Sushil Singh | 2023

Python Control Flow

Comparison Operators

Operator	Meaning
==	Equal to
!=	Not equal to
<	Less than
>	Greater Than
<=	Less than or Equal to
>=	Greater than or Equal to

A QUICK REVISION

Integer comparison

```
35 <= 45
>>> True
35 > 45
>>> False
45 == 45
>>> True
```

String comparison

```
'hello' != 'hello'
>>> False
```

Numerical 45 is not equal to String ‘45’

```
45 == '45'
>>> False
```

Float 45 and Integer 45 are same

```
45 == 45.0
>>> True
```

Boolean Operators

There are three Boolean operators: and, or, and not. Two True conditions with ‘and’ is True

```
(7 < 9) and (5 > 4)
>>> True
```

One or more False conditions with ‘and’ is False

```
(8 < 9) and (9 < 5)
>>> False
```

True and False condition with ‘or’ is True

```
(1 >= 2) or (2 == 2)
>>> True
```

Using multiple boolean conditions with comparison operators

```
(4 + 4 == 8) and not (5 - 2 == 5 ) and (4 * 4 == 14 + 2)
>>> True
```

if Statements

The if the expression is True, it will execute the following indented code.

```
name = 'Abid'
if name == 'Abid':
  print('Hi, Abid')
>>> Hi, Abid
```

if statement with else. The else statement will execute when if and elif expressions are False.

```
name = 'Abid'
if name != 'Abid':
  print('Hi, Matthew')
else:
  print('Hi, Abid')
>>> Hi, Abid
```

if statement with else. The else statement will execute when if and elif expressions are False.

```
name = 'Abid'
if name != 'Abid':
  print('Hi, Matthew')
else:
  print('Hi, Abid')
>>> Hi, Abid
```

Multiple statements with if, elif, and else.

```
name = 'Matthew'
if name == 'Abid':
  print('Hi Abid!')
elif name == 'Matthew':
  print('Hi Matthew!')
else:
  print('Unknown User')
>>> Hi Matthew!
```

Ternary Conditional Operator

It allows us to convert the if statement to one-line code.

```
profit = 100
if profit > 50:
  print('Bonus')
else:
  print('No Bonus')
>>> Bonus
```

Ternary operator equivalent

```
print('Bonus' if profit > 50 else 'No Bonus')
>>> Bonus
```

while Loop

The while statement will keep running as long as the statement is True.

The statement below becomes False after 10 iterations. It will print ‘Welcome to KDnuggets’ 10 times.

```
loop = 0
while loop < 10:
  print('Welcome to KDnuggets')
  loop = loop + 1
```

When the statement reaches a break statement, it will exit the while loop. It will print ‘Welcome to KDnuggets’ 10 times.

```
loop = 0
while True:
  print('Welcome to KDnuggets')
  loop+=1
  if loop == 10:
      break
```

for loop

The for loop iterates over a dictionary, list, tuple, set, or string The for loop will print individual items (colors) in the list.

```
colors= ['Red', 'Blue', 'Green']
for i in colors:
  print(i)
>>> Red
>>> Blue
>>> Green
```

for loop with range

```
for i in range(10):
  print('Welcome to KDnuggets')
```

for else statement

```
for i in [4, 5, 11, 9]:
  if i == 11:
    break
  else:
    print("List does not have number 11")
```