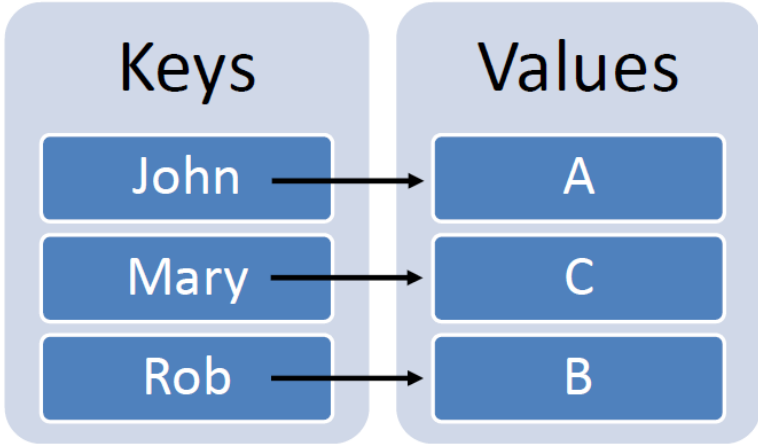
Krotki i słowniki

Podstawy programowania w języku Python



Typy sekwencyjne

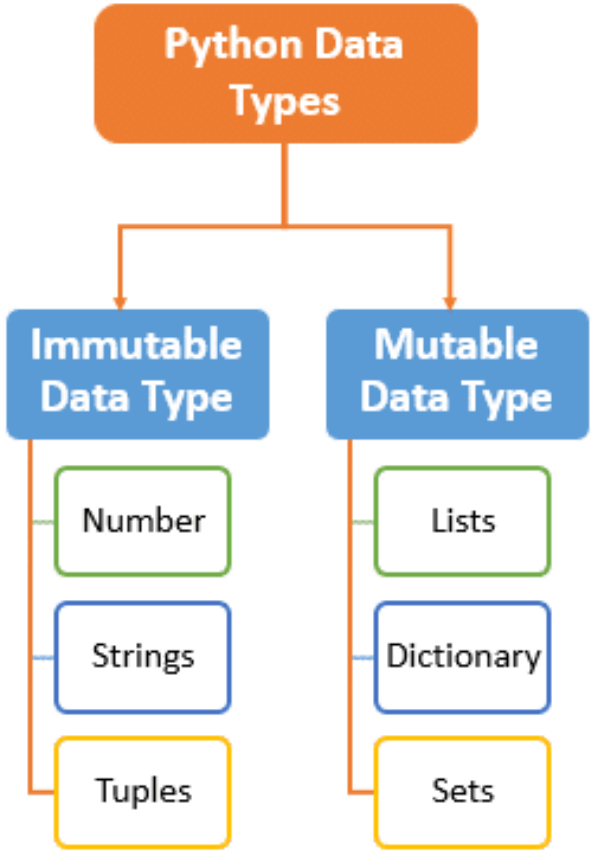
```
1    numbers = [1, 2, 3]
2    for i in numbers:
3        print(i)
```

- typ danych języka Python
- może przechowywać więcej niż jedną wartość
- wartości te mogą być przeglądane sekwencyjnie
- do skanowania sekwencji używamy pętli for

Lista jest klasycznym przykładem sekwencji języka Python, ale nie jedynym.

Mutowalność typów

- jest właściwością każdego typu danych języka Python
- zdolność danego typu do bezproblemowej zmiany wartości podczas wykonania programu
- wyróżniamy typy danych **mutowalne** i **niemutowalne**



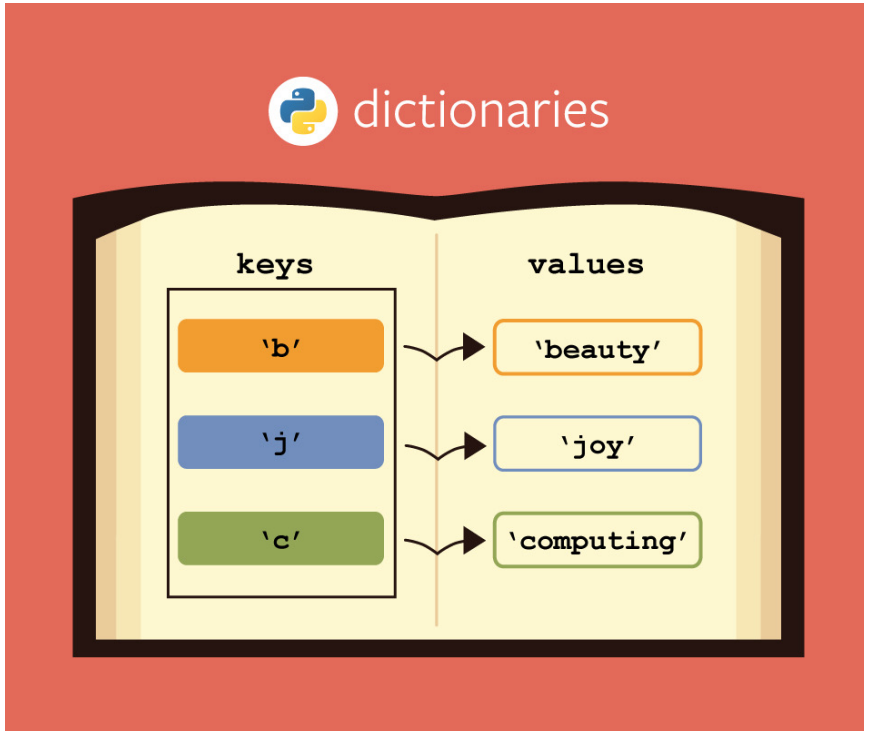
Krotki (tuples)

```
1    numbers = (1, 2, 3)
2
3    for i in numbers:
4        print(i)
```

