiące od jego napisania. Możesz także tworzyć komentarze wielowierszowe, takie jak ten

KOD

```
"""

Jestem komentarzem wielowierszowym

"""
```



CYFROWA SZKOŁA
Wielkopolsk@ 2020

Typowe błędy związane ze zmiennymi

W nazwach zmiennych rozróżniana jest wielkość liter. Na przykład wiek to nie to samo co Wiek.

KOD

```
wiek = 2
Wiek = 3
```

Upewnij się, że przesyłasz dane do prawidłowego typu danych podczas pracy ze zmiennymi. Również upewnij się, że otaczasz obliczenia nawiasami, gdy generują pojedynczą wartość.

KOD

```
liczba_1 = "1"
liczba_2 = "2"
print("1 + 2 =", (int(liczba_1) + int(liczba_2)))
```