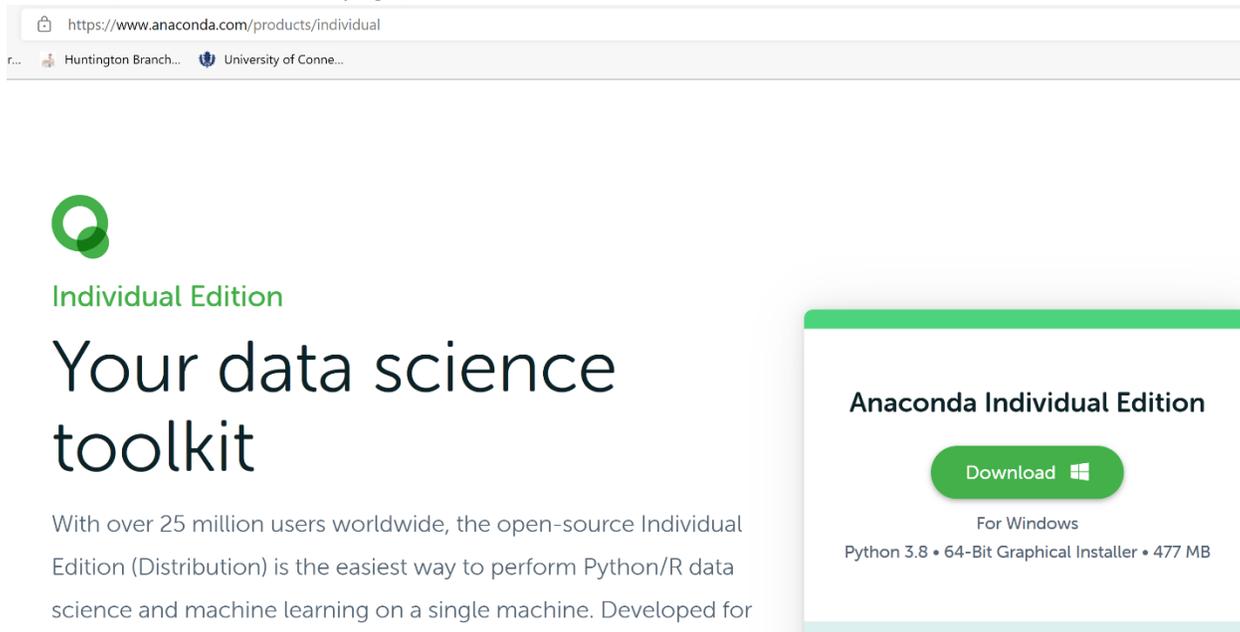
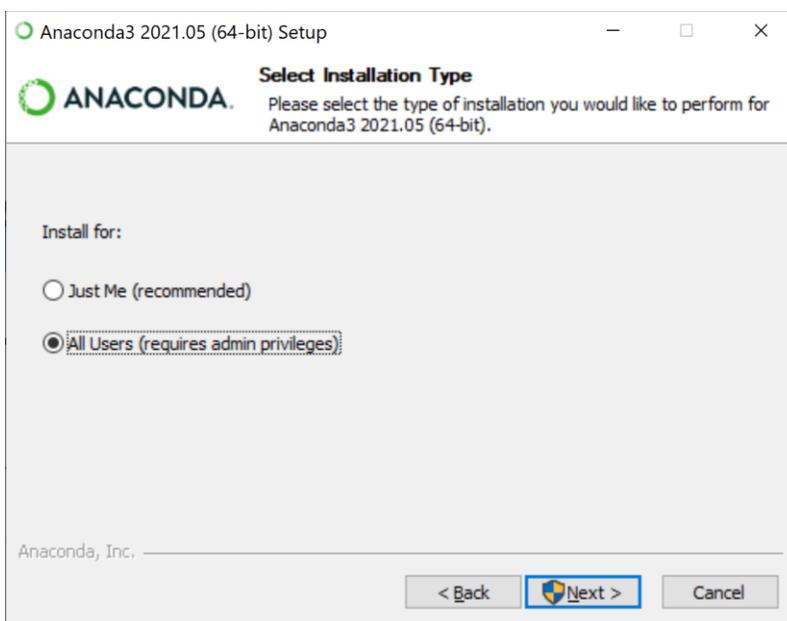


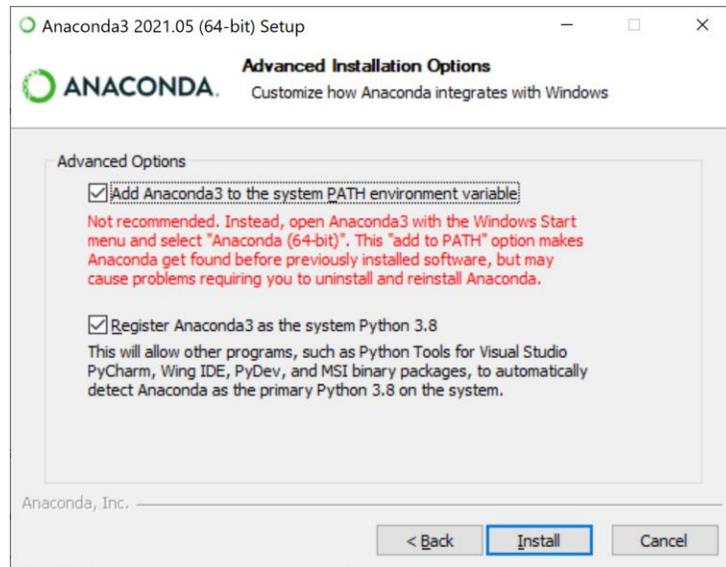
Getting Started - Installing Python for Use with Visual Studio

1. Install Visual Studio Community Edition with all Workloads from the Microsoft site:
[Visual Studio 2019 Community Edition – Download Latest Free Version \(microsoft.com\)](https://visualstudio.microsoft.com/downloads/#community-edition)
2. Install Anaconda Individual Edition (see below) IMPORTANT: both check boxes should be checked (see below on next page):



The screenshot shows a web browser window with the URL <https://www.anaconda.com/products/individual>. The page features the Anaconda logo, the text "Individual Edition", and the headline "Your data science toolkit". Below the headline, it states: "With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for". On the right side, there is a green button labeled "Download" with a Windows logo, and below it, the text "For Windows" and "Python 3.8 • 64-Bit Graphical Installer • 477 MB".

