Getting Productive with

Python and Visual Studio Code

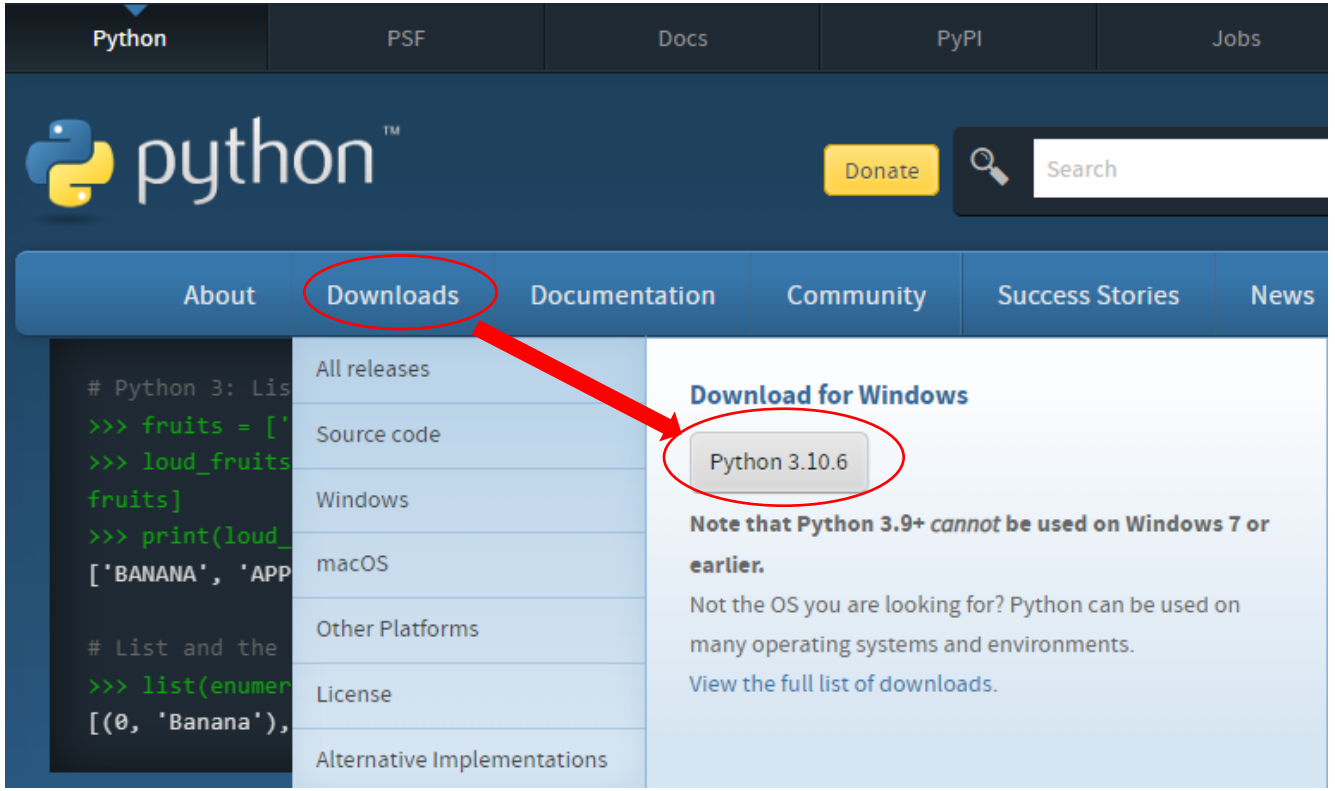
Development Environment construction

Prerequisities

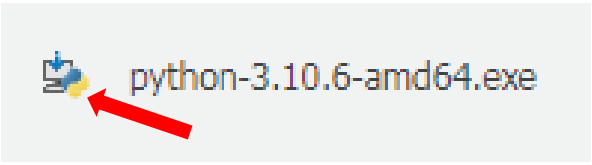
- Python installation
- Visual Studio Code (VS code) installation
- VS code for python extension

