# Python #6: Logika Pythona – instrukcje warunkowe

Instrukcje warunkowe inaczej sterujące to jedna z podstaw w każdym języku programowania. Działają dokładnie jak nazwa wskazuje – instrukcje lub blok instrukcji wykonuje się tylko, gdy określony warunek (lub zestaw warunków) jest spełniony.

## Sprawdzanie warunków

Zacznijmy od czegoś dosć oczywistego dla nas, ale czy również oczywistego dla komputera – porównania. Porównanie w języku mówionym wygląda tak: „Czy x jest większe od y?”. Chcielibyśmy otrzymać wynik tak lub nie. W matematyce zapiszemy x > y.

## Porównania prawda czy fałsz?

Zobaczmy jak z porównaniami radzi sobie Python:

```
>>> 0 < 1
True
>>> 0 > 1
False
>>> 0 <= 1
True
>>> 0 >= 1
False
>>> 1 == 0
False
>>> 1 == 1
True
>>> 1 != 0
True
>>> 1 != 1
False
```

Znak większość, mniejszości są zapewne oczywiste.

Kolejne dwa to znak mniejsze równe i większe równe, pewnie też wyglądały znajomo.

By porównać czy dwie wartości są takie same używamy ==.

Za pomocą != sprawdzimy wartość twierdzenie „nie równa się”. 1 nie równa się 0 czyli prawda 🙂

Wynik porównania zawsze zwraca nam prawdę – True lub fałsz – False.

Porównania można łączyć w bardziej złożone warunki, używając **operatorów logicznych**:

- `and` – oznaczającego, że oba warunki muszą być prawdziwe
- `or` – oznaczającego, że jednen z dwóch warunków musi być spełniony

```
>>> x = 10
>>> x < 10
False
>>> x > 1 and x < 13
True
>>> x < 1 and x < 13
False
>>> x > 1 or x < 13
True
>>> x != 10 or x > 13
False
>>> x > 1 and x < 13 or x != 10
True
```

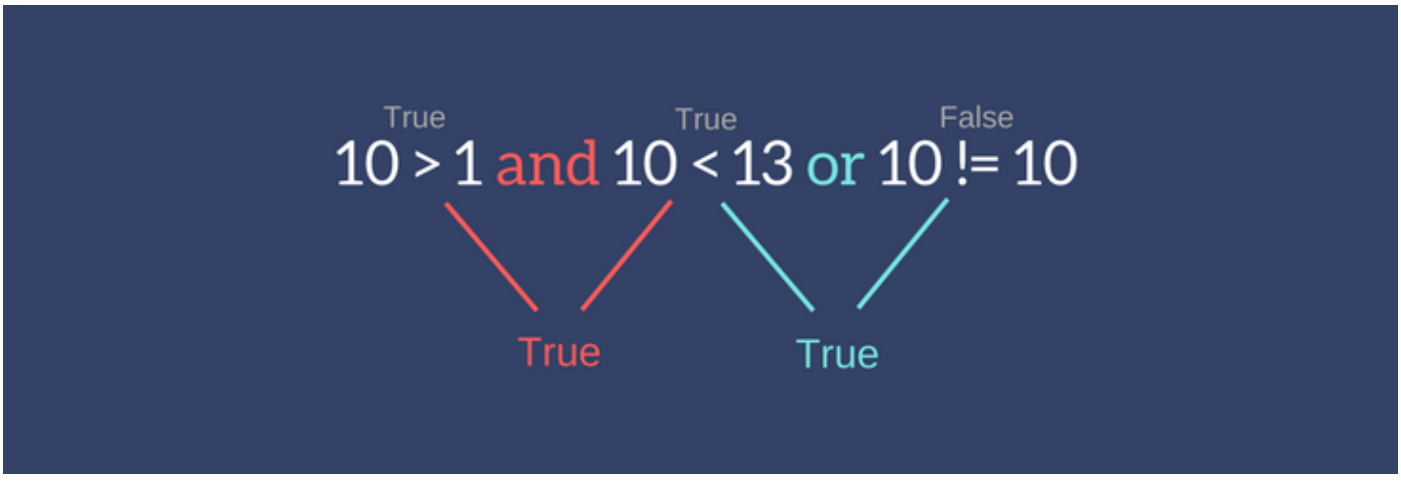
Na początku ustaliliśmy, że x wynosi 10.

