In [ ]:

```
Python Conditions and If statements

Python supports the usual logical conditions from mathematics:

Equals: a == b
Not Equals: a != b
Less than: a < b
Less than or equal to: a <= b
Greater than: a > b
Greater than or equal to: a >= b

An "if statement" is written by using the if keyword.
```

In [2]:

```
a = 33
b = 200

if (b > a):
    print("b is greater than a")
```

```
b is greater than a
```

In [1]:

```
a = 33
b = 200

if b > a:
  print("b is greater than a")
```

```
b is greater than a
```

In [3]:

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

```
a and b are equal
```

In [4]:

```
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a == b:
    print("a and b are equal")
else:
    print("a is greater than b")
```

```
a is greater than b
```

In [5]:

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
else:
  print("b is not greater than a")
```

```
b is not greater than a
```

In [2]:

```
a = 200
b = 33

if a > b:
  print("a is greater than b")
```

```
a is greater than b
```

In [4]:

```
a = 2
b = 330

print("A") if a > b else print("B")
```

```
B
```

In [5]:

```
a = 330
b = 330

print("A") if a > b else print("=") if a == b else print("B")
```

```
=
```

In [9]:

```
a = 200
b = 33
c = 500
if a > b and c > a:
  print("Both conditions are True")
```

```
Both conditions are True
```

In [10]:

```
a = 200
b = 33
c = 500
if a > b or a > c:
  print("At least one of the conditions is True")
```

```
At least one of the conditions is True
```

In [8]:

```
x = 41

if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

```
Above ten,
and also above 20!
```

In [12]:

```
a = 33
b = 200

if b > a:
  pass

# having an empty if statement like this, would raise an error without the pass statement
```

In [1]:

```
i = 1
while i < 6:
  print(i)
  i += 1
```

```
1
2
3
4
5
```

In [14]:

```
i = 1
while i < 6:
  print(i)
  if (i == 3):
    break
  i += 1
```

```
1
2
3
```

In [15]:

```
i = 0
while i < 6:
  i += 1
  if i == 3:
    continue
  print(i)

# Note that number 3 is missing in the result
```

```
1
2
4
5
6
```

In [16]:

```
i = 1
while i < 6:
  print(i)
  i += 1
else:
  print("i is no longer less than 6")
```

```
1
2
3
4
5
i is no longer less than 6
```

In [17]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

```
apple
banana
cherry
```

In [ ]:

```
for x in "banana":
  print(x)
```

In [19]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```

```
apple
banana
```

In [20]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    break
  print(x)
```

```
apple
```

In [21]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
  print(x)
```

```
apple
cherry
```

In [22]:

```
for x in range(6):
  print(x)
```

```
0
1
2
3
4
5
```

In [23]:

```
for x in range(2, 6):
  print(x)
```

```
2
3
4
5
```

In [24]:

```
for x in range(2, 30, 3):
  print(x)
```

```
2
5
8
11
14
17
20
23
26
29
```

In [25]:

```
for x in range(6):
  print(x)
else:
  print("Finally finished!")
```

```
0
1
2
3
4
5
Finally finished!
```

In [26]:

```
for x in range(6):

  if x == 3: break
  print(x)
else:
  print("Finally finished!")

#If the loop breaks, the else block is not executed.
```

```
0
1
2
```

In [27]:

```
adj = ["red", "big", "tasty"]
fruits = ["apple", "banana", "cherry"]

for x in adj:
  for y in fruits:
    print(x, y )
```

```
red apple
red banana
red cherry
big apple
big banana
big cherry
tasty apple
tasty banana
tasty cherry
```

In [1]:

```
for x in [0, 1, 2]:
  pass

# having an empty for loop like this, would raise an error without the pass statement
```

In [ ]:

```

```