Python installation

- https://www.python.org/
- Downloads⇒Python 3.10.6

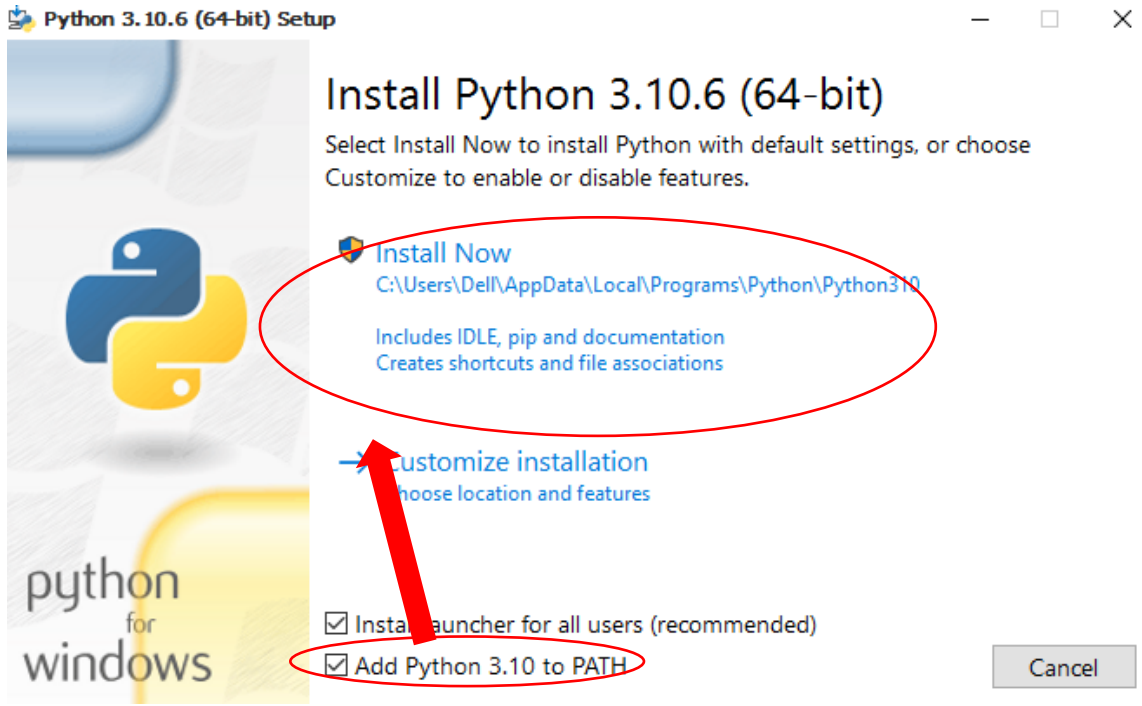


- Run the downloaded installer

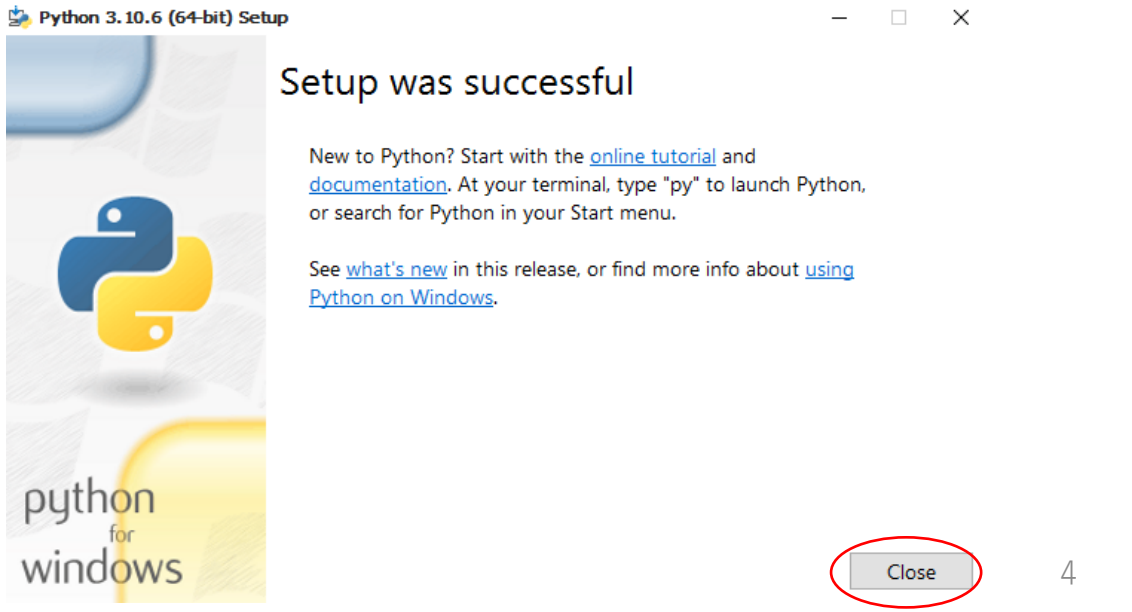


