# Python cheat sheet

## General

Python is one of many **programming languages**.

Code is made up of **statements** that use a variety of **commands**. Each statement goes on a new line.

A series of steps used to solve a problem is called an **algorithm**. Mostly you will be programming one instruction to happen after another. This is called a **sequence**.

## Variables

Lets you **store values** to use later (known as assigning a value)

Use names that make sense, e.g. *firstname*, *surname* NOT *a*, *b*

Can't use spaces or start with a number, e.g. ~~first name~~, ~~2name~~

Underscores or capitals can be useful, e.g. first_name, FirstName

```
firstname = 'Bob'
age = 32
married = TRUE
```

You can do calculations with variables

```
area = height * width
       expression
```

You can do calculations with the variable itself

```
score = score + 1   #increases score by 1
```

Don't have to create a variable beforehand, just start to use it. Python automatically figures out the **data type** of a variable.

Variables with a fixed value (never changes) are called **constants**

```
pi = 3.142
```

## input

Gets information from the user and stores it in a **variable**.

```
print('Enter value:')
variable=input()
```

Alternative method that does it all on one line:

```
variable=input('Enter value:')
```

Input( ) always gives a string value. Use **int( )** or **float( )** with it if you are working with numbers.

## Errors

Sometimes Python will tell you. Sometimes it won't!

**Syntax error:** error in the way the statement should be written:

```
print('Enter value)    #missing quote!
```

**Logic error:** error in the algorithm that means the outcome is not what is expected

```
if age < 18:
    print('You are old enough to vote.')
```

## print

**Displays** text and other information on the screen.

You can use 'single quotes' or "double quotes" (don't mix).

```
print('Hello world!')
print("Hello world!")
```

To show a quote use a backslash before it (escape character)

```
print('A computer\'s processor')
```

You can mix text and variables together:

```
print('Score is',score,'points')
print('Score is '+score+' points')
```

Using + is called **concatenating** (stringing together):
Using a comma gives a space. Using + gives no space.

## Comments

Lets you add **notes** to your code to explain what you have done.

```
#This is a comment
print('Another one?') #This is another
```

Always use comments to **explain your code**. Imagine you are giving it to someone else - help them to understand it.

## Data types

Variables can store different types of data

**String** ..... *any text (including numbers and symbols)*
**Integer** ..... *whole numbers (e.g. 5)*
**Real number (or Floating point number)**
..... *decimal number (e.g. 3.2)*
**Boolean** ..... *True or False value (or Yes/No or 0/1)*
**Character** ..... *One single letter only (e.g. A)*

## int( ), str( ), float( )

Changes the **data type** of a variable to an **integer**, **string** or **floating point number** .

To turn a variable's value into an **integer**:

```
variable=int(variable)
```

Useful when used with **input** if you want to work with numbers:

```
print('Enter value:')
variable=input()
variable=int(variable)
```

Create floating point numbers in a similar way

```
variable=float(variable)
```

To turn a variable's value into a **string**:

```
variable=str(variable)
```

Useful when **concatenating** text in **print** (has to be a string):

```
print('You have'+str(score)+'points')
```

**Programming tips:**

- plan things out
- start simple, build up
- use separate lines rather than combining into one
- use comments to explain what your code does

The Redhill Academy

python

## Selection (if, else and elif)

Use **if** when you need to choose different options depending on the value of something.

```
if lives==0:
    print('Game over!')
    print('Thank you for playing')
print('Lives left:',lives)
```

*condition* — *code block*

The indented code block is only used if the **condition** is true.
The un-indented lines after the code block are always used.
A double equals sign (==) is used as an **operator** when comparing (see box below for more operators)

Use **else** when you also want to do things if the **condition** is *not* true

```
if lives>0:
    print('Lives left:',lives)
else:
    print('Game over!')
    print('Thank you for playing')
print('This game has been designed by…')
```

*code blocks*

The first indented code block is only used if the **condition** is true.
The second indented code block is used if the **condition** is false.

Use **elif** (short for **else if**) when you want to use a further IF

```
if lives>=2:
    print('Lives left:',lives)
elif lives==1:
    print('Only 1 life left!')
else:
    print('Game over!')
print('This game has been designed by…')
```

The first indented code block is used if the first **condition** is true.
The second indented code block is used if the second **condition** is true. Finally, the third indented code block is used if neither of the **conditions** were true.
You can use **elif** more than once if you need more **conditions**.

Tip: when planning out **if** statements, use the word '**then**':
**IF…. THEN…. ELSE…**

## Operators

Arithmetic operators (used in calculations with variables)

**+** (add) **-** (subtract) ***** (multiply) **/** (divide)

Comparison operators (used in IF and WHILE statements)

| | | | |
|---|---|---|---|
| == | equal to | >= | greater than or equal to |
| > | greater than | <= | less than or equal to |
| < | less than | != | not equal to (can also use <>) |

Logical operators (used in IF and WHILE statements)

**AND** **OR** **NOT**

## Iteration (for loops)

Use a **for loop** when you need to **repeat some code a certain number of times**

```
for count in range(1,5):
    print (count)
print('Done')
```

*code block*

```
1
2
3
4
5
Done
```

**For loops** use an indented code block exactly like if statements do.

**For loops** can also be used to run through each letter of a string value in turn:

```
language='python'
for letter in language:
    print ('Letter:',letter)
```

```
Letter: p
Letter: y
Letter: t
Letter: h
Letter: o
Letter: n
```

## Iteration (while loops)

Use a **while loop** when you need to **repeat some code until a condition is met**

```
count=1
while count<=5:
    print (count)
    count=count+1
print('Done')
```

*condition* — *code block*

```
1
2
3
4
5
Done
```

**While loops** also use an indented code block.

You can exit out of a while loop early using **break**:

```
count=1
while count<=10:
    print (count)
    count=count+1
    if count==6:
        break
print('Done')
```

```
1
2
3
4
5
6
Done
```

## Random numbers

You can generate random numbers using this code:

```
import random
number=random.randint(1,10)
print(number)
```

In this example a random number is generated between 1 and 10.
The **first line is needed** at the beginning of your program otherwise the random function will not work.

**Programming tips:**

- plan things out
- start simple, build up
- use separate lines rather than combining into one
- use comments to explain what your code does

The Redhill Academy