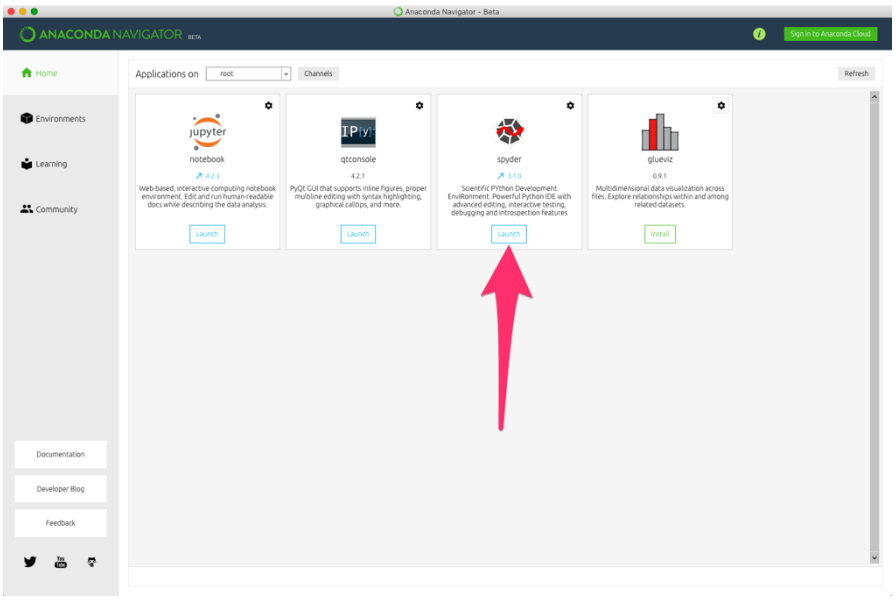
# QBUS6840: Tutorial 0 – Python Fundamental

## Objectives

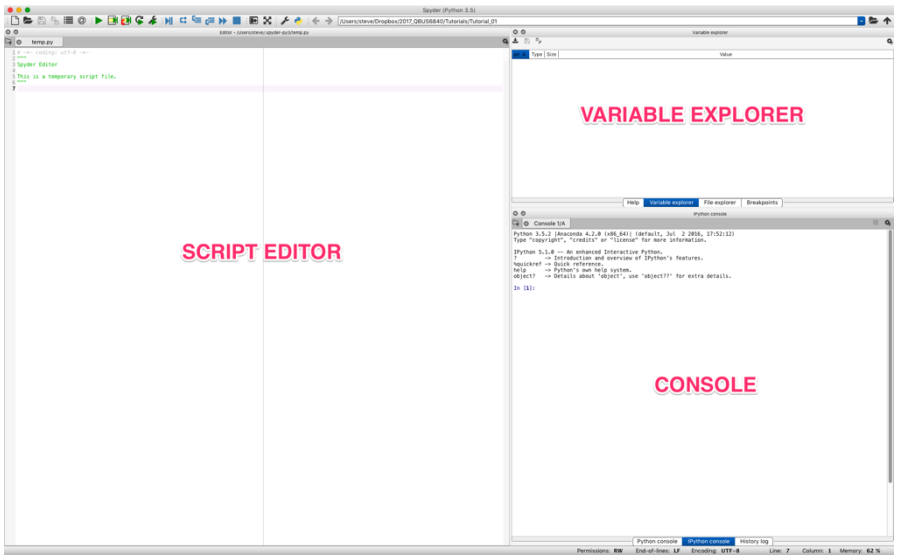
- To configure and become familiar with Anaconda and Spyder;
- To learn fundamental Python programming concepts;

### 1. Launch Spyder

First open Anaconda Navigator then click the “Launch” button for Spyder




Spyder will open to the following screen



## 2. Hello World

Python is an interpreted language. This means each piece of code is executed directly without previously compiling the code to a machine-readable format.

This means that Python can be used in an interactive manner.

Go to the Console at the bottom right of the Spyder window and type

```
print("Hello World!")
```

Congratulations! You have now written your first line of Python.

## 3. Variables

Python stores data in variables. Variables are analogous to variables that you would have experienced in mathematics.

Let's start by creating a variable called "x" with the value of "10".

```
x = 10
```

Then to confirm the value of x we can print its value like so

```
print(x)
```

Let's experiment with some more variables. Try the following

```
y = 5
z = x + y

print(z)
```

**What is the value of z?**

Let's try another example

```
y = 10
z = x + y

print(z)
```

**What happened to the value of y and z?**

We can do other mathematics operations on numbers. Try the following

```
print(z - 3)
print(z * 5)
print(z / 2)
```

**What do the operators * and / do?**

Variables aren't limited to numbers. They can be any defined data type. For example you can store a string as a variable. Try this:

```
my_string = "Hello World!"
print(my_string)

end_string = " -from Steve"
print(my_string + end_string)
```

Tip: You can check the state of variables in your current workspace by using the "Variable Explorer" in Spyder in the top right window pane.