Python installation

- Check on “ Add Python 3.10 to PATH ”
- Click “ Install Now “

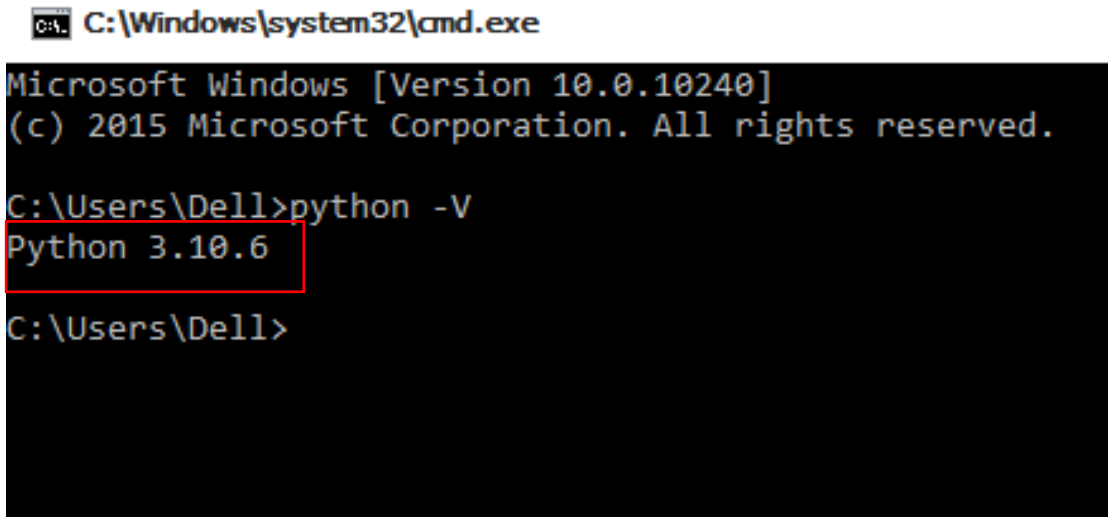


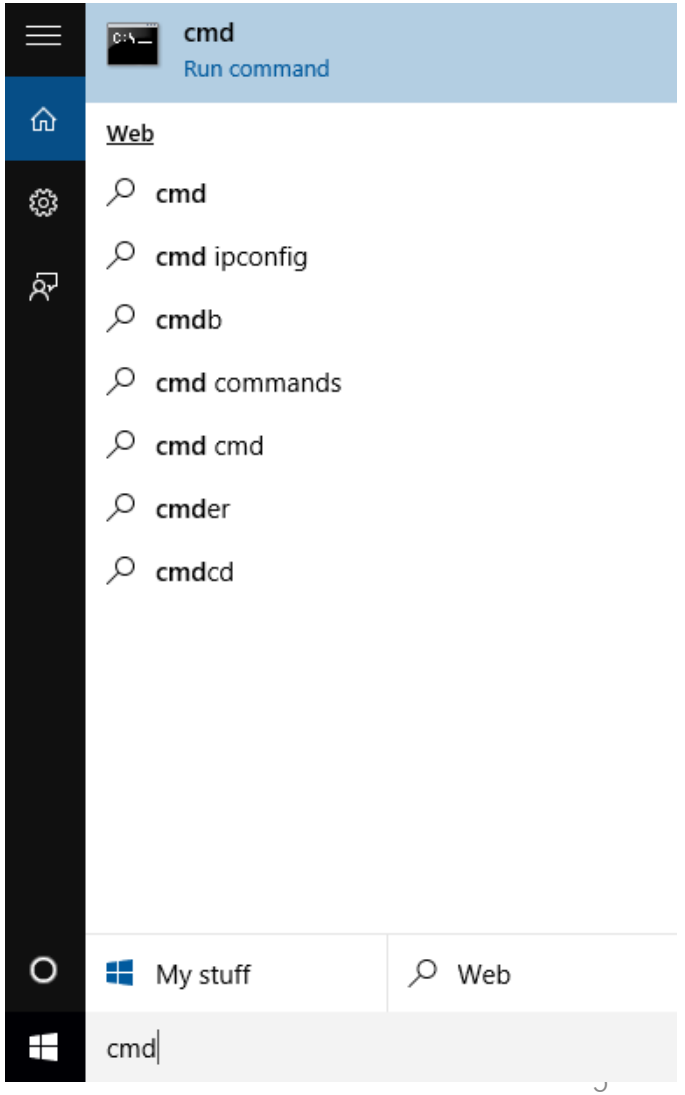
- Click “ Close ”



