Unit 3: Text-Based Programming

Lesson 5

Creating Your Bot

You are going to create a project which you will work on throughout the unit. It's going to be a bot, which will help you complete certain tasks.

Creating a New Program

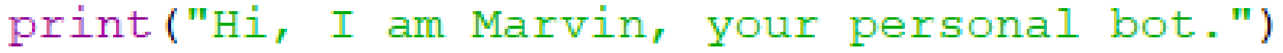
Click **New** to create a new program.

Click **Save** to save your program in your **Activities** folder with the filename **adding_bot**

Introductions

When your program starts, your bot should introduce itself by using print to put a message on the screen.

- Add the code to your program to print a welcome message.



```
print("Hi, I am Marvin, your personal bot.")
```

You can use whatever message you want. Perhaps you want to display a few messages, such as “Let’s get started”.

Things to do(try to do it without looking back at your hello_world program)

1. Use input to ask the user’s name and store it in a variable called users_name.
2. Use print and the users_name variable to display a “Welcome [name]” message.

Calculation

- **You are now going to program your bot to help add numbers together. It will ask the user for two numbers, add the numbers together, and display the result.**

  In order to do this, your program will need to create variables for the following pieces of data:

  1. The first number the user inputs
  2. The second number
  3. The result of adding the first and second inputs

- Add code to ask for the two numbers and store them in variables, by adding the following code to the end of your program.

```
print("lets add some numbers")
num_1 = input("Number 1:  ")
num_2 = input("Number 2:  ")
```

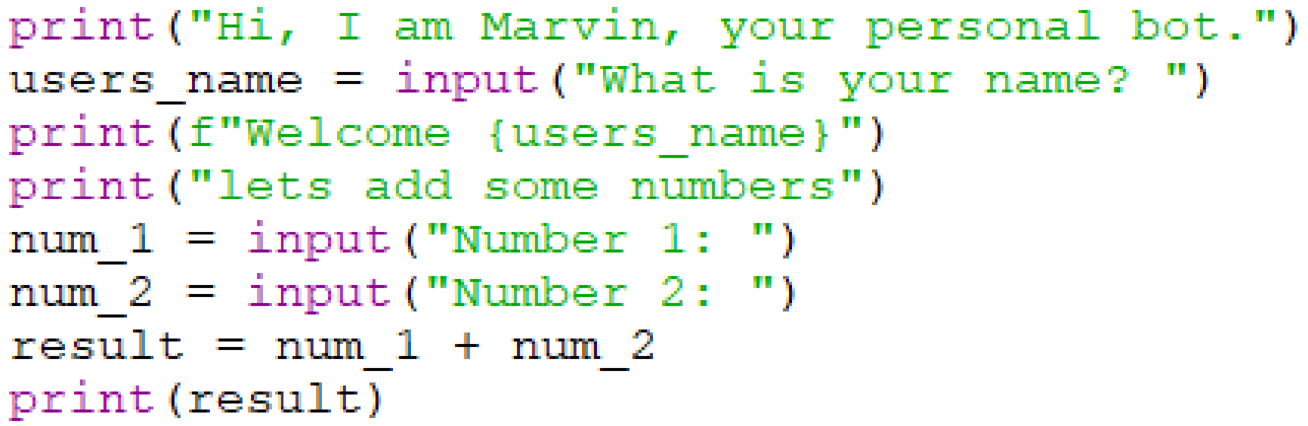
- Create a new variable to store the sum of the two numbers entered.

```
result = num_1 + num_2
```

- Print the variable result to the screen.

```
print(result)
```

Your complete code should look like this:



```python
print("Hi, I am Marvin, your personal bot.")
users_name = input("What is your name? ")
print(f"Welcome {users_name}")
print("lets add some numbers")
num_1 = input("Number 1: ")
num_2 = input("Number 2: ")
result = num_1 + num_2
print(result)
```

- Run your program, enter 2 numbers and see what the result is.

