

# Munster Programming Training



University College Cork  
*Computer*  
**Science**  
*Department*

# What is Python?



Python is a programming language.

- It communicates to the computers what to do “line after line”.
- It interprets each line from the natural language to the computer language.
- It solves various problems from scientific to entertaining.

Why Python?

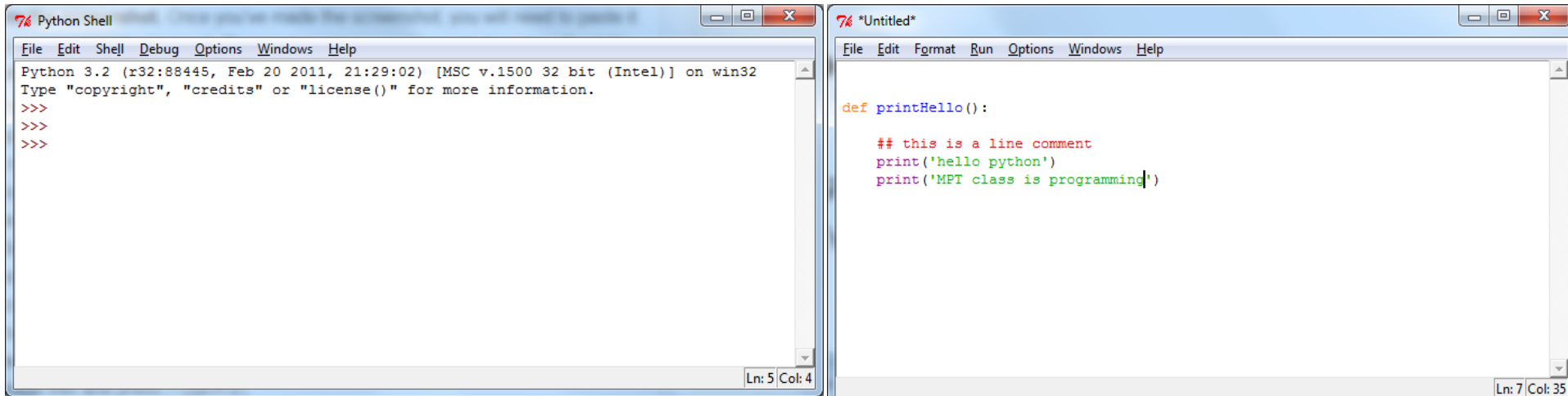
- It is simple, interpreted, procedural, object oriented etc.
- It has impact in several areas ... internet, networking, etc
- It can be used to solve scientific problems.

# What is Python

GUI tool to develop and interpret Python modules.

Two different windows:

1. Interpreter windows for executions
2. Editing window for developing / writing modules



# What is a Python Program?



Python programs are sequence of lines described in a “natural language”.

These lines include:

- ☒ Python comments
- ☒ Python key words e.g. def, import, if, for, range etc.
- ☒ Python identifiers (names) for variables, methods etc.
- ☒ Python values for numbers or text
- ☒ Python expressions

These lines are written with a certain semantics.

- ☒ One action / statement per line
- ☒ **Use of indentation**

# Python Comments



Python comments are lines that are not interpreted / executed.

Comments are for ourselves to give information / explanation about the code.

`# make a line comment`

`''' something in between these represents a block comment '''`

A block comment at the beginning of a method generates documentation.

We can read it in the interpreter window as `help`.

