Listy część 2

Podstawy programowania w języku Python

Zamiana wartości dwóch zmiennych

- wygodny sposób na zamianę wartości dwóch lub więcej zmiennych
- bez dodatkowych zmiennych pomocniczych
- w jednej linii

zmienna_1, zmienna2 = zmienna_2, zmienna_1

```
1  var_1 = 1
2  var_2 = 2
3
4  var_1, var_2 = var_2, var_1
5
6  print(var_1, var_2) #wynik: 2 1
```

Odwrócenie kolejności elementów listy

- do odwrócenia kolejność elementów listy używamy metody **reverse()**
- metoda nie pobiera żadnych argumentów
- metoda nie zwraca żadnych wartości

```
1    numbers = [4, 2, 5, 1, 7]
2
3    numbers.reverse()
4    print(numbers) #wynik: [7, 1, 5, 2, 4]
```

Sortowanie bąbelkowe

- jeden z prostszych lecz mało wydajnych algorytmów sortowania
- porównywane i w razie potrzeby zamieniane sąsiadujące elementy

$$[\underbrace{4,2}_{4>2},5,1,7] \rightarrow [2,\underbrace{4,5}_{4<5},1,7] \rightarrow [2,4,\underbrace{5,1}_{5>1},7] \rightarrow [2,4,1,\underbrace{5,7}_{5<7}]$$

$$[\underbrace{2,4}_{2<4},1,5,7] \rightarrow [2,\underbrace{4,1}_{4>1},5,7] \rightarrow [2,1,\underbrace{4,5}_{4<5},7]$$

$$[\underbrace{2,1}_{2>1},4,5,7] \rightarrow [1,\underbrace{2,4}_{2<4},5,7]$$

$$[\underbrace{1,2}_{1<2},4,5,7]$$

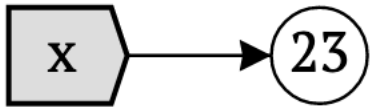
Sortowanie listy

- metoda **sort()** służy do sortowania elementów listy
- posortowane elementy domyślnie uszeregowane są rosnąco
- parametr **reverse** pozwala sortować elementy w porządku malejącym

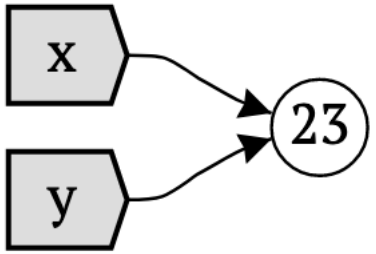
```
1  numbers = [4, 2, 5, 1, 7]
2
3  numbers.sort()
4  print(numbers) #wynik: [1, 2, 4, 5, 7]
5
6  numbers.sort(reverse=True)
7  print(numbers) #wynik: [7, 5, 4, 2, 1]
```

Operacje na typach niezmiennych

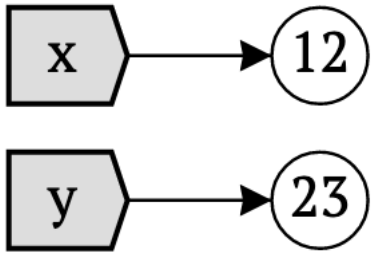
```
1    x = 23
```




```
1    x = 23
2    y = x
```

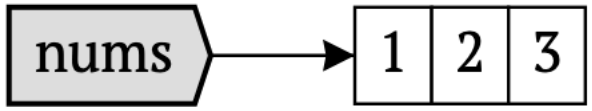



```
1    x = 23
2    y = x
3    x = 12
```

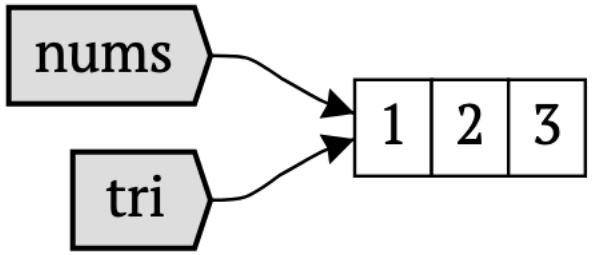


Operacje na typach zmiennych

```
1    nums = [1, 2, 3]
```

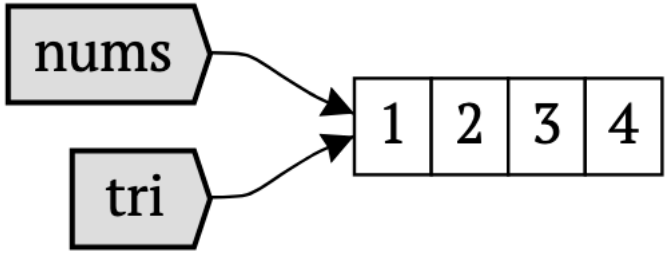



```
1    nums = [1, 2, 3]
2    tri = nums
```




```
nums.append(4)
```

```
1    nums = [1, 2, 3]
2    tri = nums
3    nums.append(4)
```