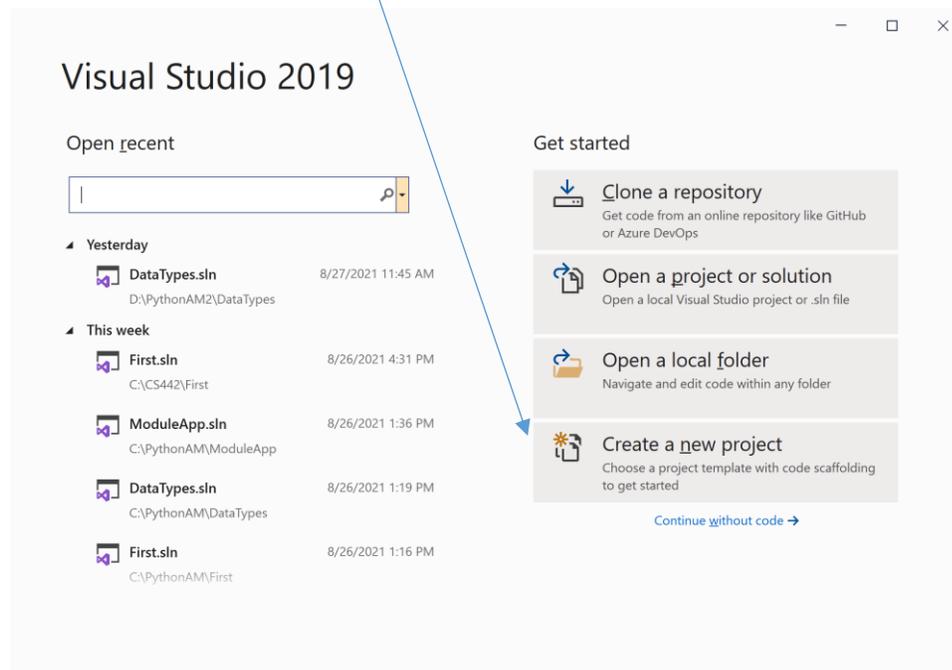




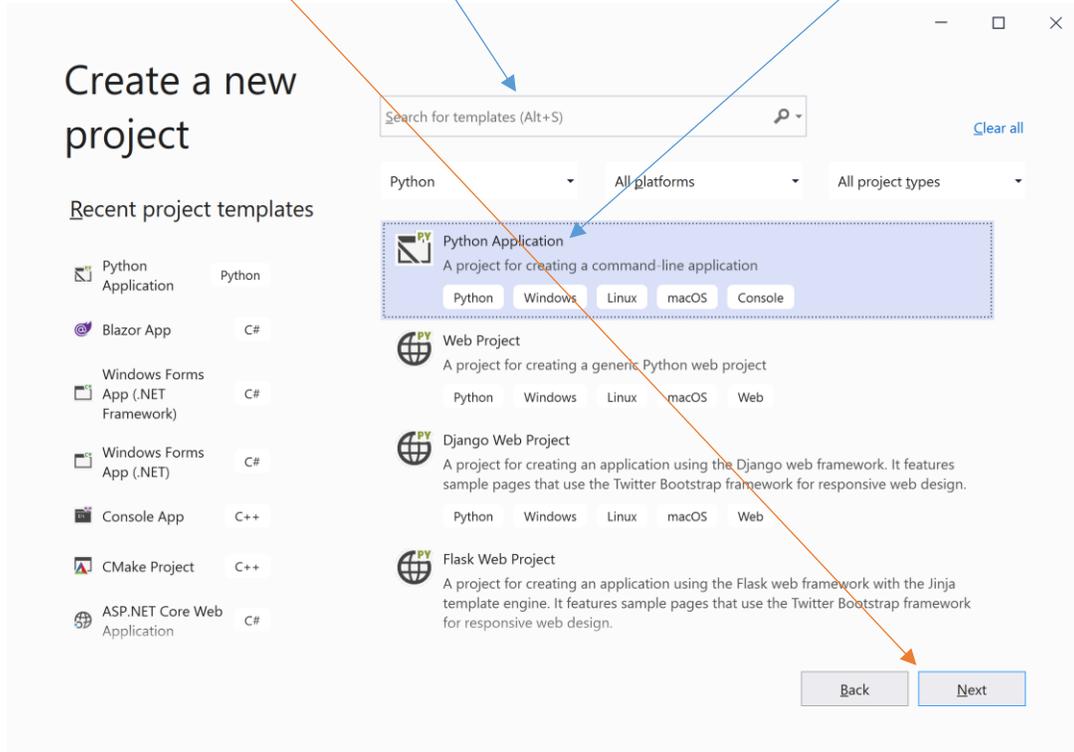
3. Launch command prompt, issue the following commands one at a time.
4. `conda create -n tensorflow2x pip python=3.8`
5. `activate tensorflow2x`
6. `pip install --upgrade tensorflow`

Creating your First Python Program:

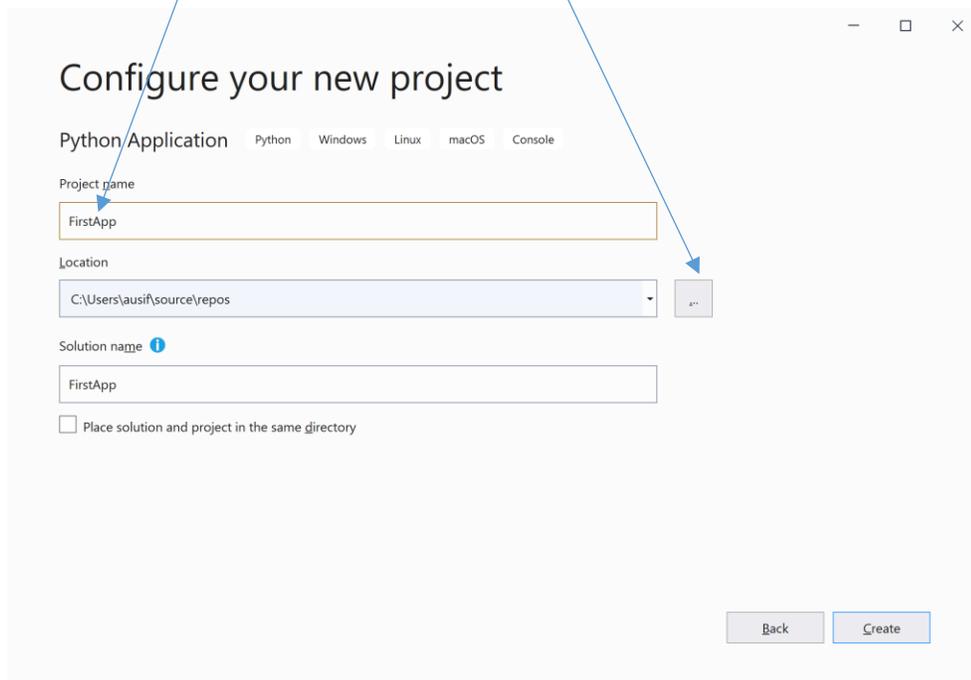
Launch Visual Studio. Choose “Create a New Project” from the following dialog