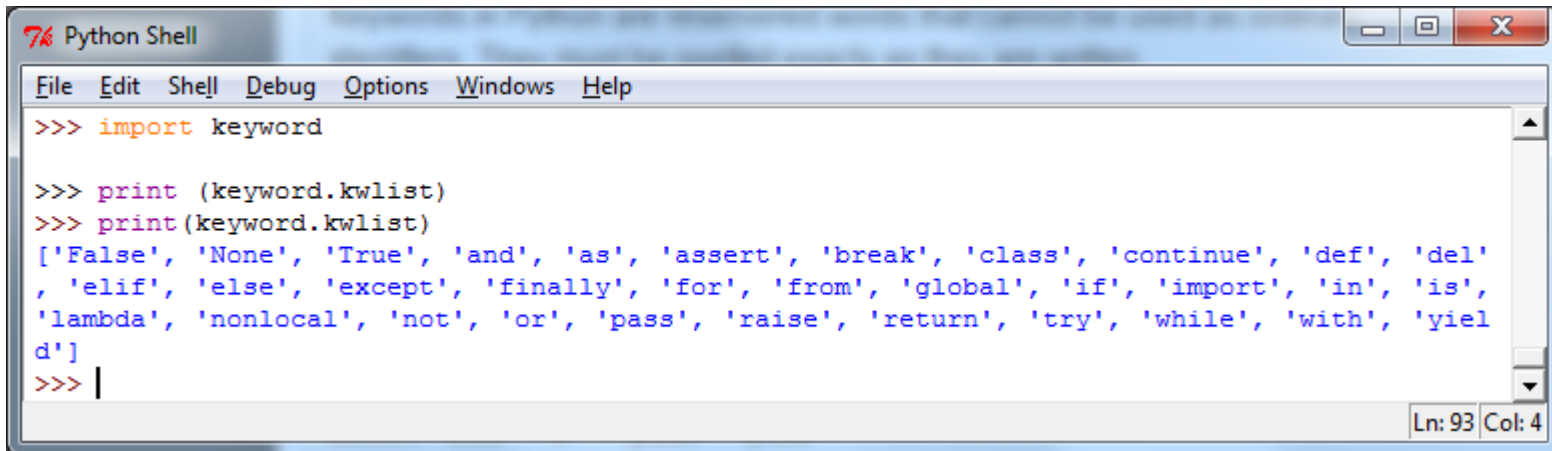
# Python Keywords

Keywords are reserved word for Python statements, datatypes etc.

Data types: **int, float, str, etc**

Statements: **if, while, for, etc**

These words cannot be used as user names / identifiers.

A screenshot of a Python Shell window titled "Python Shell". The window has a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main text area shows the following code:

```
>>> import keyword  
  
>>> print (keyword.kwlist)  
>>> print(keyword.kwlist)  
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del',  
, 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is',  
'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']  
>>> |
```

The status bar at the bottom right indicates "Ln: 93 Col: 4".

# Python Identifiers



Identifiers are names for Python variables, functions, etc.

Rules to name an identifier:

- ☒ Use only letters and digits 0-9 and \_
- ☒ Do not start with a digit
- ☒ Do not use Python Key words
- ☒ Python is case sensitive so pay attention to letter capitalisation

Golden rules

- ☒ Use meaningful names formed with multiple words.
- ☒ Start the first word with low case letter and any subsequent word with capital e.g. solveEquation, profitRate etc

# Python Values



Values represent integer or real numbers, or text Python programs.

Values can be as follows:

- ☒ **int**(eger) - sequence of digits e.g. 100, 0, 123 etc
- ☒ **float** (real numbers) - sequence of digits with a decimal point e.g. 1.0, 1.23
- ☒ **str**(ing of characters) or text – sequence of characters between ' '

Python datatypes are int, float and str.

Values can be used in Python expressions.



# Python int Expressions



Python int values are integers between -2147483648 to 2147483647.

Expressions use operators and brackets ().

int operators are as follows:

- + , - , \* , / for the main arithmetical operations

- \*\* for the exponent

- // for the results of the integer division

- % for the remainder of the integer division

# Python float Expressions



Python float values are number with a decimal point.

Python float values are represented with “some exact decimals”.

float operators are as follows:

