# Fundamentals of Programming 1

## Lecture 1 – Introduction

Aurelia Power, ITB, 2018

## Module Aims

- To introduce fundamental concepts of **software development**, specifically, problem solving and computer programming;
- Teach you a **structured approach to programming**;
- Teach you the fundamentals of programming using **Python**;
- Develop **problem solving skills**;
- Teach you **best practices in programming**;
- Gain practical experience in **designing, coding and debugging programs** in Python.

## Module Content and Syllabus Overview

A. **Basics:**

1. Software development lifecycle;
2. Elementary data types;
3. Basic Input and Output;
4. Arithmetic and Logical Operations;
5. Algorithms and program design.

# Module Content and Syllabus Overview

**B. Control Structures:**

1. Sequence;
2. Selection;
3. Iteration.

## Module Content and Syllabus Overview

C. **Functions:**

1. Modularity and Functions;
2. Function Declaration;
3. Types of functions based on parameters and return types.

## Module Content and Syllabus Overview

**D. Implementation:**

1. Program Testing;
2. Syntax, Logical and Runtime Errors;
3. Debugging/Fixing.

## Some Online Resources

- **Website: The Python Standard Library:**

https://docs.python.org/3/library/index.html

- **Website: Python for Beginners:**

https://www.python.org/about/gettingstarted/

- **Website: Python Tutorials:**

https://wiki.python.org/moin/BeginnersGuide/Programmers

- And many others…

## Assessment

A. **Continuous Assessment (CA)** carried out throughout the semester – **50%:**
   1. **MCQ – 15%**
   2. **MCQ – 15%**
   3. **Practical exam – 20%**

- **Dates to be confirmed as soon as possible**

B. **Formal Written Exam** sometimes during the January examination period (date and time to be confirmed) – **50%**

## Weekly Hours and Some Ground Rules

- **2 hours lecture** – all students attend in the same time on Wednesday from 2pm to 4pm.

- **2 hours practical work** in the lab – each individual group has its own designated time.

- **NOTE:** before each lab session, you must go over the lecture notes, as each lab is based on a corresponding lecture and it aims to allow you to practice and implement concepts introduced during the lecture.

## Weekly Hours and Some Ground Rules

- All material introduced in the module is examinable.
- All material introduced this semester is a pre-requisite for next semester.
- You are welcome to bring your own laptops during labs (not during the lecture).
- Ask questions relevant to the module.

## Moodle Page for FOP1- Enrollment

- Choose the following module:

  COMP1032 - Fundamentals Of Programming 1

- Enrolment Key:

  **FOP1_2018**

- Access course materials, such as lecture notes, lab worksheets, etc.
- MCQ and practical tests will also take place on Moodle.

## Computer Programs

- **A computer is a sophisticated machine** that:
  - stores data (numbers, words, images, etc.);
  - interacts with other devices (screen, printer, etc.);
  - executes programs (MS Word, MP4 Player, NotePad Editor, etc.)
- Computers can execute **repetitive tasks** over and over again.
- Computers can execute a wide range of **complicated and sophisticated tasks** such as electronic banking, running security scans, gaming etc.
- But… a computer must be given basic precise instructions.

## Computer Programs

- Computers execute **very basic instructions in rapid succession:**

  - put a red dot on the screen

  - add up 2 numbers

  - if this value is negative, add another number, etc.

- Computers must be told exactly what to do in a certain language.

- A **computer program** tells the computer in detail what steps it needs to take to complete a task, so..

  **A computer program is a sequence of simple/basic instructions and decisions written in a programming language.**

## Programming Languages

- A program is written by a developer in a **special language** such as:
  - C#
  - Java
  - **Python** (3)
  - etc.

## Python…

- Source:

https://www.google.com/imgres?imgurl=https%3A%2F%2Fi.ytimg.com%2Fvi%2F1t7uo2fCdHY%2Fhqdefault.jpg&imgrefurl=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3D1t7uo2fCdHY&docid=XPT0kVqnczyA_M&tbnid=FuJ0DMRQPO6bUM%3A&vet=1&w=480&h=360&client=avast&bih=679&biw=1536&ved=2ahUKEwjWk4_aqbzdAhVLzYUKHev6BwkQxiAoAnoECAEQFg&iact=c&ictx=1

## Python…

- But it is also a **general purpose programming language**
- Created by **Guido van Rossum** in the early ‘90s.
- A high level language