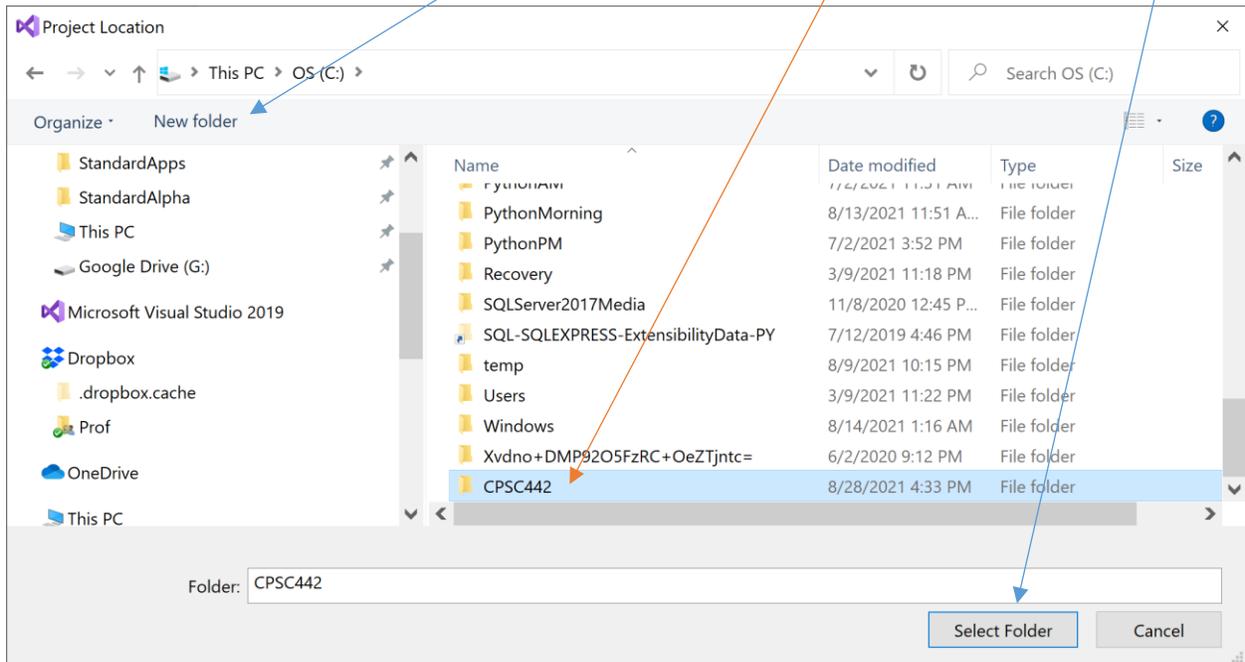
On the next dialog, choose Python as the language, and click on “Python Application” as the project type, then click next.



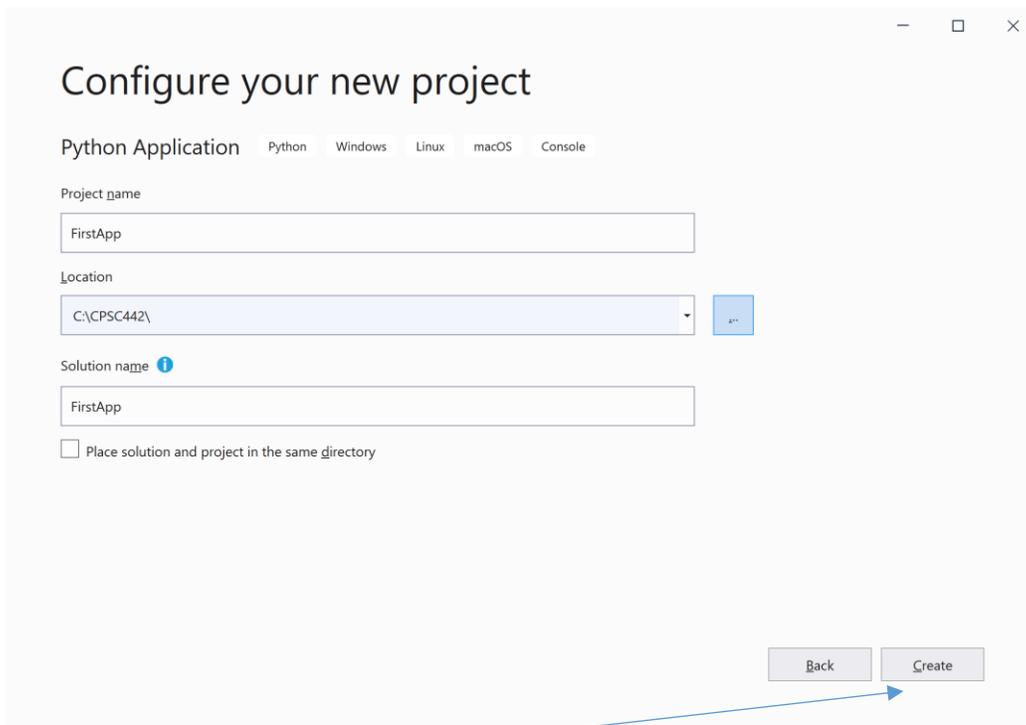
Name the project “FirstApp”, and click on the three dotted button to specify the folder where the project will be created.



