# Unit 3: Text-Based Programming

Lesson 3

# Change the Output

Two main characteristics of a computer are that it can handle input and output. You have already learned how to create an output using print to put a message on the screen.

In this activity you will:

- continue to explore the use of print to output text to the screen
- use variables to store information in your program

Displaying “hello world” on the screen isn’t a very typical way to introduce yourself, so you should change the program so it says “Hi” rather than “hello world” when it starts.

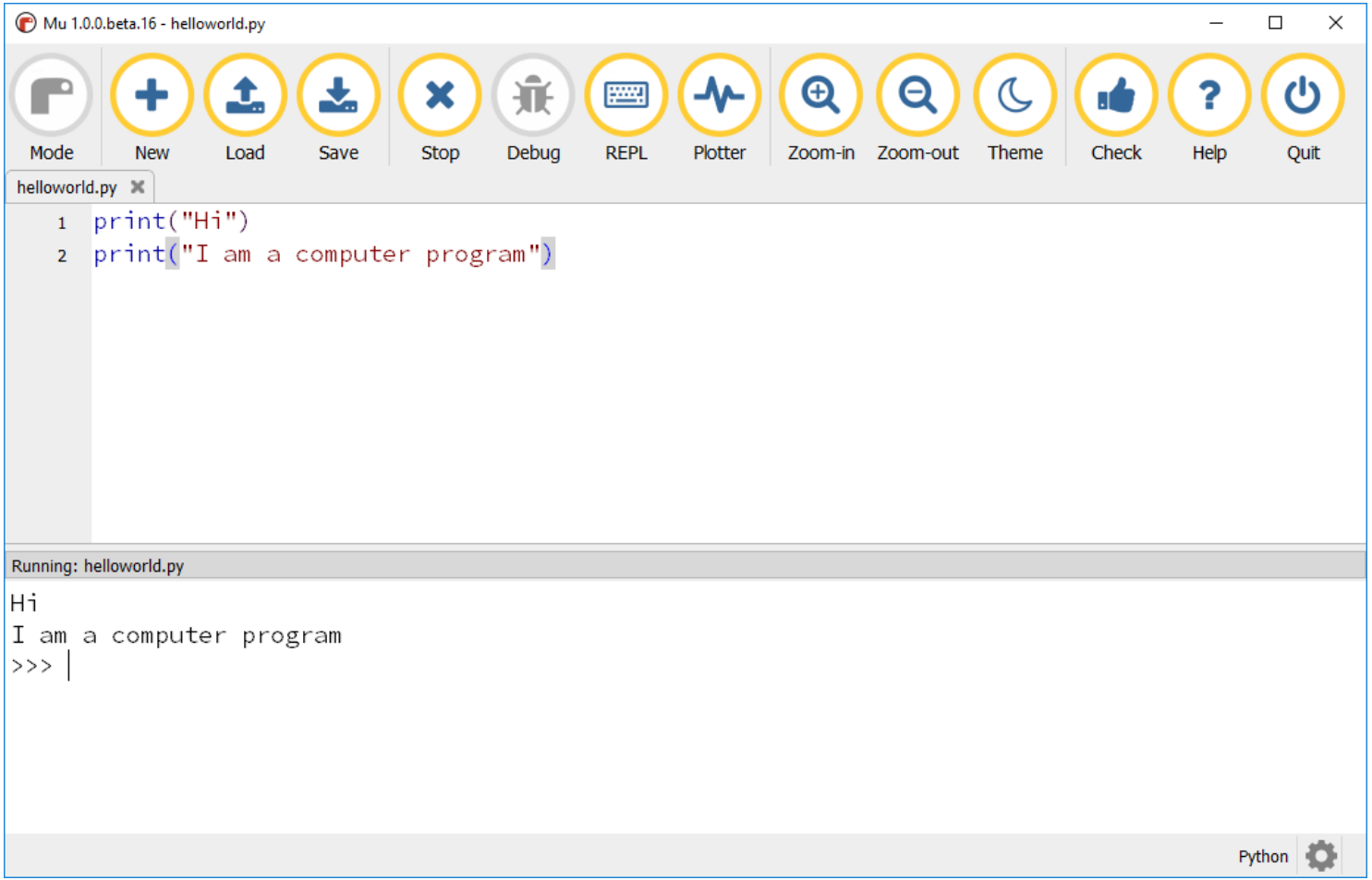
- First, open the “Hello World” program from the last lesson.
- Your program only has one instruction at the moment: `print("hello world")`. Whatever is in between the speech marks ("") is what is displayed (printed) to the screen.
- Change the `print("hello world")` instruction in your program to display “Hi”:
  - `print("Hi")`