  - it is closer to human language than to machine language

  - it has high level data structures already built-in and ready to be used
- An interpreted language :

  - no compilation or linking is necessary

  - the sequence of instructions is executed directly and instant feedback is provided
- Modular/Object Oriented language:

  - allows you to create modules/data structures that can be re-used

## Why Python? Some advantages…

1. Easy to use:

- its syntax is easy to grasp;
- Programs are typically much shorter than in other languages such as C or Java;
- Statements grouping is achieved through indentation instead of brackets;
- No need to declare data type for variables == is a dynamically typed language

2. Readability:

- Because of the simple and clear syntax
- Because of the indentation requirements

## Why Python? Some advantages…

3. Free/open source:

- The Python Software Foundation distributes pre-made binaries that are freely available for use on all major operating systems called CPython

4. Cross-Platform:

- It can run on all major platforms/OSs: **Windows**, Linux and MacOs

5. Large Support:

- Wide and active support community

6. Large Library of standard modules ready to use in our programs:

- Create files, accessing data bases, etc.

7. Extensible:

- Extensive collections of add-on modules, libraries, toolkits, etc.

## Why not Python? Some disadvantages…

1. Speed:
- because it is interpreted

2. Memory and CPU intensive
- same as above

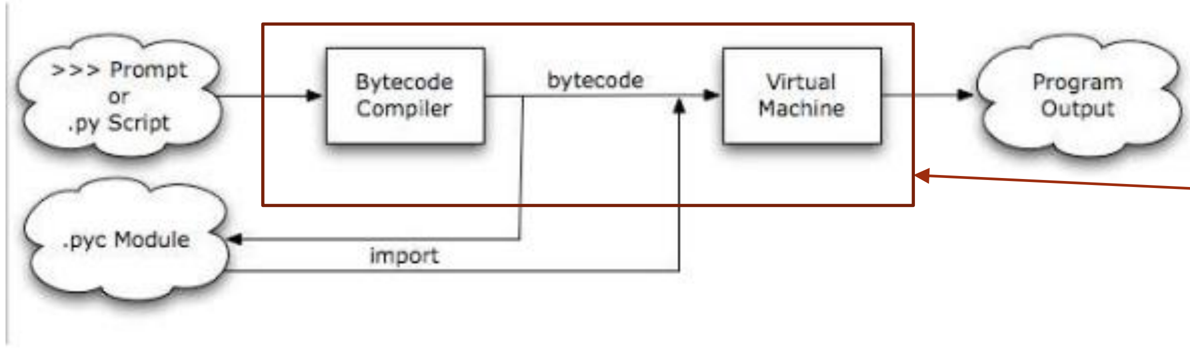
3. Runtime errors
- Because it is dynamically typed, it has the potential of many errors/bugs when you run the program
- it requires more testing and is less secure

4. Little support for mobile computing
- Because of the above issues

**But**, all considered…Python is a great choice of language to start with.

## The Python Interpreter

- Python is interpreted language → the python interpreter runs a program by executing statements written in a text editor/file.



**The interpreter**(the shell) = a computer program that executes program instructions/commands in place, providing immediate feedback.

- The python interpreter consists of two parts:
  - A python bytecode compiler
  - A virtual machine which executes Python bytecode.

Source: http:\\trizpug.org

- The compiler component of the interpreter accepts humans readable expressions as input and produces machine readable bytecode
- The virtual machine component of the interpreter accepts the bytecode as input, it executes it, and it may output something tangible.

## Python Programs

A Python Program (or a script) = a sequence of definitions and commands which are evaluated and executed by the Python interpreter;

A command (or a statement) = instructs the interpreter to do something; for instance, 'Hello year 1 students! Welcome to ITB! ' instructs the interpreter to output/print to the screen the message (also called a string) Hello year 1 students! Welcome to ITB!