Move to the C drive, then click on New folder, and name the folder CPSC442, and then click on select folder:

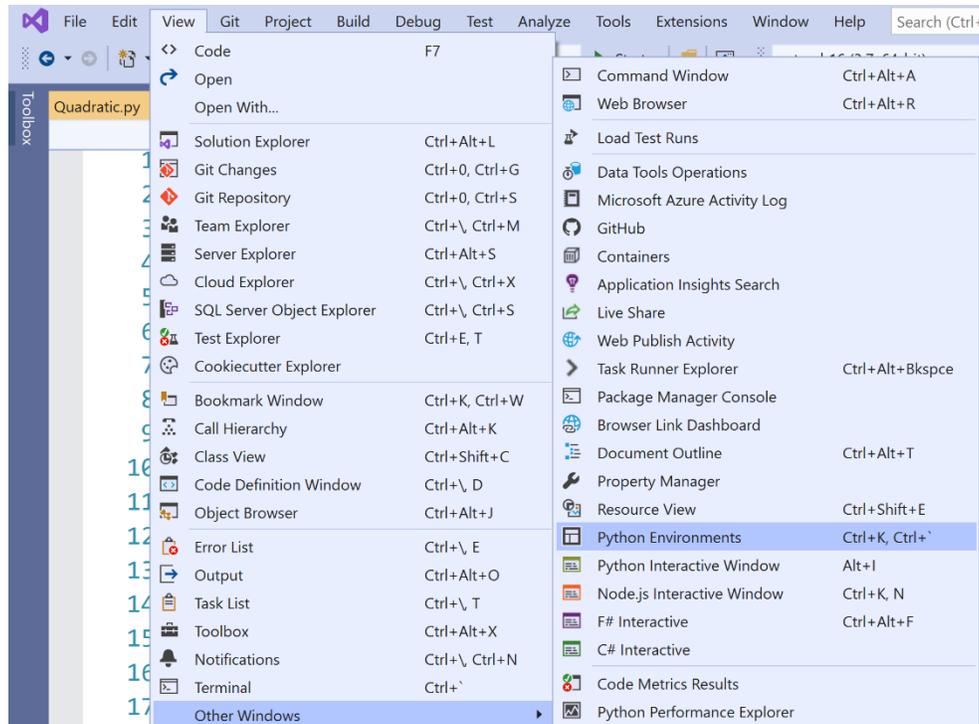


Your Visual Studio dialog will appear as:

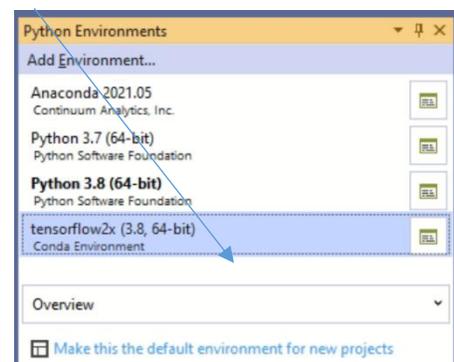


Click "Create" to create the Visual Studio Python project.

Setting the Python Environment: We had created the Python environment *tensorflow2x* earlier. Now you will set it as the default for your Python project. A Python environment contains all the necessary libraries as well as the Python runtime for running programs. From the view menu, choose “Other Windows”, then Python Environments as shown below.

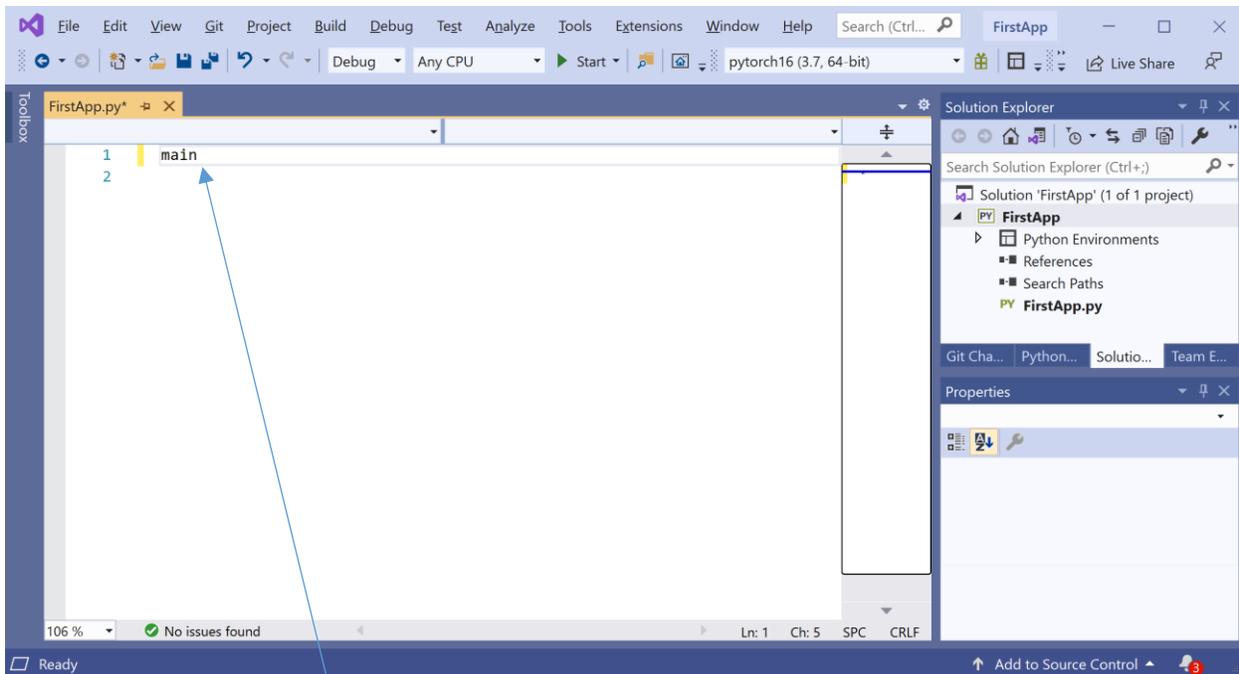


Then select *tensorflow2x*, and click on, make this the default environment for new projects.