Python installation (Confirmation)

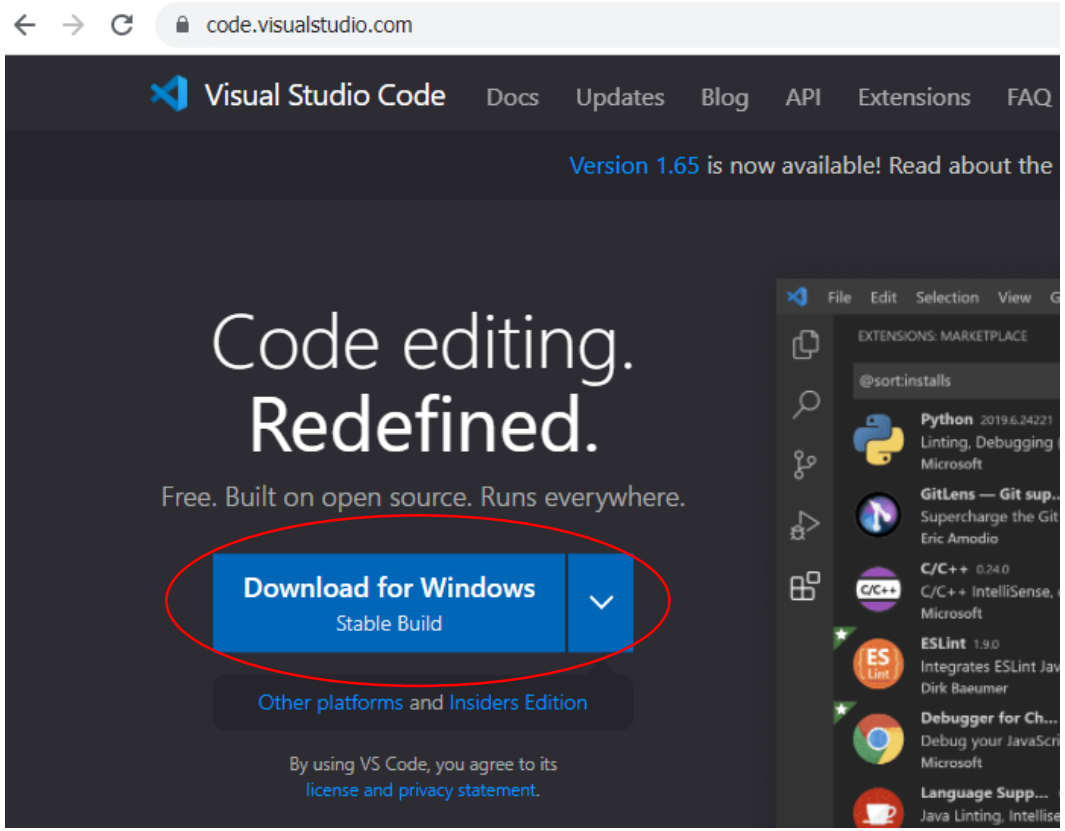
- After installation, enter “cmd” in the start menu
- Open command prompt
- Enter “python –V” to check python version