### 4. Python Scripts

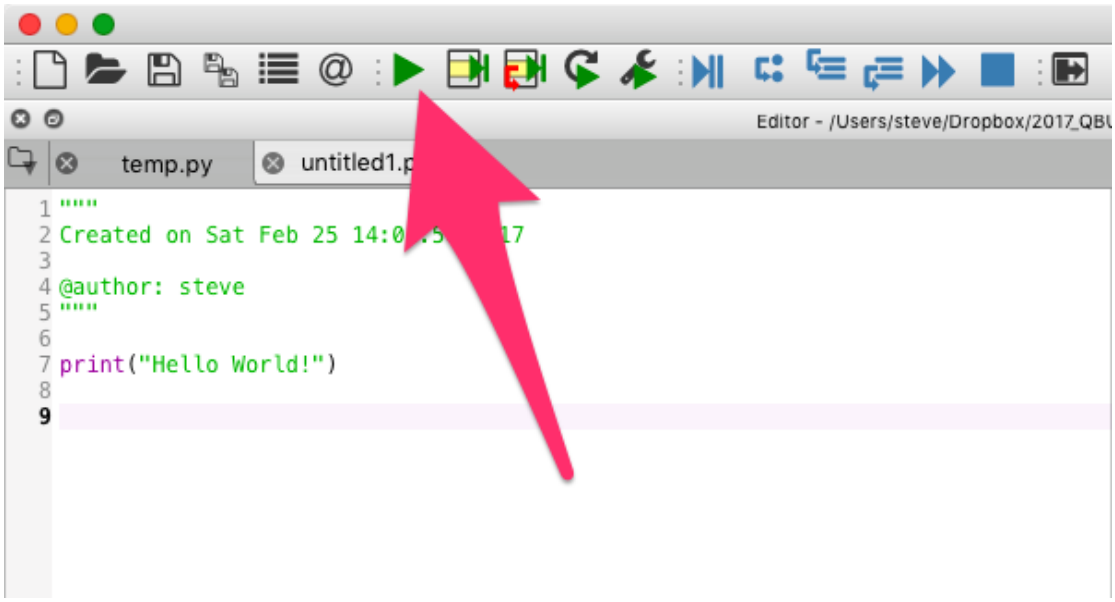
So far you've used the interactive console. The console is great for quick testing and checking variables but for more complex tasks it would be better to store your code so that you can use or run it later.

Let's create a Python script. A Python script is just a text file that contains Python code, which Python will read and execute line by line.

In Spyder go to File > New File. This will create a new empty Python script in the Script Editor (left pane of Spyder).

In the file add some Python code at the end. For example the Hello World print function.

Then click the green Run (play icon). The output will be displayed on the console. From now on we will work with Python Scripts.



## 5. Lists

Lists are a way to store an ordered collection of variables. Let's create a list and store it in the variable my_list. Try the following by adding to your Python Script then pressing Run

```
my_list = [1, 2, 4, 8, 16]
print(my_list)
```

List items can be accessed by their index. Python uses zero-indexing which means the first item starts at 0.

Try this example, print the value of the first item:

```
print(my_list[0])
```

Try this, to add items to the list you can use the append function

```
my_list.append(32)
```

To create an empty list you can use the list constructor function

```
my_empty_list = list()
```

To find out how many items are in the list use the len function

```
print(len(my_list))
```

## 6. Conditionals

So far when we press Run the code has executed each line of code in turn. But what about if we want to execute a different piece of code based on the current state or value of a variable?

This is where the conditional statements if, elif and else come into play. These statements allow us to branch the execution path.

Try the following example

```
input_number = int(input("Enter a number:"))

if input_number > 3:
    print("number is greater than 3")
elif input_number == 3:
    print("number is equal to 3")
else:
    print("number is less than 3")
```

**What's happening here?**

Tip: The input function waits to the user to enter a string that ends in a return

or new line character. The int function converts the input string to an integer.

## 7. Loops

A fundamental component of computer programming is iteration. In other words the ability to do repeated tasks.

In Python we do this using loops. There are two types of loops:

- For loop
- While loop

Let's start with an example of the For loop by iterating over our list of numbers that we created earlier and printing their values

```
for number in my_list:
    print(number)
```

You can also use list indexing to do the same thing. Try this

```
for i in range(len(my_list)):
    print(my_list[i])
```

While loops are a little more complicated. They continue iterating until a conditional statement becomes false. Try the following

```
count = 0
while (count < len(my_list)):
    print(my_list[count])
    count = count + 1
```

Tip: Be careful with loops (particularly while loops) as you may accidentally create an infinite loop which Python can't escape from. You will need to quit Spyder to stop an infinite loop.

## 8. Functions

Functions are a convenient way to divide your code into building blocks, allowing us to order our code, make it more readable, reuse it and save some time.

For example if you want to encapsulate a piece of repeated functionality we can define it once and reuse it multiple times.

Try this example

```
def welcome_message(first_name, last_name):
    print("Hello, {0} {1}! Welcome to Python.".format(first_name, last_name))

welcome_message("Stephen", "Tierney")
```

The function "welcome_message" has a two parameter's which we called "first_name" and "last_name". Inside the function Python will replace each

occurrence of the parameters with the values that we specified.

TIPS:

- def keyword means “define”
- The . (dot) syntax calls a function belonging to that object. In this example we are calling the “format” function belonging to the string object.
- The format function replaces occurrences of numbered curly braces with corresponding parameter values. In this case we insert the first and last name into the string.