The result might not be what you were expecting. If you enter 1 and 2, you get the result 12 and not 3.

I bet you thought computers were good at math.

But remember they are good at doing what you tell them to, even if what you tell them is incorrect.

Instead of adding the two numbers, your program has **concatenated** the two inputs as if they were strings.

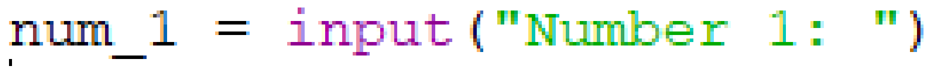
To resolve this problem you need to tell the computer how to interpret the data it’s been given. It has interpreted the input as a piece of text like "1", but we want it to interpret it as a number like 1.

Reflection

Before you move on to debugging, explain to yourself what each line of your code is doing and why the program doesn’t give you the expected result.

Text and Numbers

In the previous step you didn’t get the result you were expecting because input stores a string in the variable num_1, instead of a number.



```
num_1 = input("Number 1:  ")
```

Variables in Python not only store data; they also have a **data type** which says what sort of data the variable holds.

Variables that store **text** data have the **data type string**, whereas variables that hold **whole numbers** have the **data type integer.**

When the user types some input, it is stored as a string. In order to get Python to add the numbers together, you need to convert the variables `num_1` **and** `num_2` **from strings to integers**. Converting variables from one data type to another is known as **casting.**

To cast your string variables `num_1` and `num_2` to integer you can use the int function (the name is short for ‘integer’).

- After you have got the two numbers using input with these lines:

```
num_1 = input("Number 1:  ")
num_2 = input("Number 2:  ")
```

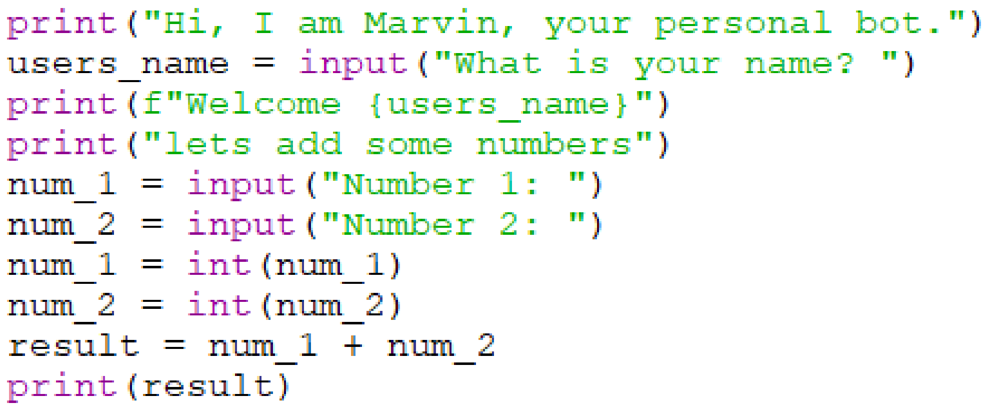
- add the following code convert num_1 and num_2 into integers.

```
num_1 = int(num_1)
num_2 = int(num_2)
```

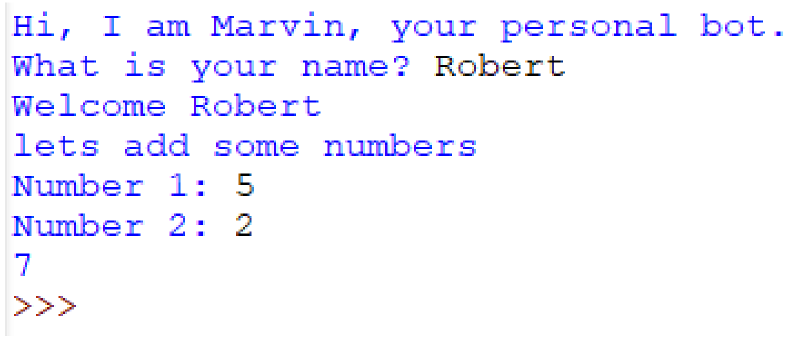
- NOTE* you could also create two new variables with different names, but it wouldn’t benefit you, it would actually increase the number of variables that you have to remember in your program.