# Running the Program

- Run the program by clicking Run.
  - The message “Hi” should now be displayed in the REPL.
  - To add more messages, you can use additional print instructions on the lines below saying other things.
- Add a second message to your program:

```
print("Hi")
print("I am a computer program")
```

- Run the program to see your second message.
  - **Tip**: - you may need to stop your program before you can run it again, by clicking the Stop icon.


Mu 1.0.0.beta.16 - helloworld.py
Mode
New
Load
Save
Stop
Debug
REPL
Plotter
Zoom-in
Zoom-out
Theme
Check
Help
Quit
helloworld.py
1 print("Hi")
2 print("I am a computer program")
Running: helloworld.py
Hi
I am a computer program
>>>
Python

Note how the messages are displayed in this order:

```
Hi
I am a computer program
```

This represents an important concept in computer programming: **sequencing.** The messages are guaranteed to always appear in this order, as the computer runs each of these instructions in the order given and one at a time.

# Variables

**Variables** are a way of storing pieces of data in your program. When you create a variable, an area of the computer's memory is reserved for you and the data is stored in it. This area of memory is now controlled by you: you can retrieve data from it, change the data, or get rid of it all together.

To create a variable in Python you give it a name and make it equal to a value. This is known as **assignment**. For example:

```
my_variable = "some useful data"
```

Next you will change your program to store your name in a variable and then display it.

- Add the following code to the bottom of your program to create a new variable called my_name to hold your name.

  ```
  my_name = "Martin"
  ```

  You should put your name (or any other name you like!) in between the speech marks: `"Martin"`. `Putting the text inside a set of quotes lets python know that the variable is a string.`

When your computer runs this instruction, a variable called `my_name` will be created and the text of your name will be stored.

Once a variable has been created, it can then be used in your program.

- Use print to output your name to the screen by adding this code to the bottom of your program.
  ```
  print(my_name)
  ```

Notice how when using print you put the **name of the variable** between the () instead of text. The print instruction will retrieve the value from the my_name variable and display it.