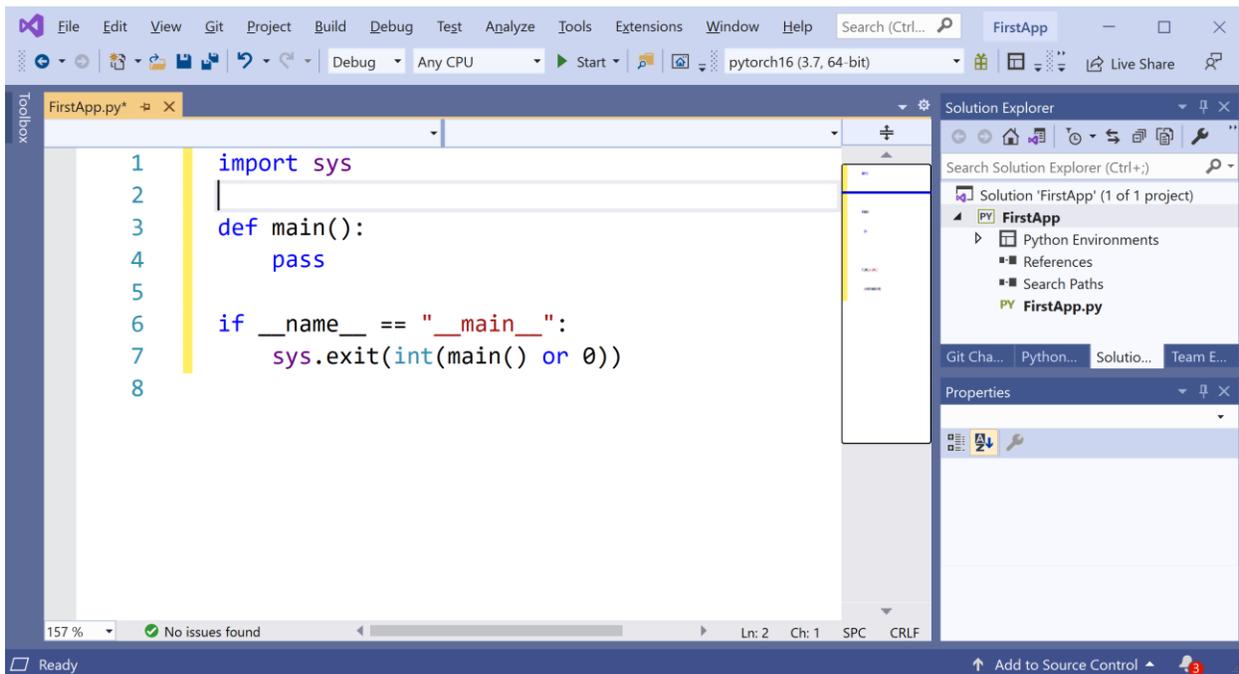


Then click on the “Solution Explorer” tab (next to the Python Environments tab) to view the files in your project.

Even though, in Python, we can start writing code in a file without a “main” entry point, it is recommended to have a main function as the starting point for execution of the program.



Type main followed by the tab key twice (i.e., hit the tab key, wait a second, and then hit the tab key again). Now your code will appear as:

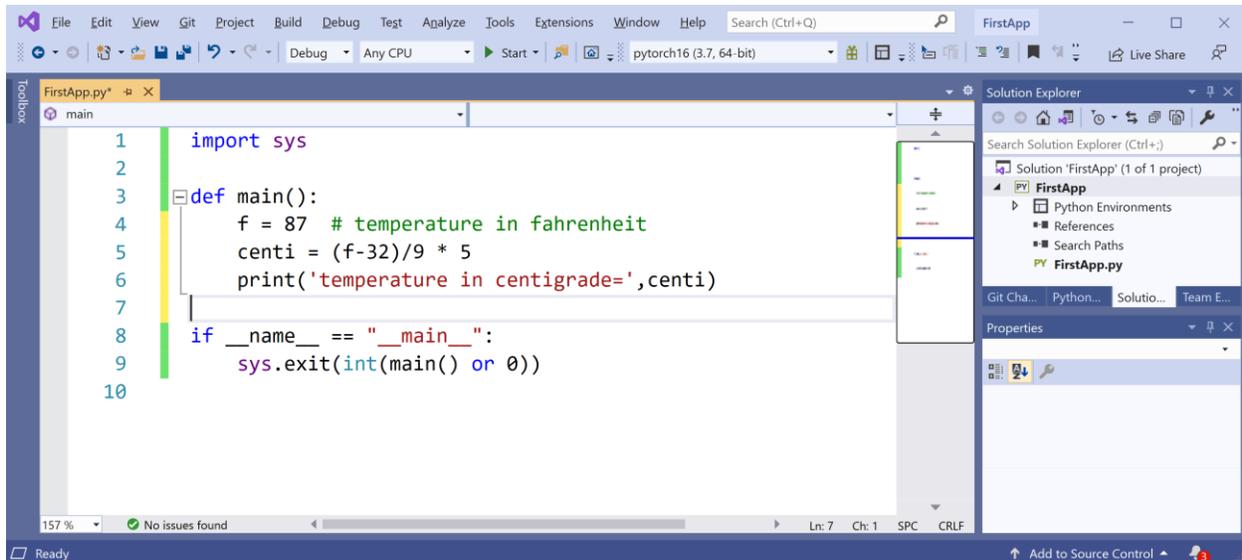


Suppose we wanted to write code to convert Fahrenheit temperature to centigrade. We know the formula for converting the Fahrenheit to centigrade is:

$$c/5 = (f-32)/9$$

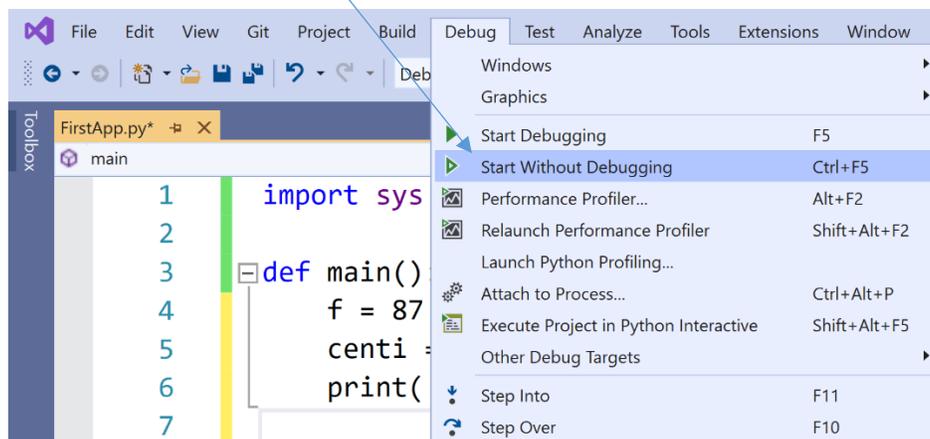
NOTE: Python is case sensitive, so if a variable is initially declared as lower case, it should be used with the same case later.