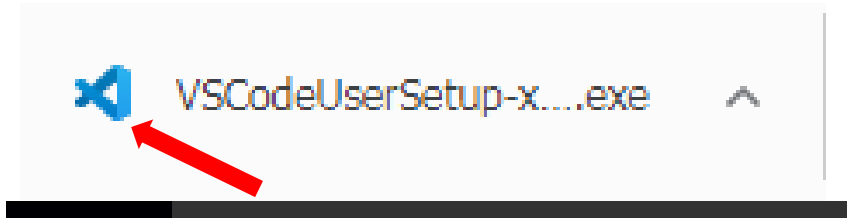




Visual Studio Code Installation

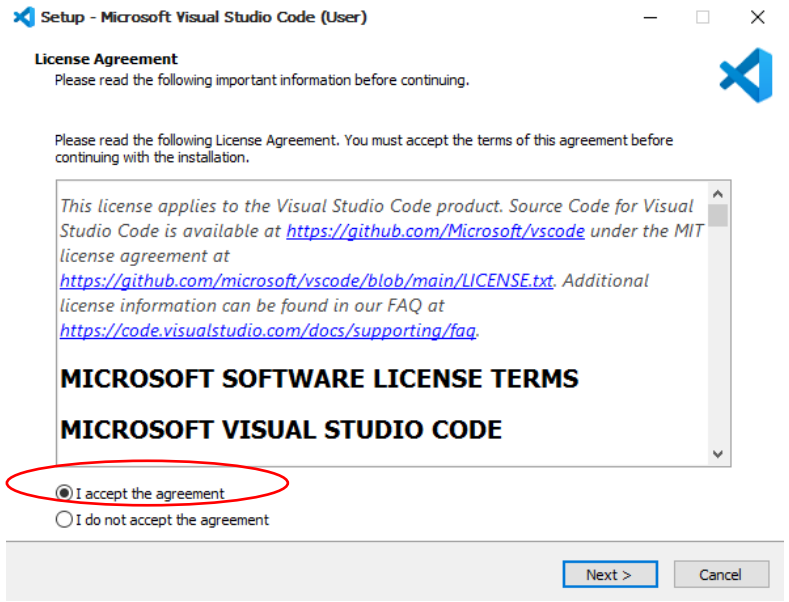
- https://code.visualstudio.com/
- Click the Download for Windows
- Run the installer