## Python Programs

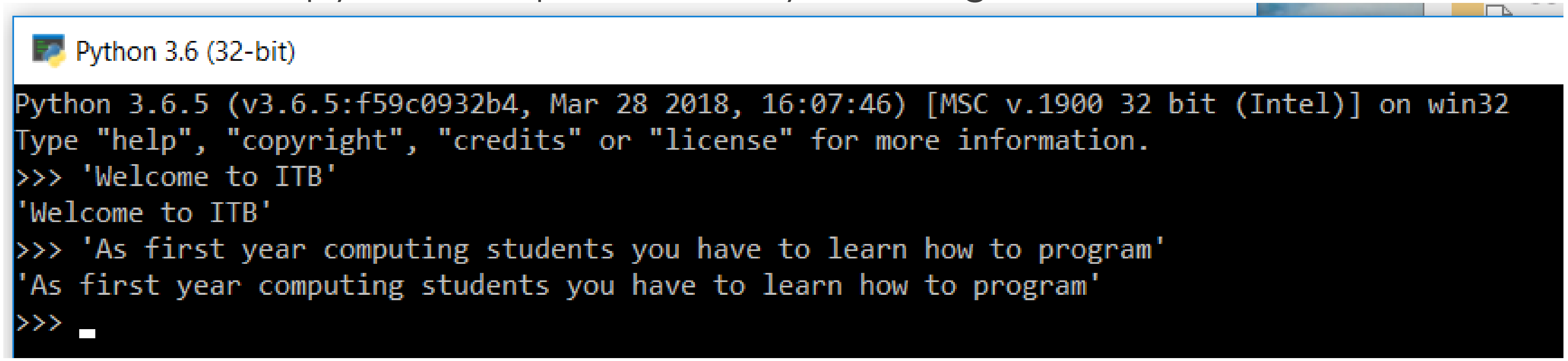
There are typically 3 main phases of a python program/script:

1. **Editing:** writing code in the text editor/to a python file → .py files
2. **Running**: The code is then interpreted and executed by the interpreter in 2 sub-phases:
    1. First the code is translated into **bytecode** == the py files are translated into .pyc files
    2. Then the interpreter executes the bytecode (the contents of the .pyc files) using the virt**ual machine**
3. **Outputting**: The program may or may not produce tangible, human interpretable output, such as messages, files, graphical components, etc.

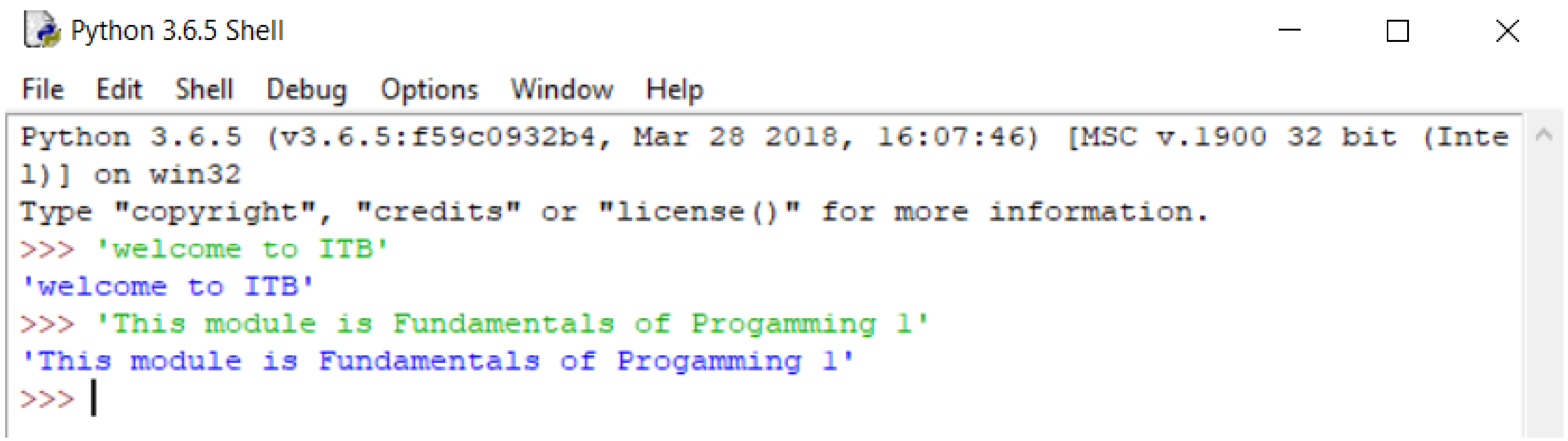
Note: the interpreter may use additional modules using the keyword *import*

## The Python Interpreter in action

- The interpreter can be used interactively to test out instructions on the fly.
- No need to write separate programs; it will give you immediately results.
- We can use the python interpreter directly or through the IDLE environment



```
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 'Welcome to ITB'
'Welcome to ITB'
>>> 'As first year computing students you have to learn how to program'
'As first year computing students you have to learn how to program'
>>>
```