Our Python code in FirstApp will appear as:

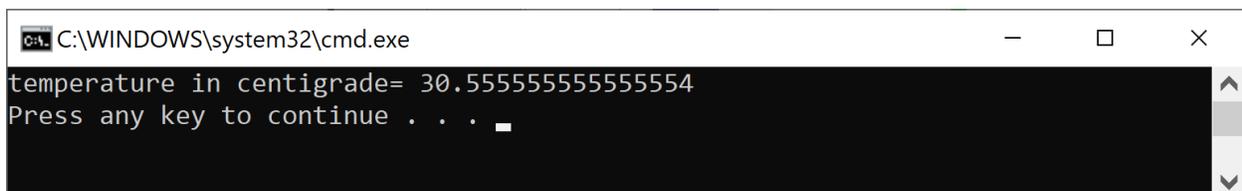


```
1 import sys
2
3 def main():
4     f = 87 # temperature in fahrenheit
5     centi = (f-32)/9 * 5
6     print('temperature in centigrade=',centi)
7
8 if __name__ == "__main__":
9     sys.exit(int(main() or 0))
10
```

Run the program by choosing Debug->Start Without Debugging.



The output will appear as:



```
C:\WINDOWS\system32\cmd.exe
temperature in centigrade= 30.555555555555554
Press any key to continue . . .
```

Python Data Types:

A computer internally stores different types of data differently e.g., integers such as 23, 75, 89 are stored differently from decimal numbers such as 3.5, 82.76. The more popular data types in Python are int, float and string (int for integers, float for representing decimal numbers, and strings for representing characters, words and sentences).

Python is a dynamic language, which means the type of a variable can change as the program executes. When data is assigned to a variable, Python decides how that data will be stored internally in the memory of the computer. For example, type and run the following code.

```

1  import sys
2
3  def main():
4      f = 87 # temperature in fahrenheit
5      centi = (f-32)/9 * 5
6      print('temperature in centigrade=',centi)
7
8      a = 25
9      print('data type of a=',type(a))
10
11     b = 23.85
12     print('type of b=',type(b))
13
14     c = 'hello'
15     print('data type of c=',type(c))
16
17 if __name__ == "__main__":
18     sys.exit(int(main() or 0))

```

Run the program by choosing Debug->Start without Debugging. Your output will appear as:

```

C:\WINDOWS\system32\cmd.exe
temperature in centigrade= 30.555555555555554
data type of a= <class 'int'>
type of b= <class 'float'>
data type of c= <class 'str'>
Press any key to continue . . .

```

Writing Functions in Python:

If a computation needs to be repeated many times, then we define the computation needed inside a block of code referred to as a function. A function has a name and after its name, we indicate what input(s) it will receive to perform the computation. At the end of the code in a function, we return the result being produced by the function.

For example, the function to do the temperature conversion from Fahrenheit to centigrade can be written as:

```

def fahrenheit_to_centigrade(f):
    centi = (f-32)/9 * 5
    return centi

```

input to function

result being returned from function

Note that Python Python uses indentation of code to indicate it is part of a function. For example centi, and return lines in above block of code are indented by 4 spaces (or a tab).

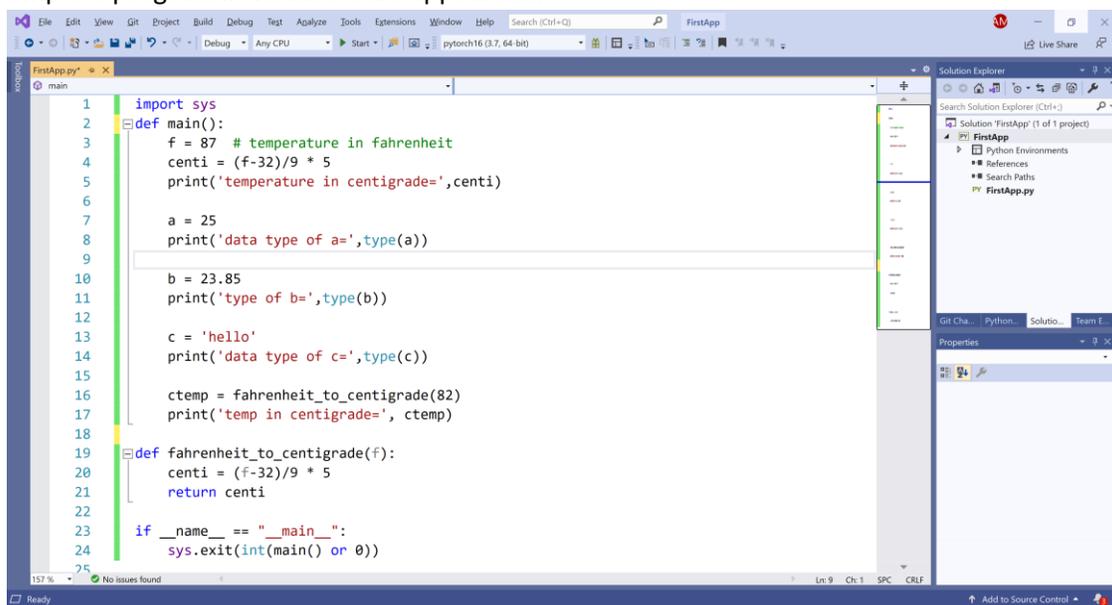
A function does not do anything until it is called. If we call the above function as:

```

ctemp = fahrenheit_to_centigrade(82)
print('temp in centigrade=', ctemp)

```

The complete program in Visual Studio appears as:



```

1  import sys
2  def main():
3      f = 87 # temperature in fahrenheit
4      centi = (f-32)/9 * 5
5      print('temperature in centigrade=',centi)
6
7      a = 25
8      print('data type of a=',type(a))
9
10     b = 23.85
11     print('type of b=',type(b))
12
13     c = 'hello'
14     print('data type of c=',type(c))
15
16     ctemp = fahrenheit_to_centigrade(82)
17     print('temp in centigrade=', ctemp)
18
19 def fahrenheit_to_centigrade(f):
20     centi = (f-32)/9 * 5
21     return centi
22
23 if __name__ == "__main__":
24     sys.exit(int(main() or 0))
25

```

The output will appear as:



```

C:\WINDOWS\system32\cmd.exe
temperature in centigrade= 30.555555555555554
data type of a= <class 'int'>
type of b= <class 'float'>
data type of c= <class 'str'>
temp in centigrade= 27.777777777777778
Press any key to continue . . .

