# Day 1

Input – Output – Arithmetic – Variables

ORBTRONICS

# Objectives

## Students should be able to:

- Discuss what is programming and what are programming languages.
- Use the Python shell and create source files.
- Use the print and input functions.
- Understand sequential execution.
- Use arithmetic operations and variables effectively.
- Understand integer, floating point, and string types.
- Read and write comments and documentation.

# What is Programming?

# What is Programming?

- Programming is essentially telling a digital device to do something.
- This can be a desktop, laptop, phone, tablet
- Or any machine with digital electronics, e.g., cars, fridges, speakers, robots, etc.
- We tell the device what to do by giving it instructions to follow.
- For example, if you are making an app your instructions could be:
  1. Display two buttons on the screen.
  2. If the user clicks on the first button, then bring them to this other screen.
  3. On this other screen, display these photos.

# What is Programming?

- These instructions are specified in a language called a programming language.
- There are plenty of programming languages but the main one we will be using is called Python.

# Print Function

# Print Function

- Type print(“Hello World!”) into the python shell and press enter.
- What happens?
- Why do you think that happens?
- The print function in python is used to display a message onto the screen.
- Note that the message must have quotation marks and parentheses around it.

# Test out the print function

- Try printing out several messages about yourself.
  - For example: print(“I like programming!”)
- Try leaving out the parentheses or the quotation marks (or both).
- Try using single quotation marks ‘ ’ instead of double “ ”.
- Try putting two messages separated by a comma in the parentheses, e.g.
  - print(“Hello”, “My name is John”)

# Test out the print function

- Leaving out either the parentheses or the quotation marks (or both) gives an error.
- You can use single quotation marks ‘ ’ instead of double “ ”.
- Putting two messages in the parentheses causes both messages to be printed one the same line with a space between them.

# Arithmetic Operations

# Arithmetic Operations

- Enter each of the following lines of code into the Python shell and see what is displayed.

```
2 + 3
4.3 – 6.2
7 * -5
8 / 3
```

# Results

```
>>> 2 + 3
5
>>> 4.3 - 6.3
-2.0
>>> 7 * -5
-35
>>> 8 / 3
2.6666666666666665
>>>
```

# Arithmetic Operations

- As you can see, the Python shell gives the answer to each of the calculations.
- Multiplication is done using the asterisk (*) symbol.
- Division using the forward slash (/) symbol.
- Calculations can be done on whole numbers, negative numbers and numbers with decimal points.

# Integers and Floats

- In programming, positive and negative whole numbers are known as integers (ints).
- Numbers with decimal points are known as floating point numbers (floats).
- This is important to know since most programming languages treat them differently.
- For example, when 4.3 - 6.3 was calculated, the answer was displayed as -2.0 instead of -2 exactly. The reason is that since floats were used in the calculation, Python will give the answer in the form of a float, so it will display the number with a decimal point.

# More Arithmetic Operations

- Also try out the following and try to figure out why they give the answers that they do.

  5**2

  20 / 4

  21 // 4

  23 % 4

# Results

```
>>> 5**2
25
>>> 20 / 4
5.0
>>> 21 // 4
5
>>> 23 % 4
3
>>>
```

ORBTRONICS

# Arithmetic Operations

- The double asterisks ** is the exponent in Python. It means “to the power of”. So, 5**2 means “5 to the power of 2”, hence the answer of 25.
- You might be wondering why did 20 / 4 give an answer of 5.0 instead of 5 exactly. The reason is that in Python, when a division is calculated, the answer is automatically given in the form of a float.
- The double forward slash (//) is integer division. It is like regular division, but it gives the rounded-down answer instead of the exact answer. So, 21 // 4 gives an answer of 5 rather than 5.25.
- The percent sign is known as the modulo operator. It is used to find the remainder when of division. 23 divided by 4 is 5 with a remainder of 3. So, 23 % 4 is asking what is the remainder when dividing 23 by 4. Hence the answer shown is 3.

# Operations Summary

- Addition: +
- Subtraction: -
- Multiplication: *
- Division: /
- Integer Division: //
- Exponent: **
- Modulo: %

# PEDMAS / BODMAS

- An arithmetic calculation can have several operations and parentheses can be used as well.
- Python uses the regular PEDMAS / BODMAS order of operations in these situations.