+ , - , \* , / for the main arithmetical operations

\*\* for the exponent

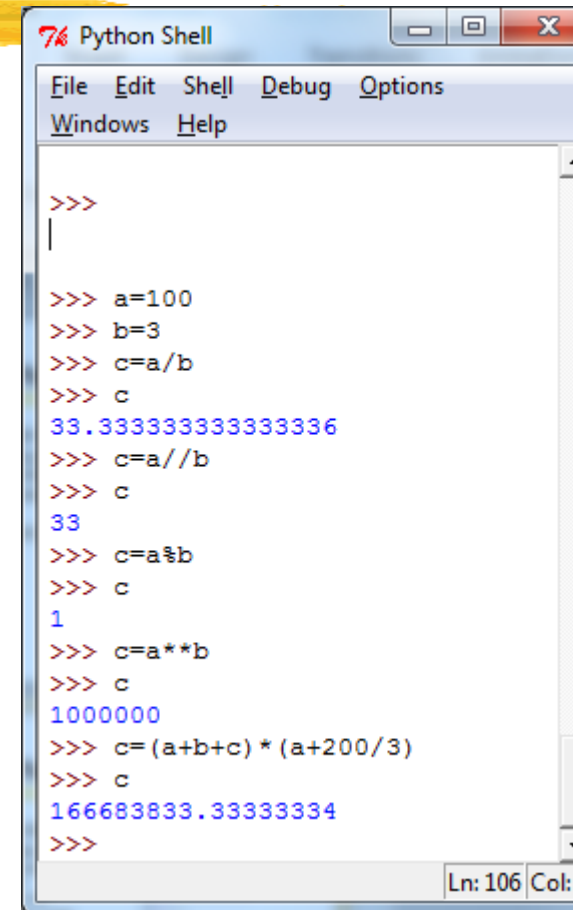
# Example of Expressions

```
>>>a=100
```

```
>>>b=3      # What is the nature of a, b?
```

```
>>>c=a/b     # What is the nature of c?
```

```
>>>c=a//b    # What is the nature of c?
```



```
Python Shell
File Edit Shell Debug Options
Windows Help

>>>
>>> a=100
>>> b=3
>>> c=a/b
>>> c
33.333333333333336
>>> c=a//b
>>> c
33
>>> c=a%b
>>> c
1
>>> c=a**b
>>> c
1000000
>>> c=(a+b+c) * (a+200/3)
>>> c
166683833.33333334
>>>
```

# Variables

Variables are Python identifies which carry a value.  
a, b, c are all variables.

**Variables must be initialised / assign value before used.**

Variable assignments:

var = value

var = expression

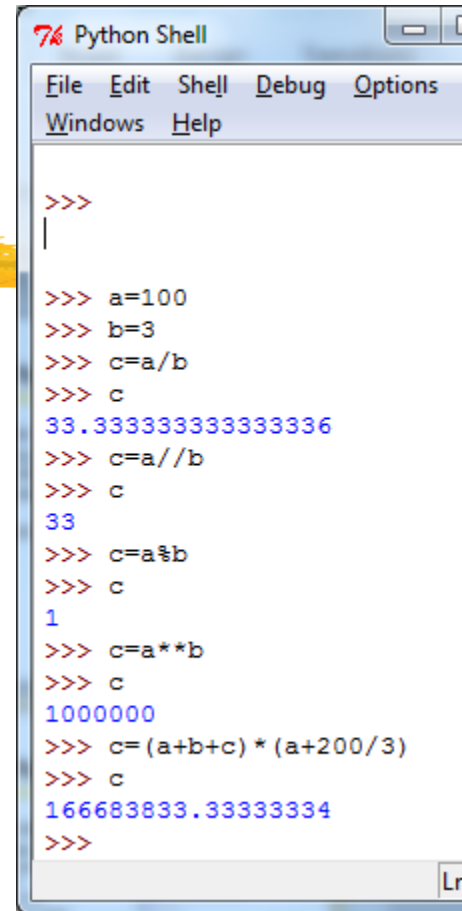
var1, var2 = value1, value2

Example:

a = 100

c = a+b

a, b = 100, 3

A screenshot of a Python Shell window. The title bar says 'Python Shell'. The menu bar includes 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The shell shows a series of commands and their outputs: an empty prompt, 'a=100', 'b=3', 'c=a/b' (output: 33.333333333333336), 'c=a//b' (output: 33), 'c=a%b' (output: 1), 'c=a\*b' (output: 1000000), and 'c=(a+b+c)\*(a+200/3)' (output: 166683833.33333334).

```
>>>  
|  
  
>>> a=100  
>>> b=3  
>>> c=a/b  
33.333333333333336  
>>> c=a//b  
33  
>>> c=a%b  
1  
>>> c=a*b  
1000000  
>>> c=(a+b+c)*(a+200/3)  
166683833.33333334  
>>>
```

# print()

print() is to print any value or variable from the arguments

```
print(var)
print(var1, var2,...)
```