```

Reading Data from Keyboard:

Sometimes, we want the user to input the data needed by the program. For example, we may want the user to type in the Fahrenheit temperature on the keyboard. For reading data typed via the keyboard, Python provides an easy to use built-in function called 'input'.

For example, the following code asks the user to type in the temperature:

```
ftemp = input('please type in the temperature in fahrenheit:')
```

if the user types in 85 and hits enter, then the above program will read the 85 and store it in the ftemp variable. However, note that any data read from the keyboard (or file) is read as a string. If we wanted to apply the temperature conversion formula to ftemp, it will have to be converted to float first. Python provides data conversion functions such as int(), float(), str() to convert a given data to int or float, or string.

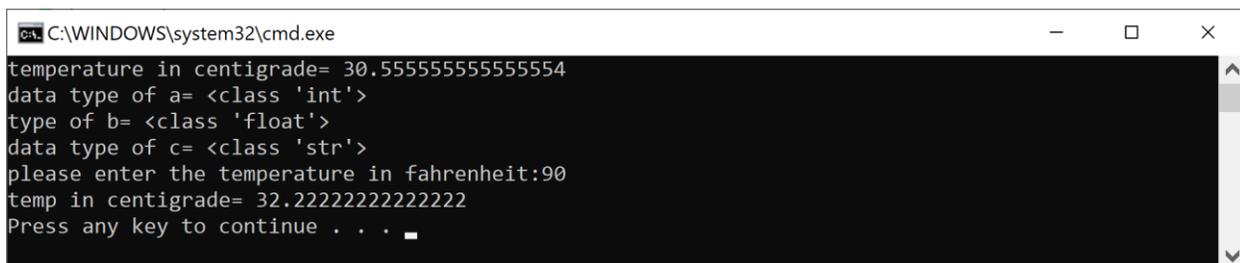
In the above example, to convert ftemp variable, from a string data type to float data type, we will do the following:

```
ftemp = float(ftemp)
```

The complete code that reads the fahrenheit temperature data from the keyboard and converts it to centigrade appears as:

```
16 ftemp = input('please enter the temperature in fahrenheit:')
17 ftemp = float(ftemp) # keyboard input is read as a string
18 ctemp = fahrenheit_to_centigrade(ftemp)
19 print('temp in centigrade=', ctemp)
20
21 def fahrenheit_to_centigrade(f):
22     centi = (f-32)/9 * 5
23     return centi
```

If you run it, type 90 and hit enter, the output will appear as:



```
C:\WINDOWS\system32\cmd.exe
temperature in centigrade= 30.555555555555554
data type of a= <class 'int'>
type of b= <class 'float'>
data type of c= <class 'str'>
please enter the temperature in fahrenheit:90
temp in centigrade= 32.222222222222222
Press any key to continue . . .
```

Example: Create a Python program to compute the roots of the quadratic equation. The value of a, b, c coefficients will be input by the user.

The formula for computing the roots of a quadratic equation $ax^2 + bx + c = 0$ is:

$$r1, r2 = (-b \pm \sqrt{b^2 - 4ac})/2a$$

For example, if we are given the equation $x^2 + 6x + 8 = 0$, r1 will be -4, and r2 will be -2.

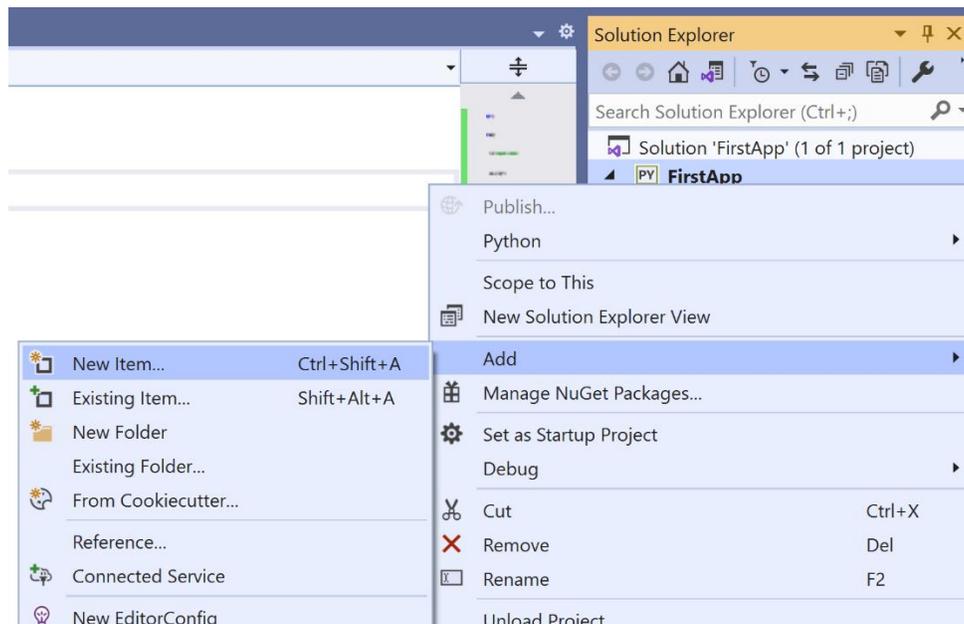
When we write the above formula in Python, it has to be broken in two statements as (Note: ** means raised power in Python, e.g., 5^{*3} is 5^3):

```
r1 = (-b + math.sqrt(b**2 - 4*a*c))/(2*a)
```