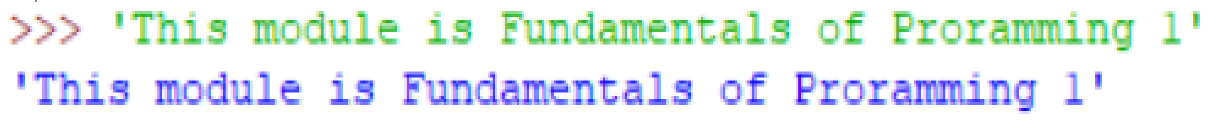
```
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Inte
l)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 'welcome to ITB'
'welcome to ITB'
>>> 'This module is Fundamentals of Progamming 1'
'This module is Fundamentals of Progamming 1'
>>>
```

# The IDLE Development Environment

- IDLE is an Integrated Development Environment (IDE) = a bundled set of software tools
- **IDLE's basics**:
  - A text editor to create and modify programs
  - A translator/the python shell for executing instructions
  - A program debugger to output error messages in order to find errors/bugs
- There are many other IDEs that you might try, such as Pycharm, Jupiter notebook, spider etc.
- IDLE shell prompts the user to input something using the >>> symbols, and it immediately responds
- Unlike the cmd, IDLE allows us to save the code we wrote and then run/modify it later as needed.

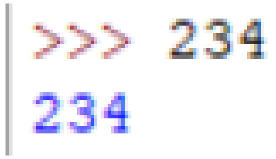
# Some simple commands in IDLE shell…

- Simple messages/strings



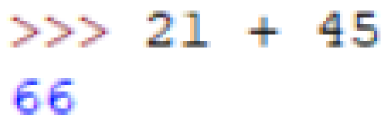
```
>>> 'This module is Fundamentals of Proramming 1'
'This module is Fundamentals of Proramming 1'
```

- Numbers



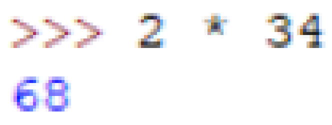
```
>>> 234
234
```

- Addition



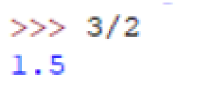
```
>>> 21 + 45
66
```

- Multiplication



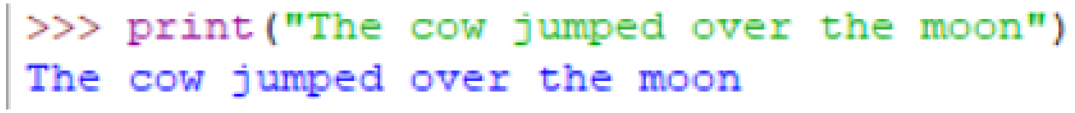
```
>>> 2 * 34
68
```

- Division



```
>>> 3/2
1.5
```

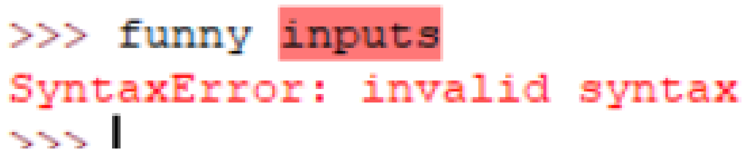
- Using the **print** function to output; note the difference: the message is no longer within the quotation marks



```
>>> print("The cow jumped over the moon")
The cow jumped over the moon
```