# PEDMAS / BODMAS

Consider this example

```
>>> -2 + (3.0 - -4.1) * 2 / 6**2
-1.6055555555555556
>>>
```

In this example, the order of operations goes as follows:

- (3.0 - -4.1) is calculated first, giving 7.1
- 6**2 is calculated next, giving 36
- 7.1 * 2 is calculated next, giving 14.2
- 14.2 / 36 is calculated next, giving 0.4055…
- -2 + 0.04055 is calculated last, giving -1.6055…

ORBTRONICS

# Challenge

- Try calculating your age in seconds using the shell as a calculator.
- Try to do it as accurately as possible.
- Try to account for leap years, the number of months and days since your last birthday.
- Try to do as much of the calculations in the shell.

# Source Files

# Source Files

- In IDLE, if you click on “File” then “New File” a blank window will open. This is the editor.
- In the editor you can type and edit as many lines of code as you want without having to retype the code when you run it.
- You can click on “Run” then “Run Module” (or just press F5) to run the code.
- You may be asked to save it into a file. This sort of file is called a source file.
- Running the program will open the shell where the output is displayed.
- If the program does not work how wanted, you can close or minimize the shell and edit the program in the editor.

# Displaying Calculations

- Unlike the shell, just typing a calculation into the editor and running the program will not display the answer.
- It needs to be placed into a print statement.

Source File

```
print(5.0 + 2.25 * -1.5)
```

Output

```
1.625
>>>
```

# Variables

# Variables

- Variables allow us to give a name to a particular value and we can then use that name instead of retyping the value.

# Variables

Source File

```
x = 5
print(x)
y = x + 2
print(y)
x = 7
print(x + y)
```

Output

```
5
7
14
>>>
```

Explanation

- We create a variable called x and give it the value 5 using the equal sign. Whenever we use x in our program, Python will know that it represents the number 5. When we print x on the next line, the value 5 gets displayed in the output.
- We create another variable called y and give it the value of x+2 which gets calculated to be 7. Whenever we use y in our program, it will represent the number 7. We then print y so 7 gets displayed.
- We can change the value of x to 7. From now on in our program, x will represent 7 and not 5 anymore. When we print x+y we get 14 since both x and y are now equal to 7.

# String Variables

Source File

```
x = "Hello"
y = "everyone"
print(x, y)
```

Output

```
Hello everyone
>>> |
```

Explanation

- Variables can also be created for text. Text in programming are known as strings.
- Firstly, we create a variable called x and set it to the string “Hello”.
- Secondly, we create another variable called y and set to it the string “everyone”.
- Lastly, we print the two of them.
- Later, we will see more things we can do with string variables.

# Variables

- Variable names do not have to be a single letter. They can contain several letters, underscores, and numbers
- However, the names cannot start with a number.
- The reason why we must put text in quotation marks when we want to print it, such as with print(“Hello World”), is that the quotation marks tell Python that this is a string and not a variable.

# Input Function

# Input Function

- The input function allows the user of a program to enter data into the program.
- The data entered by the user is always treated as a string.
- If the program wants the user to enter a number, then the string must be converted to either an integer or float. This is done using the int and float functions.

# Input Function Example Program

```
print('Enter your name')
name = input()
print('Hello', name)
print('Enter your age')
age = int(input())
print('In 10 years you will be', age+10, 'years old')
print('Enter your height in metres')
height = float(input())
print('Your height is', height, 'm')
```

# Input Function Example Output

```
Enter your name
John
Hello John
Enter your age
25
In 10 years you will be 35 years old
Enter your height in metres
1.8
Your height is 1.8 m
>>>
```

# Input Function Example Explanation

- Firstly, we ask the user for their name and they entered John. This is assigned to the variable name. We then print out “Hello” and their name.
- We then ask for their age. Because age is an integer, we must convert it to an integer using the int function. We then print out their age in 10 years by adding 10 to it.
- We then ask for their height in metres which is a float, so it needs to be converted using the float function.