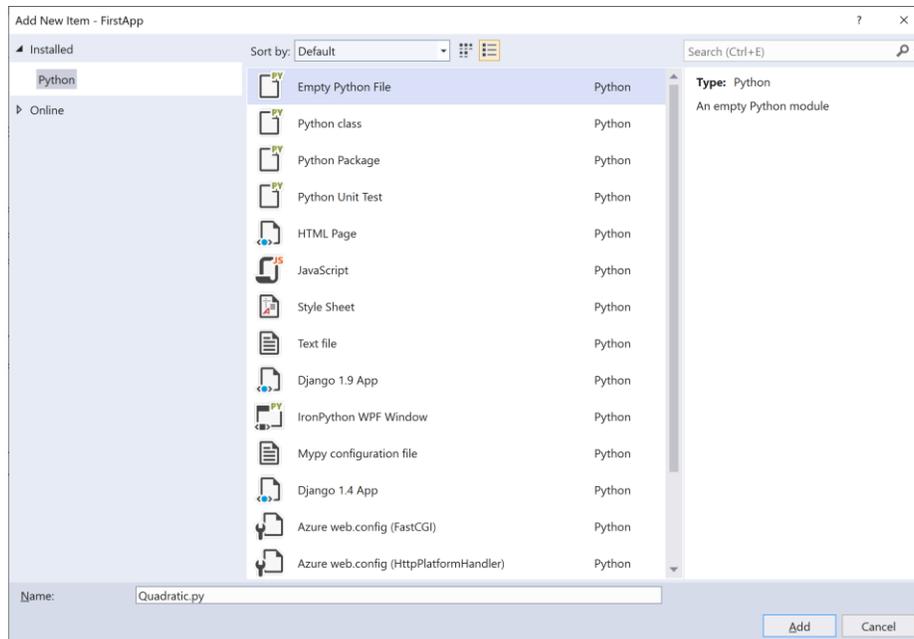
```
r2 = (-b - math.sqrt(b**2 - 4*a*c))/(2*a)
```

Python provides a math library where we have several built-in functions such as sqrt, sin, cos, etc..

Add a file to the project called Quadratic.py by right clicking on the project name “FirstApp”, and choosing “Add -> New Item” as:



Then select “Empty Python File” and choose Quadratic.py as:



Add a main function to the Quadratic.py by typing main followed by a tab and a tab. Then type the following code in Quadratic.py.

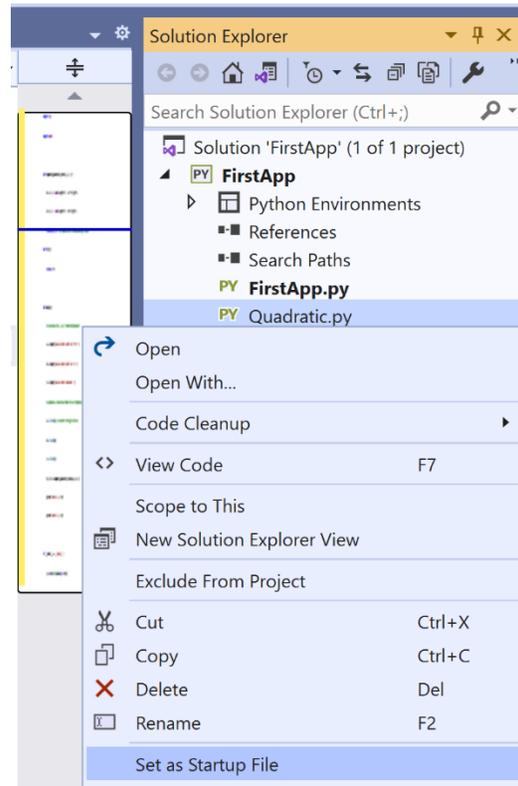
```

1  import sys
2  import math
3
4  def compute_quadratic_roots(a, b, c):
5      x1 = (-b + math.sqrt(b*b - 4 *a*c))/2*a
6      x2 = (-b - math.sqrt(b*b - 4*a*c))/2*a
7      return x1, x2 # a function can return multiple items
8  def f1(x):
9      return x*2
10
11 def main():
12     # read value of a, b, c from the keyboard
13     a = input('please enter coeff. of x^2: ')
14     b = input('please enter coeff. of x: ')
15     c = input('please enter constant: ')
16     # anytime we read data from file or keyboard, it is read as a string
17     a = float(a) # convert string to float
18     b = float(b)
19     c = float(c)
20     r1, r2 = compute_quadratic_roots(a, b, c)
21     print('root1 = ', r1)
22     print('root2 = ', r2)
23
24 if __name__ == "__main__":
25     sys.exit(int(main() or 0))

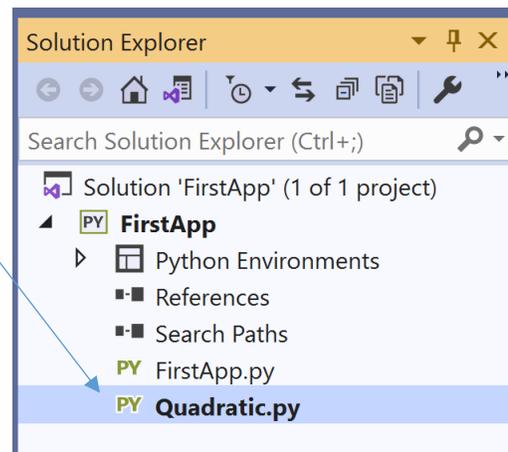
```

Note that Python uses underscores by convention in naming variables and functions with multiple words, e.g., `compute_quadratic_roots`. Variable and function names are typically written in lower case letters. File names and class names (you will learn this later) are typically declared with the first letter in the name as uppercase.

Right click on the Quadratic.py and select “Set as Startup file”



You will notice that, it will be shown in bold indicating that if you choose “Debug-> Start without Debugging”, this file will be run.



Run the program by choosing Debug->Start without Debugging. Your output will appear as:

```

C:\WINDOWS\system32\cmd.exe
please enter coeff. of x^2: 1
please enter coeff. of x: 6
please enter constant: 8
root1 = -2.0
root2 = -4.0
Press any key to continue . . .
  
```