- Now let's type in: **funny inputs**

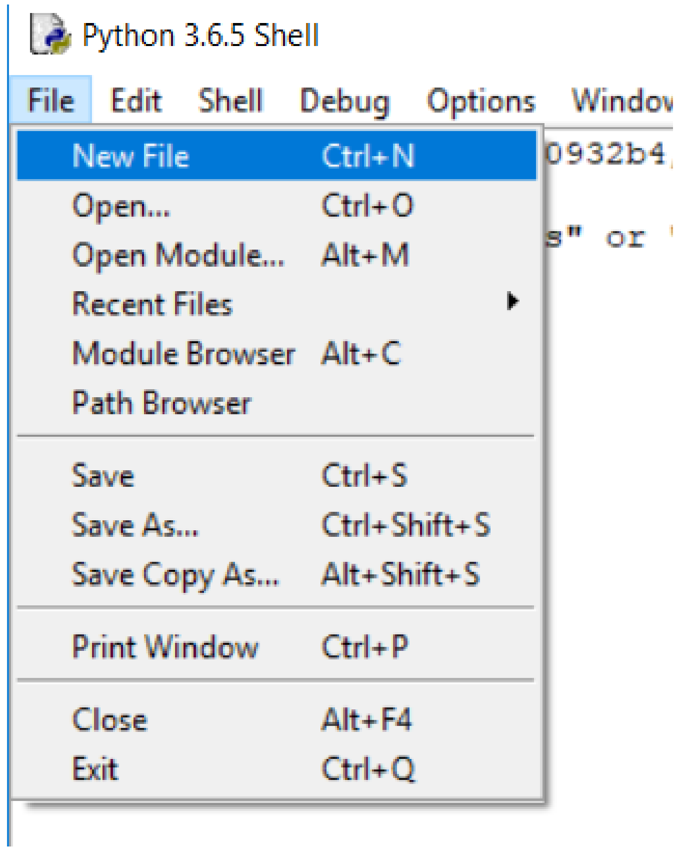
**You will get an error because the message funny inputs must be enclosed inside quotation marks**



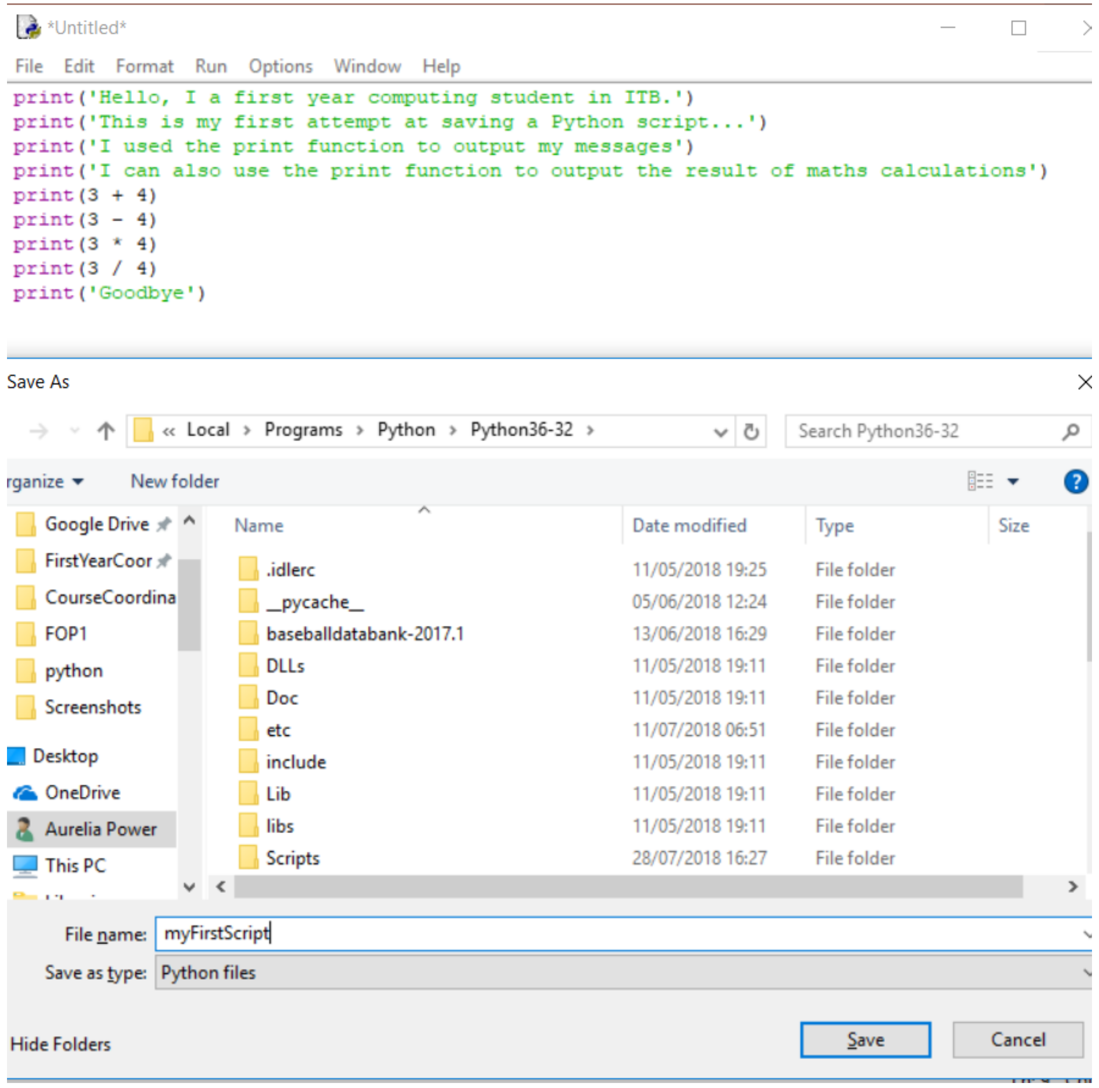
```
>>> funny inputs
SyntaxError: invalid syntax
>>> 
```