Wycinanie (slicing)

`list[start:end]`

- wycinek to element składni języka Python, który umożliwia **tworzenie zupełnie nowej kopii listy lub jej części**
- pobiera elementy z listy źródłowej z indeksami od **start** do **end - 1**
- można stosować indeksy ujemne

```
1    numbers = [7, 4, 9, 2]
2    slice = numbers[1:3] #kopia fragmentu listy
3    print(slice) #[4, 9]
```

Operatory członkostwa in, not in

- służą do sprawdzania, czy wartość lub zmienna została znaleziona w sekwencji (łańcuch znaków, lista, krotka, zbiór lub słownik)

```
elem in list
elem not in list
```

```
1  numbers = [10, 8, 6, 4, 2]
2  print(5 in numbers)  #czy na liście znajduje się 5, wynik: False
3  print(7 not in numbers)  #upewniamy się, że 7 nie ma na liście, wynik: True
```

Wyrażenie listowe

- prosty i elegancki sposób na tworzenie list
- wykonywanie złożonych operacji przy użyciu jednej linijki kodu
- tworzenie list w oparciu o jakąś sekwencję lub inną listę, którą można iterować

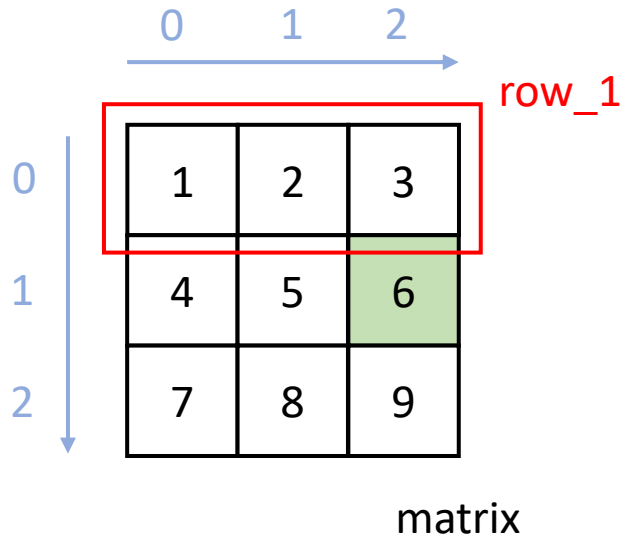
```
[expression for item in list]
```

```
1    squares = [x*x for x in range(1, 6)]
2    print(squares) #wynik: [1, 4, 9, 16, 25]
```

Listy wielowymiarowe

- listy mogą zawierać inne listy
- nie ma ograniczenia poziomu zagłębiania

```
1  row_1 = [1, 2, 3]  #lista 1
2  row_2 = [4, 5, 6]  #lista 2
3  row_3 = [7, 8, 9]  #lista 3
4
5  # lista list (tablica dwuwymiarowa)
6  matrix = [row_1, row_2, row_3]
7
8  print(matrix[1][2])  #wynik: 6
```



Pytanie

Która instrukcja posortuje elementy listy **numbers** malejąco?

a) `numbers.reverse()`
b) `sort numbers`
c) `numbers.sort(reverse=False)`
d) `numbers.sort(reverse=True)`
e) `numbers.sort()`

Odpowiedź: d)

Do posortowania listy możemy zastosować metodę **sort()**. Parametr **reverse** ustawiony na True pozwala nam uzyskać malejącą kolejność elementów.

Pytanie

Co wyświetli się na ekranie po wykonaniu poniższego skryptu?

a) [”jeden”, ”dwa”, 3]
b) [1, 2, 3]
c) [”jeden”, ”dwa”]
d) [1, 2]
e) wystąpi błąd NameError

```
1  list_1 = [1, 2]
2  list_2 = ["jeden", "dwa"]
3
4  list_1 = list_2
5  list_1.append(3)
6
7  print(list_2)
```

Odpowiedź: a)
Od linii 4 nazwa list_1 i list_2 wskazują na tę samą listę [”jeden”, ”dwa”].

Pytanie

Jak usunąć 3 ostatnie elementy z listy numbers = [1, 2, 3, 4, 5]?

```
a) del numbers[3]
b) del numbers
c) del numbers[-3:]
d) del numbers[1:3]
e) del numbers[:]
```

Odpowiedź: c)