- pod wieloma względami są podobne do list (typ sekwencyjny), choć między wymienionymi typami istnieją dwie ważne różnice
- krotka jest niezmienna (niemutowalna), dlatego też po utworzeniu krotki nie można zmienić jej zawartości, a nawet wielkości, o ile wcześniej nie zostanie utworzona kopia tej krotki
- krotkę zapisujemy za pomocą nawiasu okrągłego ( ) , a nie kwadratowego [ ]

Słownik (dictionary)

- kolejna struktura danych języka Python
- służy do przechowywania danych w parach klucz:wartość
- nie może mieć dwóch pozycji z tym samym kluczem
- jest typem zmiennym (mutowalnym)
- jest definiowany za pomocą nawiasów klamrowych



Tworzenie słowników

- stosujemy nawiasy klamrowe {}
- klucze od wartości rozdzielamy dwukropkiem :

```
1    words = {"kot": "cat", "pies": "dog", "ptak": "bird"}
```

- możemy także utworzyć pusty słownik i dodać do niego wartości

```
1    words = {}
2
3    words["kot"] = "cat"
4    words["pies"] = "dog"
5    words["ptak"] = "bird"
```

Pobieranie elementów

- użycie klucza w nawiasach kwadratowych
- użycie klucza jako argumentu metody **get()**

```
1  words = {"kot": "cat", "pies": "dog", "ptak": "bird"}
2
3  print(words["kot"])
4  print(words.get("pies"))
5
6  print(words["krowa"])  # błąd!
7  print(words.get("krowa"))  # wynik: None
```

Modyfikowanie elementów słownika

- zmiana wartości elementu

```
1  words = {"kot": "cat", "pies": "dog", "ptak": "bird"}
2
3  words["kot"] = "kitten"
```

- usuwanie elementu

```
1  del words["kot"]
```

- aktualizowanie elementu lub elementów metodą update()

```
1  words.update({"kot": "cat", "chomik": "hamster"})
```

Funkcje i metody

- funkcja **len()** pozwala ustalić liczbę elementów słownika
- metoda **items()** zwraca wszystkie elementy słownika w parach (krotkach) klucz, wartość
- metoda **keys()** zwraca wszystkie klucze słownika
- metoda **values()** zwraca wszystkie wartości słownika

```
1  words = {"kot": "cat", "pies": "dog", "ptak": "bird"}
2
3  print(len(words)) #wynik: 3
4  print(words.keys()) #wynik: dict_keys(['kot', 'pies', 'ptak'])
```

Słownik i pętla for

- słownik nie jest typem sekwencyjnym, ale…
- możemy iterować po liście kluczy słownika

```
1  words = {"kot": "cat", "pies": "dog", "ptak": "bird"}
2
3  for key in words.keys():
4      print(key, "->", words[key])
```

- możemy iterować po liście elementów słownika

```
1  words = {"kot": "cat", "pies": "dog", "ptak": "bird"}
2
3  for key, value in words.items():
4      print(key, "->", value)
```

Pytanie

Krotka to w Pythonie typ danych:

a) uporządkowany, mutowalny, sekwencyjny
b) nieuporządkowany, niemutowalny, sekwencyjny
c) uporządkowany, niemutowalny, sekwencyjny
d) uporządkowany, niemutowalny, niesekwencyjny
e) nieuprządkowany, niemutowalny, niesekwencyjny

Odpowiedź: c)

Krotka jest typem uporządkowanym, bo zachowuje kolejność wstawianych elementów. Jest typem niemutowalnym, bo nie można aktualizować jej elementów bez tworzenia kopii krotki. Jest typem sekwencyjnym, bo można używać pętli for, aby iterować po krotce podobnie jak po liście.

Pytanie

Co wyświetli się na ekranie po wykonaniu poniższego skryptu?

```
a)  None
b)  3.0, 5.0
c)  nic się nie wyświelti
d)  [2.0, 2.0, 2.0]
e)  wystąpi błąd KeyError
```

```
1    students = {
2        "Adam": [3.0, 5.0],
3        "Karolina": [5.0],
4        "Olaf": [2.0, 2.0, 2.0]
5    }
6
7    print(students["Matylda"])
```

Odpowiedź: e)

Przy próbie odwołania się do elementu słownik za pomocą nieistniejącego klucza oraz konstrukcji z nawiasami kwadratowymi otrzymamy błąd.

Pytanie

Co się wydarzy jeżeli spróbujemy wstawić do słownika wartość z kluczem, który istnieje już w słowniku?

a) wystąpi błąd, bo słownik nie może przechowywać duplikatów
b) wartość w słowniku dla tego klucza zostanie zaktualizowana
c) nowa para zostanie dopisana na końcu słownika
d) nowa wartość zostanie zapisana w słowniku z losowym kluczem

Odpowiedź: b)

Słownik może posiadać tylko unikalne klucze, dlatego próba wstawienia wartości z istniejącym w słowniku kluczem, spowoduję aktualizację tej pozycji nową wartością.