# Saving programs in IDLE…

- Open the IDLE shell and go to File, then select New File



- Start typing in commands in the untitled file, then save it using a name of your choice …



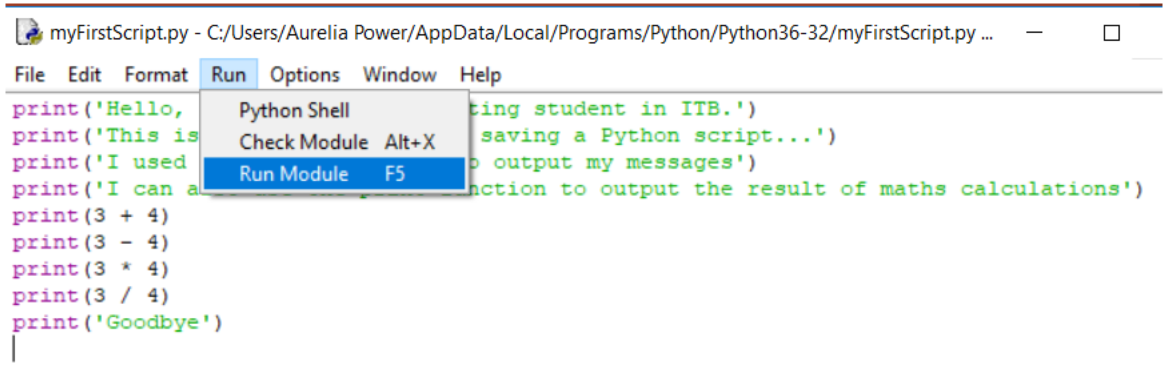
```
print('Hello, I a first year computing student in ITB.')
print('This is my first attempt at saving a Python script...')
print('I used the print function to output my messages')
print('I can also use the print function to output the result of maths calculations')
print(3 + 4)
print(3 - 4)
print(3 * 4)
print(3 / 4)
print('Goodbye')
```

| Name | Date modified | Type | Size |
|---|---|---|---|
| .idlerc | 11/05/2018 19:25 | File folder | |
| __pycache__ | 05/06/2018 12:24 | File folder | |
| baseballdatabank-2017.1 | 13/06/2018 16:29 | File folder | |
| DLLs | 11/05/2018 19:11 | File folder | |
| Doc | 11/05/2018 19:11 | File folder | |
| etc | 11/07/2018 06:51 | File folder | |
| include | 11/05/2018 19:11 | File folder | |
| Lib | 11/05/2018 19:11 | File folder | |
| libs | 11/05/2018 19:11 | File folder | |
| Scripts | 28/07/2018 16:27 | File folder | |