W związku z tym ostatni przykład przeczytalibyśmy mniej więcej tak:

*Czy x jest większe równe 1* `ORAZ` *x jest mniejsze niż 13* `LUB` *x nie jest równe 10?*

- Czy 10 jest większe niż 1 – True
- Czy 10 jest mniejsze niż 13 – True
- Czy 10 nie równia się 10 – False

Ponieważ między dwoma pierwszymi porównaniami jest słowo `and` to oba warunki muszą być spełnione – prawdziwe. Jednak między drugim warunkiem, a trzecim jest słowo `or`, w związku z czym wystarczy, że tylko jedna strona warunku jest spełniona. Razem całość jest prawdziwa.




Możemy też dodać nawiasy sugerujące kolejność łączenia warunków:
np

`x > 1 and (x < 13 or x != 10)` True
`(x > 1 and x < 13) and x != 10` False

To nie wszystkie porównania zwracające prawdę lub fałsz.

```
>>> x = 10
>>> x is 10
True
>>> x is not 13
True
>>> x is '10'
False
>>> x in [ 1, 12, 10 ]
True
>>> x in [ 1, 12, 'txt' ]
False
>>> x not in [ 1, 12, 'txt' ]
True
>>> 'str' is 'str'
True
>>> 'str' is 'string'
False
>>> 'str' in 'string'
True
```

Kolejne porównania:
– `is` – sprawdza czy dwie wartości są identyczne,
– `in` – sprawdza czy zmienna jest zawarta w innym obiekcie
– `not` – dodaje zaprzeczenie

W związku z powyższym może nasunąć się pytanie:

### Czy `is` jest tym samym co `==`?

Żeby się przekonać, czy oba porównania działają tak samo sprawdzimy:
`'aa' == 'aa'` True
`'aa' is 'aa'` True

`'aa' == 'a' * 2` True
`'aa' is 'a' * 2` True

```
x = "a"
'aa' == x * 2 True
'aa' is x * 2 False
```

Ha! Tu Cię mam! x przechowuje wartość stringa „a”, w związku z czym „a”*2 powinno nam dać nowy string „aa”.

Wszystko się, zgadza, dopóki nie dodamy słowa *nowy*. **Znak == sprawdza czy po obu stronach równania mamy takie same wartości, natomiast `is` sprawdza czy dwie strony równania są identyczne – dwie zmienne wskazują na ten sam obiekt.**

Chwila chwila, ale x = 10, `x is 10` zwróciło True.
Owszem, przypadek działa tylko dlatego, że Python cachuje małe obiekty typu integer, co jest szczególnym przypadkiem.

Dla większych liczb całkowitych to nie działa:
`1000 == 10**3` True
`1000 is 10**3` False

Jest jeszcze jedna rzecz, którą warto zapamiętać
– wartości liczbowe możemy również wyrazić jako wartości logiczne True/False.
Przyjmuje się, że 1 odpowiada True, a 0 – False.

Czas na szybkie podsumowanie:

W Pythonie możemy skorzystać z kilku różnych operatorów porównania:

- >
- <
- >=
- <=
- ==
- !=
- is
- in
- not

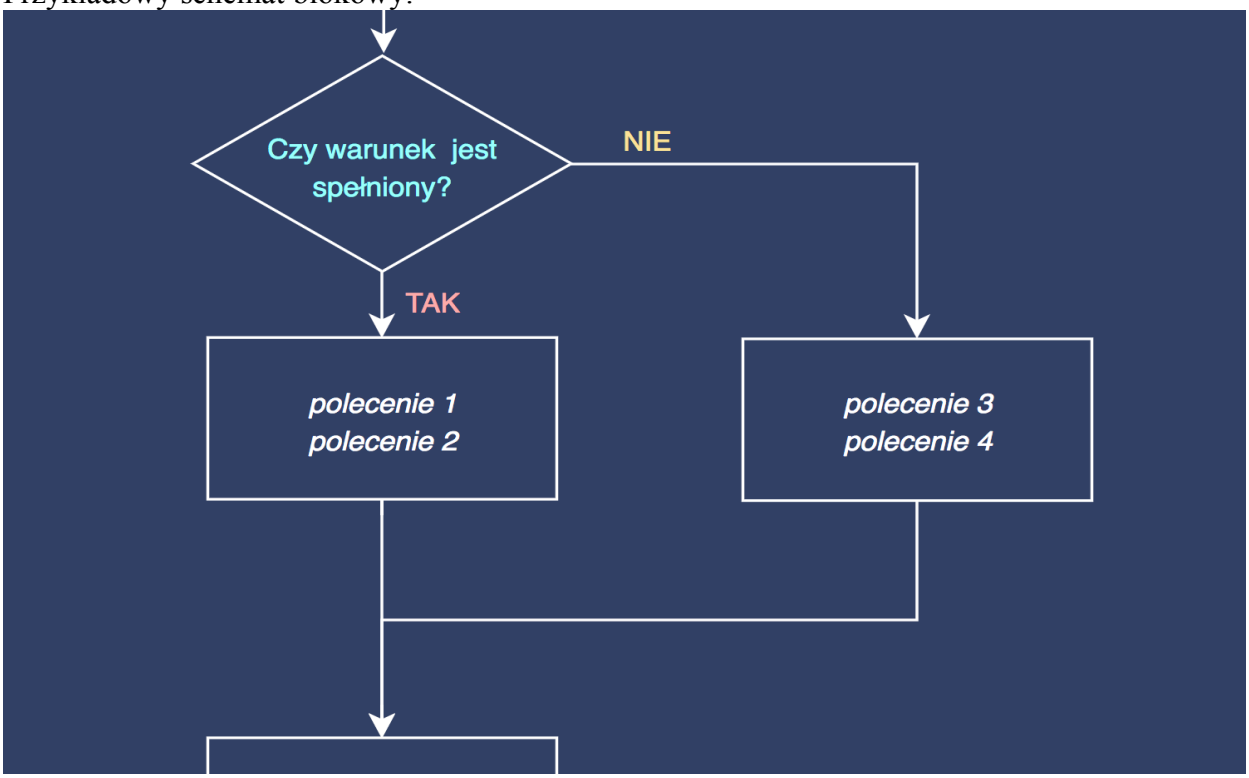
Do łączenia warunków użyjemy operatorów logicznych:

- and
- or

## Instrukcje `if`

Na początku wpisu wprowadziłam pojęcie instrukcji warunkowej. **Instrukcja sterująca służy do sprawdzenia czy zadane wyrażenie warunkowe jest prawdziwe.** W zależności od tego czy wyrażenie warunkowe jest prawdziwe czy nie, zależy czy zostanie wykonany fragment kodu.

Przykładowy schemat blokowy:




```
if ( warunek ):
   polecenie1
   polecenie2
else:
   polecenie3
   polecenie4
```

Instrukcja `if (warunek)`: dostarcza informacji do interpretera i znaczy jeśli/jeżeli.

**if (warunek)** – *jeżeli warunek jest prawdziwy wykonaj kod polecenia 1 i polecenia 2, w przeciwym wypadku wykonaj polecenie 3 i 4.*

Warto zwrócić uwagę na wcięcia. W Pythonie przyjmuje się, że wcięcie powinno składać się z 4 spacji. Dopuszcza się również użycia tabulacji jako wcięcia informacji o bloku kodu.

Stworzymy skrypt *adult.py*:

```
print("Ile masz lat?")
age = int( input() )

if ( age >= 18 ):
    print("Jesteś dorosłym człowiekiem")
else:
    print("Trochę Ci zostało do pełnoletności")
```

Powyższy skrypt zapyta użytkownika o wiek i zwróci informację czy jest już dorosły. Na podstawie dotychczasowych części kursu dodaj wyświetlanie informacji, ile lat zostało do pełnoletności 😉

## Instrukcje warunkowe: if…elif … else

Dotychczasowe kody działały wykonując jeden z dwóch bloków kodu. A co jeśli chcemy sprawdzić dodatkowy warunek?

Wówczas możemy ponownie wykorzystać słowo `if` lub przyda nam się słowo `elif` (else if).