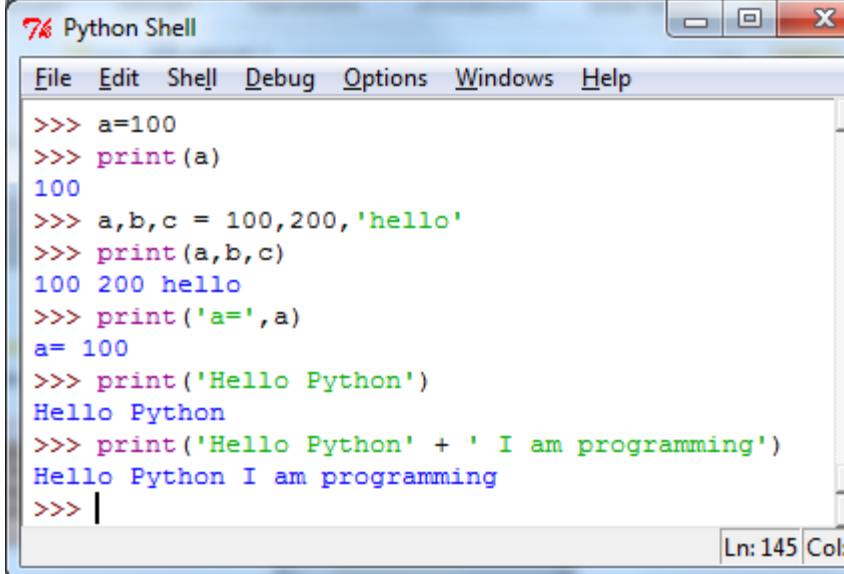
```
>>> print(a)    # prints the variable a
```

```
>>> print(a,b,c) # prints the variables a,b,c
```

```
>>> print('hello python') # print text
```

```
>>> print('a=', a)
```

print the text 'a=' then the value of a



```
Python Shell
File Edit Shell Debug Options Windows Help
>>> a=100
>>> print(a)
100
>>> a,b,c = 100,200,'hello'
>>> print(a,b,c)
100 200 hello
>>> print('a=',a)
a= 100
>>> print('Hello Python')
Hello Python
>>> print('Hello Python' + ' I am programming')
Hello Python I am programming
>>> |
```

Ln: 145 Col: 1

# input()

input() is to read from keyboard until press 'enter'

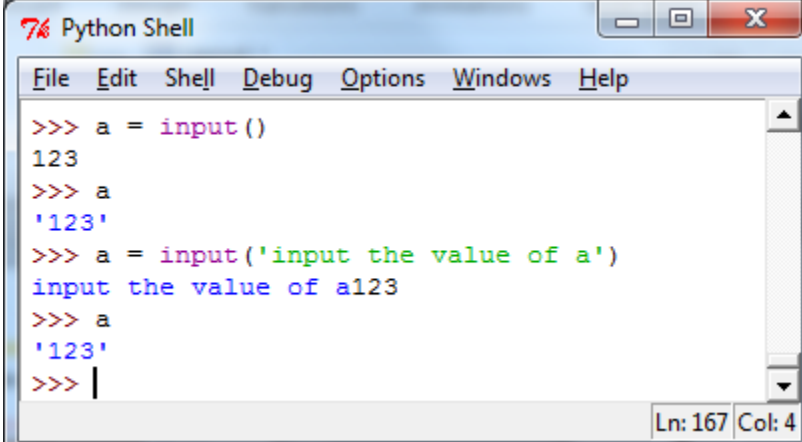
The function returns a string / text value with what it has typed

```
var = input()
var = input('message to print before typing')
```

```
>>> a=input() # read a value and assign it to a.
```

```
>>> a=input('input value for a')
```

```
# print the text and read the value
```



```
Python Shell
File Edit Shell Debug Options Windows Help
>>> a = input ()
123
>>> a
'123'
>>> a = input('input the value of a')
input the value of a123
>>> a
'123'
>>> |
```

Ln: 167 Col: 4

# Value Conversion



input() always returns a str value.

