Pętle

Pętle służą do wykonywania tego samego fragmentu kodu wielokrotnie. W Pythonie, pętle wykonywane są na obiektach wieloelementowych, albo iteratorach.

Pętla `for`

Pętla `for` wykonuje się na zestawie elementów. Dosłownie można tę instrukcję przeczytać jako "Dla iksów będących wartościami listy, wykonaj instrukcję:"

```
for x in [1, 3, 4, 2]:
    print(f'Value is: {x}')
```

```
for x in ['Max', 3, 'Peck', 2.8, [1, 'José', 'Jiménez']]:
    print(f'Value is: {x}')
```

```
for x in range(0, 30):
    print(f'Value is: {x}')
```

```
for x in range(0, 30, 5):
    print(f'Value is: {x}')
```

```
for key, value in [(0, 0), (1, 1), (1, 2)]:
    print(f'{key} -> {value}')
```

```
slownik = {'x': 1, 'y': 2}

for key in slownik:
    print(slownik.get(key))

for key, value in slownik.items():
    print(key, value)
```

Pętla `while`

Pętla while wykonuje się dopóki argument jest prawdą.

```
x = 0

while x <= 10:
    print(f'Value is: {x}')
    x = x + 1
```

```python
while True:
    number = input('Type number: ')

    if number:
        break
```

Słowa kluczowe w pętlach

`break` - powoduje przerwanie pętli.

`continue` - powoduje przerwanie aktualnie wykonywanej iteracji.

Inline `for`

Pętla `for` może być także napisana jako jednoliniowy generator. wtedy wyrażenie my_function(x) jest wykonywane dla każdego x z podanego zakresu, a wynik jest zapisywany do nowej listy.

```python
def my_function(i):
    return f'This is my function of {i}'

my_list = [my_function(x) for x in range(0, 100)]
```

Do takiego iteratora można także dodać instrukcję warunkową.

```python
my_list = [my_function(x) for x in range(0, 100) if x > 50]
```