Your complete code should now look like this:

Code **Output**



```python
print("Hi, I am Marvin, your personal bot.")
users_name = input("What is your name? ")
print(f"Welcome {users_name}")
print("lets add some numbers")
num_1 = input("Number 1: ")
num_2 = input("Number 2: ")
num_1 = int(num_1)
num_2 = int(num_2)
result = num_1 + num_2
print(result)
```

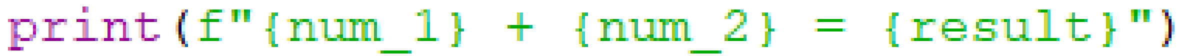


```
Hi, I am Marvin, your personal bot.
What is your name? Robert
Welcome Robert
lets add some numbers
Number 1: 5
Number 2: 2
7
>>>
```

Run your program. You should now get the correct result.

At the moment the output of the result is very basic. Using f-strings, we can improve the result so it looks like this: 1 + 2 = 3.

- replace the instruction print(result) with:

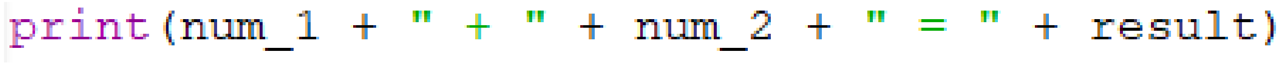


```
print(f"{num_1} + {num_2} = {result}")
```

  Then run your program again

- Try to create a print statement that will output the same message as above but use concatenation(the one with the plus signs). This should show you why f-strings are better.

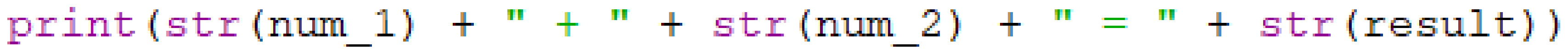
- You may have tried a print statement similar to what is shown below



```
print(num_1 + " + " + num_2 + " = " + result)
```

  But if you tried it you would get a TypeError saying that you can’t use + with an int and a str.