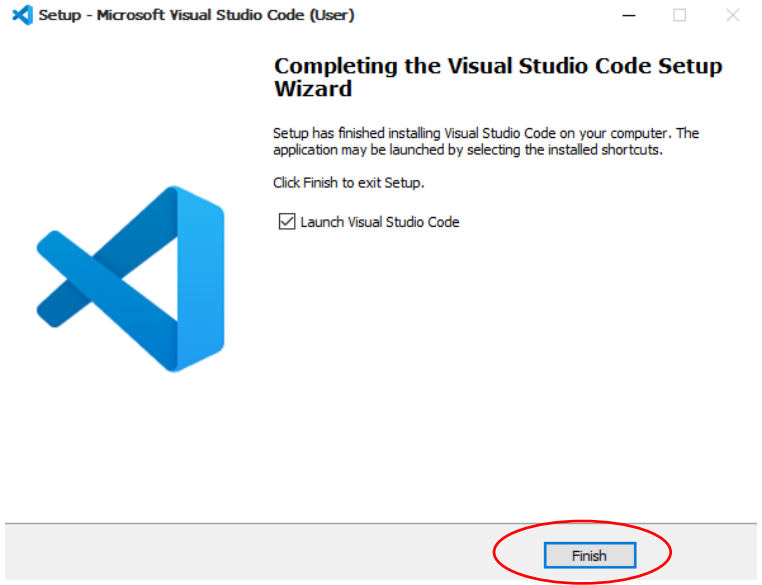




Visual Studio Code Installation

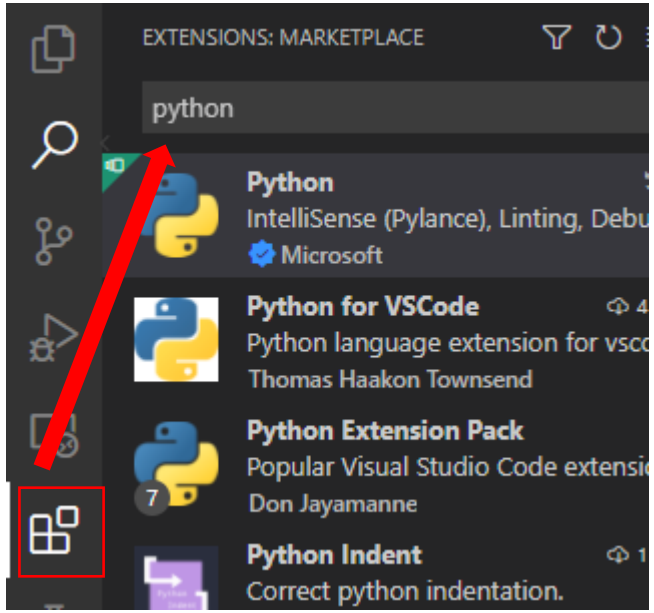
- Choose “ I accept the agreement ”
- Click “Next” button to continue installation
- Click "Finish"

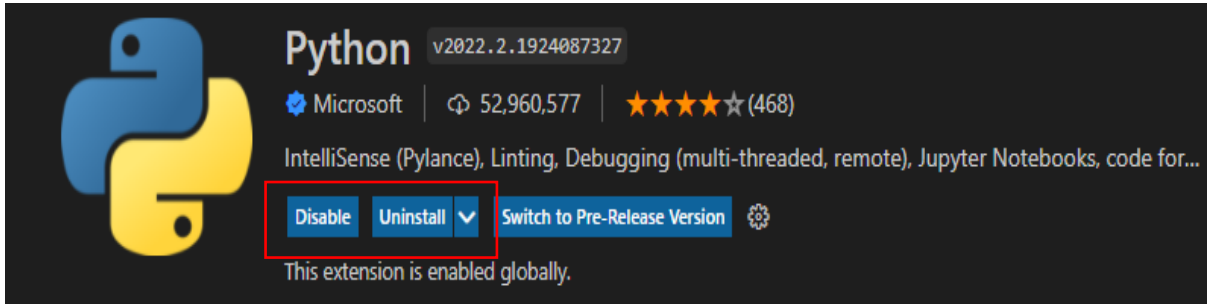


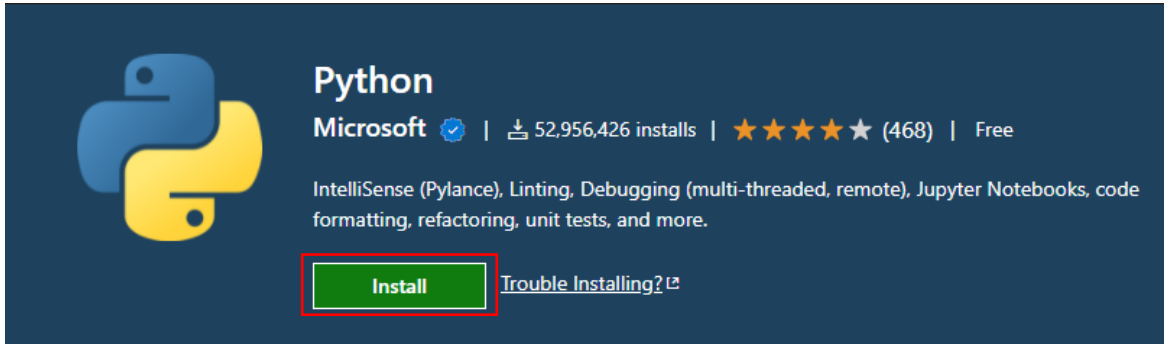


Python Extensions (python installation)

- Open the VS code IDE
- See EXTENSION MARKETPLACE for installing extensions
- Enter "python" in the search box
- Click Install
- After installation, check the installed extensions.