Conversion is possible using `type(value)` which converts value to the specified type.

```
>>> a = '123'           # a is a str variable
>>> b = int(a)          # b is an int variable
>>> a = int(input('type the value of a; '))
```

```
var = type(input('var='))
```

# What is a Python program like?

A Python program is made of some functions that can be executed.

Sections of a Python program

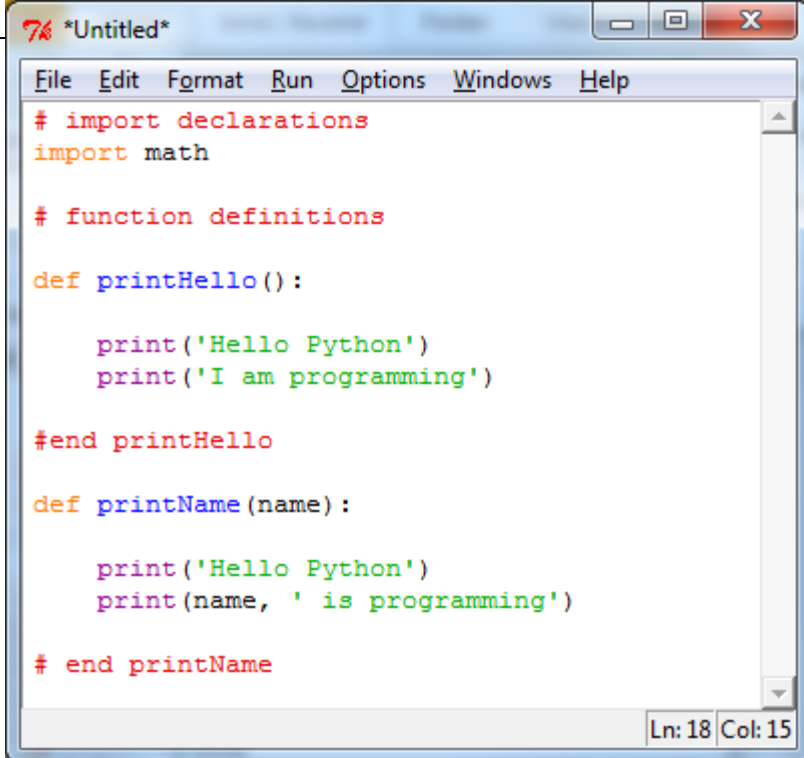
1. import libraries declarations
2. function definitions

```
def functionName(a1,a2,...):
```

```
    line1
```

```
    line2
```

```
    ....
```



```
*Untitled*
File Edit Format Run Options Windows Help
# import declarations
import math

# function definitions

def printHello():

    print('Hello Python')
    print('I am programming')

#end printHello

def printName(name):

    print('Hello Python')
    print(name, ' is programming')

# end printName

Ln: 18 Col: 15
```

Any function can be called / executed on the interpreted.



# Problem Solving



Write a Python program to solve the equation  $a*x**2 + b*x + c = 0$ .

Input Variables: a, b, c float

Output Variables: x1, x2 as float

Calculation:

$$x1 = (-b + \sqrt{b*b - 4*a*c}) / (2*a)$$

$$x2 = (-b - \sqrt{b*b - 4*a*c}) / (2*a)$$

Python program with one function:

- 1) Input a,b,c
- 2) Calculate x1, x2
- 3) Print x1, x2

# Problem Solving

```
*equation1.py - C:/Users/stabirca/Desktop/python/equation1.py*
File Edit Format Run Options Windows Help

# import modules
import math
import cmath

def solveEquation():

    print('\n\n===== \n\n')

    a = float(input('a='))
    b = float(input('b='))
    c = float(input('c='))

    d = b**2-4*a*c

    x1 = (-b+math.sqrt(d))/(2*a)
    x2 = (-b-math.sqrt(d))/(2*a)

    print('x1=', x1)
    print('x2=', x2)

    print('\n\n===== \n\n')

* end solveEquation
```

Ln: 26 Col: 19

```
Python Shell
File Edit Shell Debug Options Windows Help

>>>
>>> solveEquation()

=====

a=1
b=4
c=3
x1= -1.0
x2= -3.0

=====

>>> |
```

Ln: 201 Col: 4

# To do List



1. Read Chapter 3 from the Python tutorial.
2. Solve the HW problems