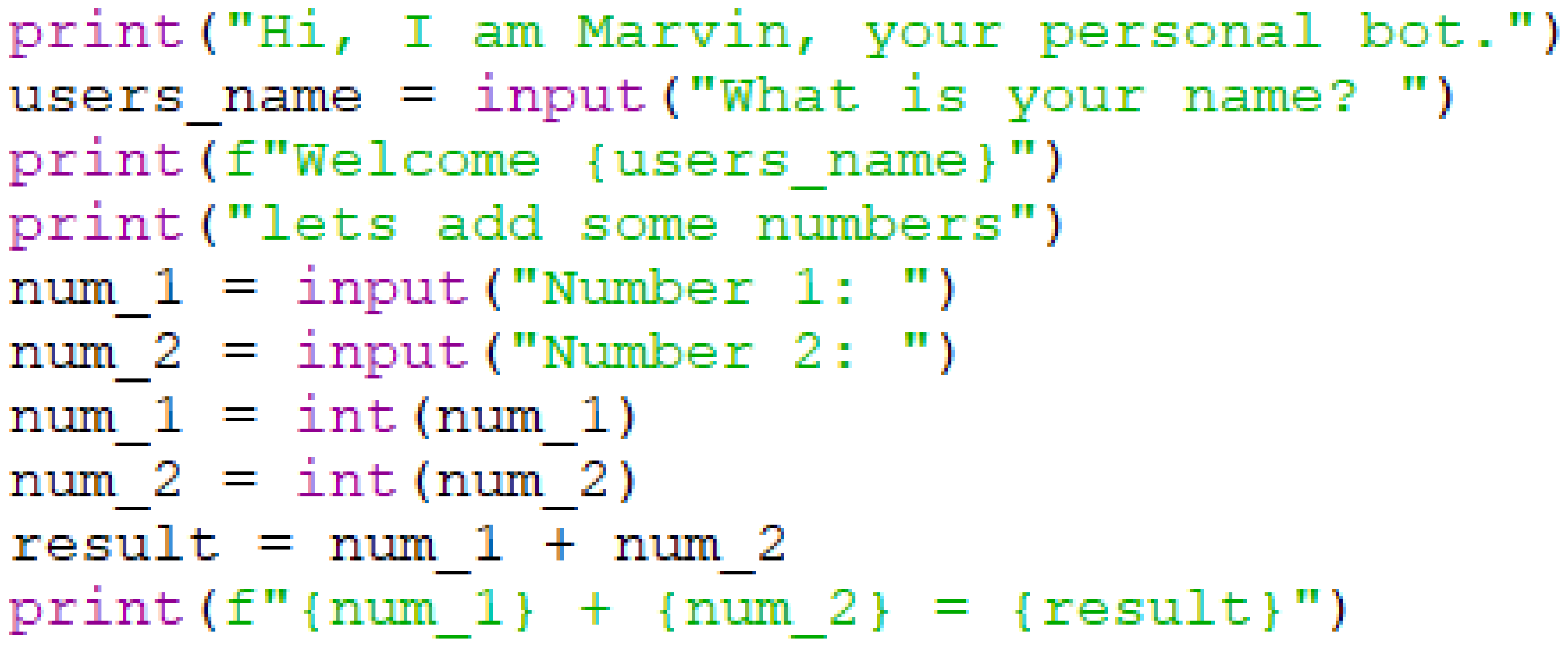
- The above print statement is messy enough, but to get it to work you would have to convert your variables back into strings, like this print statement



```
print(str(num_1) + " + " + str(num_2) + " = " + str(result))
```

- I don’t think that anyone would say that concatenation is better.

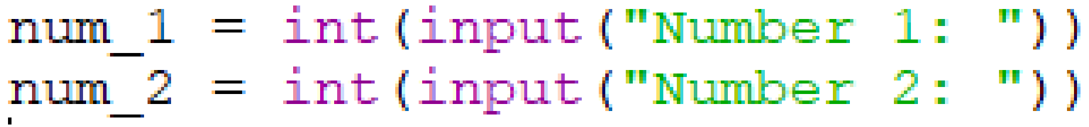
Your program should look like this now



```python
print("Hi, I am Marvin, your personal bot.")
users_name = input("What is your name? ")
print(f"Welcome {users_name}")
print("lets add some numbers")
num_1 = input("Number 1: ")
num_2 = input("Number 2: ")
num_1 = int(num_1)
num_2 = int(num_2)
result = num_1 + num_2
print(f"{num_1} + {num_2} = {result}")
```

Shortening your file

- When you create a program you should aim to make it efficient, aka shorter
- One thing we can do in our program is to combine the input and conversions together, replace lines 5 – 8 with the following.



```
num_1 = int(input("Number 1:  "))
num_2 = int(input("Number 2:  "))
```

- You should convert your input to the proper type of variable whenever you can.

Other Data Types

Other commonly used variable types in Python:

Data type	Description	Example
boolean	Used for storing values which can only ever be true or false	True or False
float	Similar to integer but for non whole (decimal) numbers	1.234
list	Used to store many items of data in an order e.g. a list of people’s names	["Martin", "Rik", "Hitesh"]

Portfolios

Create two new programs and save them in your **portfolio** folder that will:

1. adding three numbers together
   - save it as your_name_u3_port_2
2. calculating the area of a rectangle
   - save it as your_name_u3_port_3