

3 Conditions

The solutions to many programming problems require an action to occur only if a particular condition is true.

A condition can only result in true or false.

(These values are called Booleans. True and False are reserved words in the Python language. True and False are values / objects of type bool)

A condition often involves an expression which compares one value with another.

Comparisons between numerical values are made using the same comparison operators that are used in mathematics:

>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

and the equality / inequality operators:

==	equal to (the proper equals operator)
!=	not equal to

Boundary cases

Human languages are imprecise.

Does "every child over 5" mean "aged 5 and over" or "aged 6 and over"? The difference between \geq and $>$ is very important.

If you mean "every child aged 5 and over" you can express this as $\text{age} > 4$ or $\text{age} \geq 5$.

Any lack of clarity needs to be removed at the design phase. Always clarify before coding. The boundary cases are where errors in programming often occur.

Boundary cases are the values at and just outside the specified limits.

In the case "aged 5 and over" the specified limit is 5 and the age values of 4 and 5 are the boundary cases.

Only one
condition can
execute

Only one
condition can
execute

```

conditions.py - C:/Users/khouripa/Documents/Year 11 programming/conditions.py (3.6.1)
File Edit Format Run Options Window Help

my_variable = 89

# example 1
if my_variable < 90:
    print("{} is less than 90".format(my_variable) )

#example 2
if my_variable < 89:
    print("{} is less than 89 ".format(my_variable) )

#example 3
if my_variable < 89:
    print("{} is less than 89".format(my_variable))
else:
    print("{} is not less than 89".format(my_variable))

#example 4
if my_variable < 89:
    print("{} is less than 89".format(my_variable))
elif my_variable == 89:
    print("{} is exactly 89".format(my_variable))
else:
    print("{} is more than 89".format(my_variable))

```

The code after the colon is “tabbed in” by 4 spaces.
The tabbed in code is what will run if the condition is met.

Running the code on pytutor : <http://www.pythontutor.com/visualize.html>

The screenshot shows the PyTutor interface with the following code and output:

```

Python 3.6
1 my_variable = 89
2
3 # example 1
4 if my_variable < 90:
5     print("{} is less than 90".format(my_variable) )
6
7
8 #example 2
9 if my_variable < 89:
10    print("{} is less than 89 ".format(my_variable) )
11
12
13 #example 3
14 if my_variable < 89:
15    print("{} is less than 89".format(my_variable))
16 else:
17    print("{} is not less than 89".format(my_variable))
18
19 #example 4
20 if my_variable < 89:
21    print("{} is less than 89".format(my_variable))
22

```

Print output (drag lower right corner to resize):

```

89 is less than 90
89 is not less than 89
89 is exactly 89

```

Global frame

```

my_variable 89

```

Program terminated

3.1 Activities

1. Consider a program that asks for a user's age and then:
 - if their age 12 or less, tells them that they should be at primary school,
 - if their age is more than 12 but less than 18, they need to go to secondary school
 - if they are 18 years old, it tells them that they have become an adult
 - and if they are older than 18 tells them that they must have left school.
2. To go to university a person needs to be:
 - over the age of 20
 - **or** be 16 or older and have level 3 NCEA
 - **and** (for both conditions) the person need to be competent with English.

Create a program that given the following variables:

age= 25

ncea = False

english = True

Will give a correct response about whether a person can go to university or not.

Test with other values.