Do poprzedniego kodu dodaj kod:

```
if (age > 100):
    print ("To naprawdę twój wiek?")
```

następnie zmień if na elif:

```
elif (age > 100):
    print ("To naprawdę twój wiek?")
```

Skrypt adult_if.py:

```
print("Ile masz lat?")
age = int( input() )

if ( age >= 18 ):
   print("Jesteś dorosłym człowiekiem")
if ( age > 100 ):
   print("To naprawdę twój wiek?")
else:
   print("Trochę Ci zostało do pełnoletności")
```

Skrypt adult_elif.py

```
print("Ile masz lat?")
age = int( input() )

if ( age >= 18 ):
   print("Jesteś dorosłym człowiekiem")
elif ( age > 100 ):
   print("To naprawdę twój wiek?")
else:
   print("Trochę Ci zostało do pełnoletności")
```

Możecie zauważyć, że kolejny `if` sprawdzi warunek każdorazowo, natomiast `elif` – kod wykona się, gdy warunek z `if` jest wyrażeniem fałszywym, a `elif (warunek)` będzie prawdziwy.

*Zadania nie są ułożone stopniem trudności. Jeśli jakiegoś nie możesz zrobić – przejdź do następnego, może za chwilę wpadniesz na rozwiązanie* 😉

**Zadanie 1** – pojawiło się powyżej
Skrypt zapyta użytkownika o wiek. Jeżeli użytkownik jest przed 18 wyświetli informację „Użytkownik niepełnoletni" oraz zwróci, ile lat zostało użytkownikowi do pełnoletności. Użytkownikom pełnoletnim wyświetli informację „Użytkownik pełnoletni". Sprawdź czy wiek użytkownika nie przekracza 100 lat i wyświetl komunikat „200 lat 🎵".

**Zadanie 2 – kalkulator BMI**
W poprzednich częściach pisaliśmy kalkulator BMI, teraz wzbogacimy go o interpretację wyniku. Program powinien sprawdzić czy BMI należy do jednego z 4 wyników: niedowaga/waga normalna/lekka nadwaga i nadwaga. Ponadto w przypadku nadwagi chcemy sprawdzić, czy mamy do czynienia z otyłością.

Poniżej tabela z klasyfikacją wyników:

| | |
|---|---|
| Niedowaga | < 18,5 |
| Waga normalna | 18,5 – 24 |
| Lekka nadwaga | 24 – 26,5 |
| Nadwaga | > 26,5 |
| Otyłość I stopnia | 30 – 35 |
| Otyłość II stopnia | 30 – 40 |
| Otyłość III stopnia | > 40 |

**Zadanie 3 – sortowanie liczb**
Trzy dowolne liczby zapisz do trzech zmiennych.
Znajdź największą liczbę.
Wyświetl liczby od największej do najmniejszej.

**Zadanie 4 – imiona**
Utwórz zbiór imion męskich i żeńskich. Poproś użytkownika o podanie imienia. Sprawdź, czy imię jest męskie czy żeńskie i wyświetl na ten temat informację. Jeżeli imienia nie ma na liście wyświetl komunikat „Nie znamy tego imienia." Następnie użytkownik poda informację czy imię jest męskie czy żeńskie. Dodaj imię do listy.

Podpowiedź: warto przypomnieć sobie zadanie 2 z lekcji 4

**Zadanie 5 – twierdzenie Pitagorasa**
Pamiętacie z matematyki twierdzenie Pitagorasa?
Trójkąt pitagorejski, to trójkąt prostokątny, którego boki są wyrażone liczbami naturalnymi a, b, c związanymi warunkiem: $\mathbf{a^2+b^2=c^2}$.

a) Poproś użytkownika o podanie długości boków A, B i C i sprawdź czy w ogóle możliwe jest utworzenie z nich trójkąta 🙂

b) Odpowiedz, czy trójkąt jest trójkątem pitagorejskim.

c) Szczególnym przypadkiem jest trójkąt egipski o stosunkach długości 3:4:5. Sprawdź czy trójkąt pitagorejski jest trójkątem egipskim.

d) Uwzględnij, że kolejność danych nie musi mieć znaczenia! Tzn. długość C użytkownik może podać jako pierwszą wartość. Przypomnij sobie zadanie 3 🙂

Przykładowe dane:

- nie-trójkąty: [6, 9, 20], [3, 6, 11], [31, 14, 17]
- trójkąty nie-pitagorejskie: [4, 4, 4], [31, 17, 16], [10, 5, 8]
- trójkąty pitagorejskie: [3, 4, 5], [6, 8, 10], [5, 12, 13], [9, 40, 41], [8, 15, 17]
- trójkąty egipskie: [9, 12, 15], [3, 4, 5], [15, 20, 25], [6, 8, 10]