**Comparisons**

Istnieją pewne standardowe operatory porównania, których będziemy używać w sposób bardzo zbliżony do tych używanych w równaniach matematycznych. Spójrzmy na nie:

>>> 1 < 2

True

>>> 0 > 2

False

>>> 2 == 1

False

>>> 2 != 1

True

>>> 3.0 >= 3.0

True

>>> 3.1 <= 3.0

False

Jeśli spróbujemy dokonać porównań typów, które się nie zgadzają, napotkamy błędy:

>>> 3.1 <= "this"

Traceback (most recent call last):

File "", line 1, in

TypeError: '<=' not supported between instances of 'float' and 'str'

>>> 3 <= 3.1

True

>>> 1.1 == "1.1"

False

>>> 1.1 == float("1.1")

True

Możemy porównać więcej niż tylko liczby. Oto jak wygląda porównanie ciągów:

>>> "this" == "this"

True

>>> "this" == "This"

False

>>> "b" > "a"

True

>>> "abc" < "b"

True

Zauważ, że łańcuch „b” jest uważany za większy niż łańcuchy „a” i „abc”. Znaki są porównywane pojedynczo alfabetycznie, aby określić, która jest większa. Ta koncepcja służy do sortowania ciągów alfabetycznie.

**The in Check**

Często otrzymujemy listy informacji, które musimy zapewnić, aby zawierały (lub nie zawierają) konkretny przedmiot. Aby dokonać tego sprawdzenia w Pythonie, użyjemy operacji in i inoperation.

>>> 2 in [1, 2, 3]

True

>>> 4 in [1, 2, 3]

False

>>> 2 not in [1, 2, 3]

False

>>> 4 not in [1, 2, 3]

True

**if/elif/else**

Mając do czynienia z porównaniami, możemy teraz przyjrzeć się, w jaki sposób możemy uruchamiać różne elementy logiki w oparciu o wartości, z którymi współpracujemy przy użyciu warunkowych. Słowa kluczowe warunkowych w Pythonie to if, elif i in. Warunkowe są pierwszą funkcją językową, której używamy, która wymaga od nas użycia białych znaków w celu oddzielenia naszych bloków kodu. Zawsze użyjemy wcięć 4 spacji. Podstawowy kształt instrukcji if to:

if CONDITION:

pass

Część CONDITION może być czegokolwiek, co ma wartość True lub False, a jeśli wartość nie jest jawnie logiczna, zostanie przekonwertowana w celu określenia sposobu przeprowadzenia przechodzenia obok warunku (zasadniczo za pomocą konstruktora bool).

>>> if True:

... print("Was True")

...

Was True

>>> if False:

... print("Was True")

...

>>>

Aby dodać alternatywną ścieżkę kodu, użyjemy słowa kluczowego else, po którym następuje dwukropek (:) i wcięcie kodu poniżej:

>>> if False:

... print("Was True")

... else:

... print("Was False")

...

Was False

Nawet jeśli chcemy sprawdzić wiele potencjalnych warunków, możemy użyć instrukcji elif CONDITION:. Oto bardziej solidny przykład:

>>> name = "Kevin"

>>> if len(name) >= 6:

... print("name is long")

... elif len(name) == 5:

... print("name is 5 characters")

... elif len(name) >= 4:

... print("name is 4 or more")

... else:

... print("name is short")

...

name is 5 characters

Zauważ, że wpadliśmy do pierwszego bloku instrukcji elif, a następnie drugi blok elif nigdy nie został wykonany, mimo że był prawdziwy. Możemy wykonać tylko jedną gałąź w instrukcji if.