File name: myFirstScript
Save as type: Python files

# Running Saved programs in …

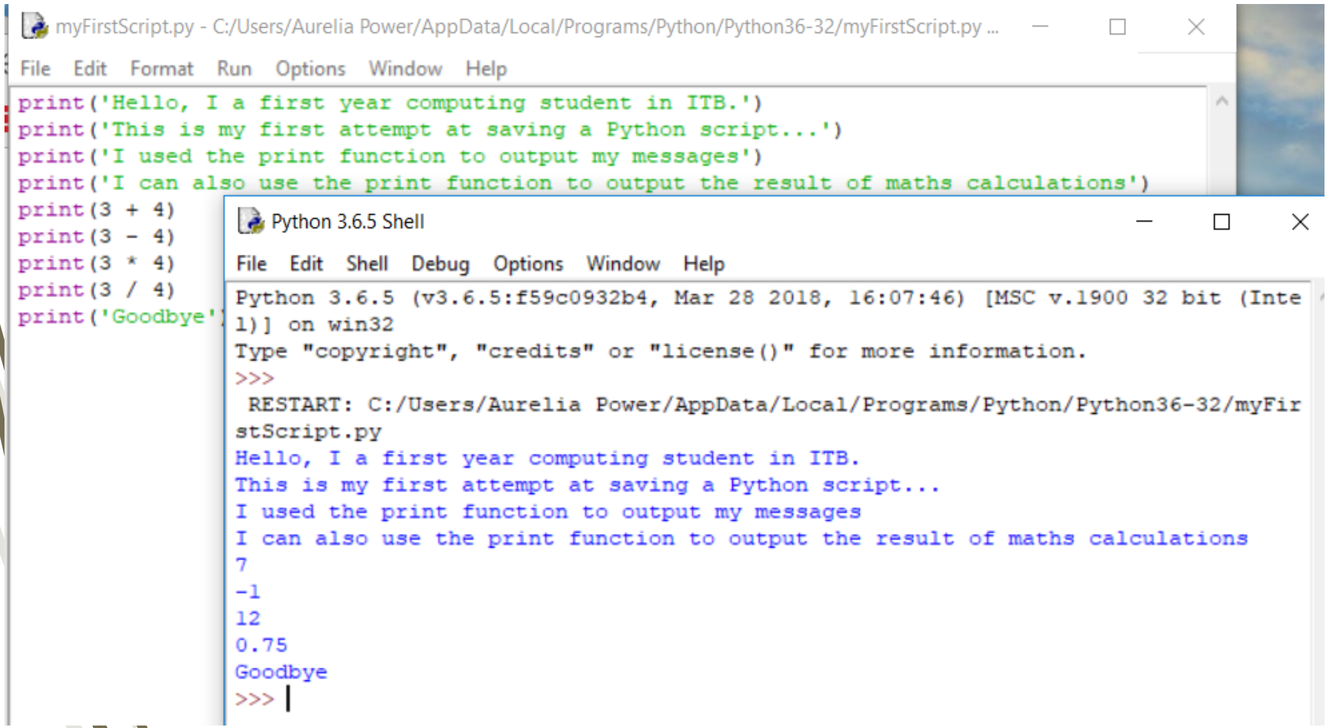
- Select Run, then Run Module



myFirstScript.py - C:/Users/Aurelia Power/AppData/Local/Programs/Python/Python36-32/myFirstScript.py ...
File Edit Format Run Options Window Help
Python Shell
Check Module Alt+X
Run Module F5

```
print('Hello,      ting student in ITB.')
print('This is      saving a Python script...')
print('I used       o output my messages')
print('I can a      nction to output the result of maths calculations')
print(3 + 4)
print(3 - 4)
print(3 * 4)
print(3 / 4)
print('Goodbye')
```

- This will invoke the interpreter and it will instantly execute the commands and produce output



myFirstScript.py - C:/Users/Aurelia Power/AppData/Local/Programs/Python/Python36-32/myFirstScript.py ...
File Edit Format Run Options Window Help
Python 3.6.5 Shell
File Edit Shell Debug Options Window Help

```
print('Hello, I a first year computing student in ITB.')
print('This is my first attempt at saving a Python script...')
print('I used the print function to output my messages')
print('I can also use the print function to output the result of maths calculations')
print(3 + 4)
print(3 - 4)
print(3 * 4)
print(3 / 4)
print('Goodbye')
```

```
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Inte
l)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:/Users/Aurelia Power/AppData/Local/Programs/Python/Python36-32/myFir
stScript.py
Hello, I a first year computing student in ITB.
This is my first attempt at saving a Python script...
I used the print function to output my messages
I can also use the print function to output the result of maths calculations
7
-1
12
0.75
Goodbye
>>> 
```

NOTE:
- To access a .py file you can go to File
Then select Open
**OR**
Select Open Module
**OR**
Select Recent Files (if you edited your file recently)

## Knowledge check

1. What is a computer program?

   A sequence of basic instructions and decisions written in a programming language

2. Name 2 advantages of Python?

   Ease of use and open source

3. Name 2 disadvantages of Python?

   Memory and CPU intensive and potential of many runtime bugs

4. What are the typical phases of a Python program?

   Editing, interpreting and outputting

5. What is the Interpreter?

   A program that executes commands in place and provides immediate feedback

6. What are the components of the Interpreter?

   The bytecode compiler and the virtual machine

7. What does IDLE include?

   Editor, translator and debugger.