# Running the Program

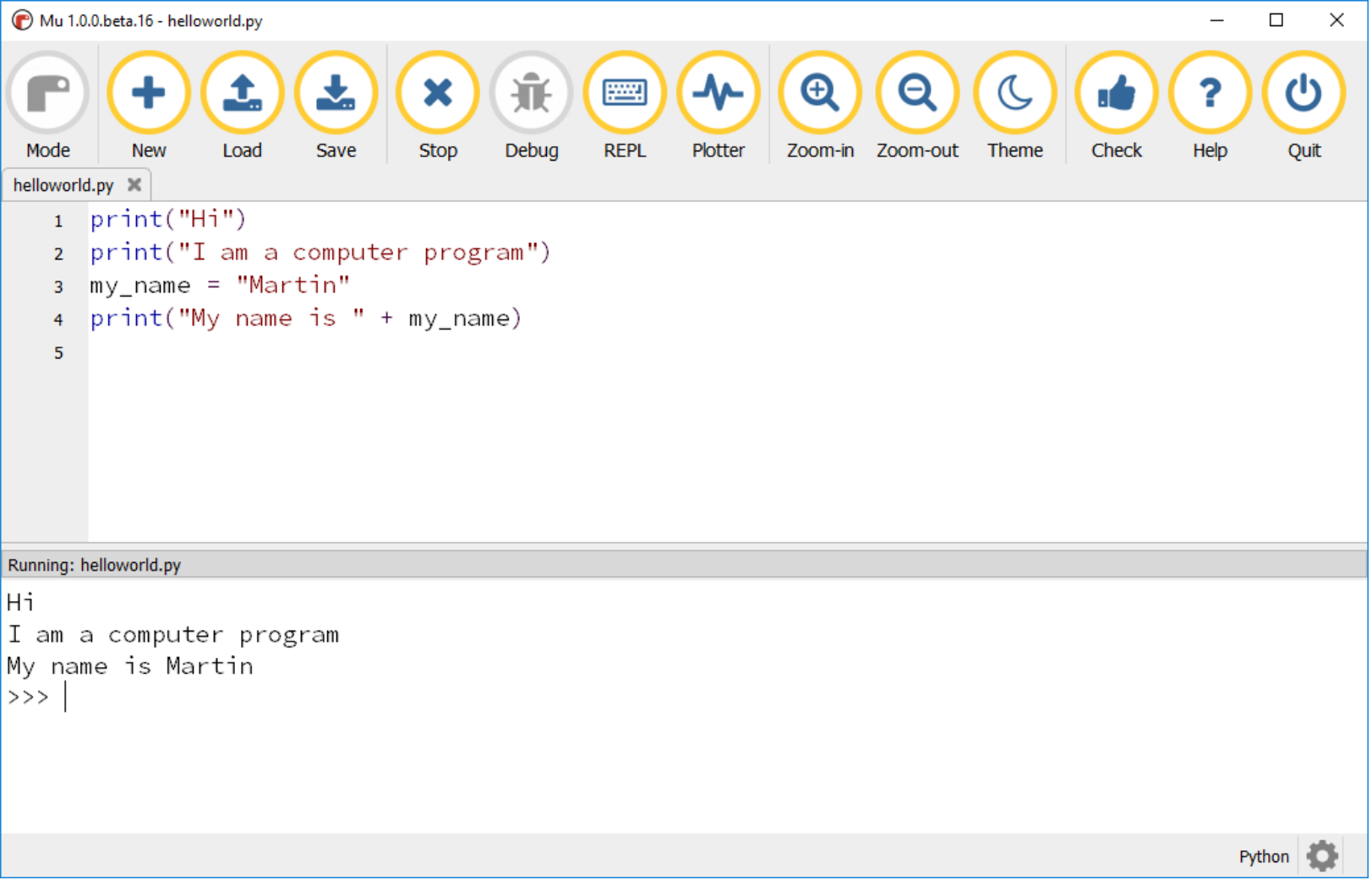
- Run your program to see your name appear under the first 2 print statements.
  **Tip:** a common problem is to put the variable name inside speech marks, e.g. `print("my_name")`. ***What will this do? If you are not sure, try it!***

You can improve the message by putting the text “My name is “ in front of your name.

- Modify the code to add the text "My name is " before the my_name variable:
  `print("My name is " + my_name)`

Note the use of the + sign to add the first piece of text to the text in the variable. Adding pieces of text together like this is known as **concatenation.**

- Run your program.




You need to show this program to your teacher. It will count towards your mark.

# Creating Output Statements

There are many ways to create output statements in python. These statements can be used to with the print function and can also be used to write information to a file.

- Concatenation,
  - This only works for strings(text). We will encounter this later when we use other variable types.
  - You have to insert spaces to make it look correct

| Code | Output |
| --- | --- |
| `my_name = "Martin"`<br>`print("My name is " + my_name + ".")` | `My name is Martin.`<br>`>>>` |

# Creating Output Statements

- Use multiple arguments(information passed to a function)
  - *The arguments are separated by a comma*
  - *The print function automatically inserts a space between each argument*

| Code | Output |
| --- | --- |
| `my_name = "Martin"`<br><br>`print("My name is", my_name, ".")` | `My name is Martin .`<br>`>>>` |

# Creating Output Statements

- F strings
  - *This is the preferred method*
  - *A f goes before the opening quote and variables go in curly brackets { }*
  - *The statement will be displayed with the same format that you enter. So spaces are easy to deal with.*

| Code | Output |
| --- | --- |
| `my_name = "Martin"`<br><br>`print(f"My name is {my_name}.")` | `My name is Martin.`<br>`>>>` |

# Portfolio 1

Create a program where you can enter that has a **variable** named age, where you will store your age in years. Put your age in quotes so it will be stored as a string.

You need to use the **three different methods** to create  shown above to display the same statement three times.

Sample output:

```
I am 37 years old.
I am 37 years old.
I am 37 years old.
>>>
```

Save your program in the Portfolio folder in your Unit 3 folder.

Name the files as shown below

robert_brake_u3_port_1.py

- Don’t forget to back up your Unit 3 folder in your Google Drive.

- The portfolios will be submitted towards the end of Unit 3. You can upload your files into Google Classroom, but don’t submit it until I tell you that all of the portfolios have been assigned.