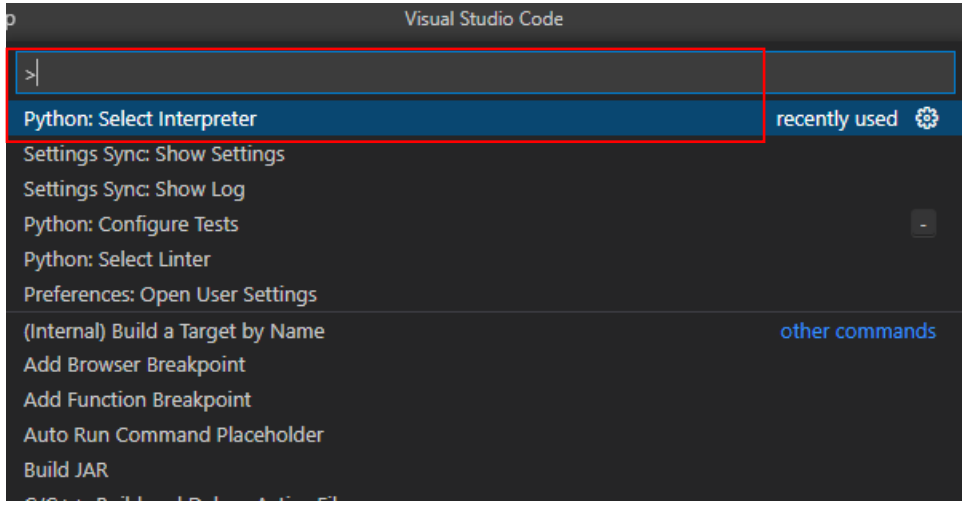


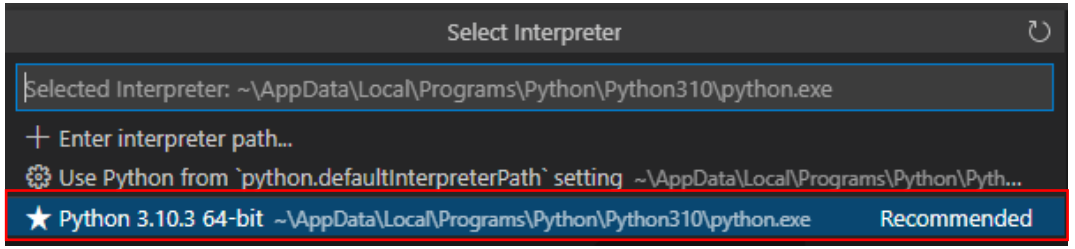
Python Interpreter for VS code

- To run Python code and get Python IntelliSense, you must tell VS Code which interpreter to use.

- **Command Palette** (Ctrl+Shift+P), then type **Python: Select Interpreter**

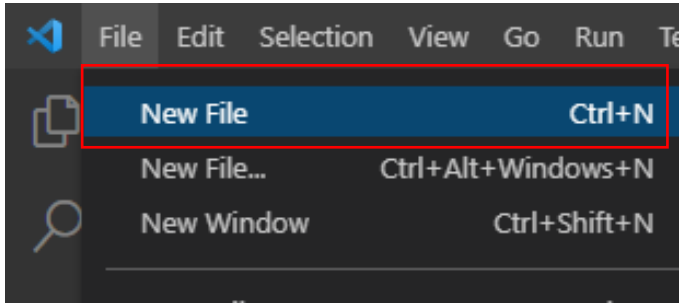
- Select the Recommended Interpreter



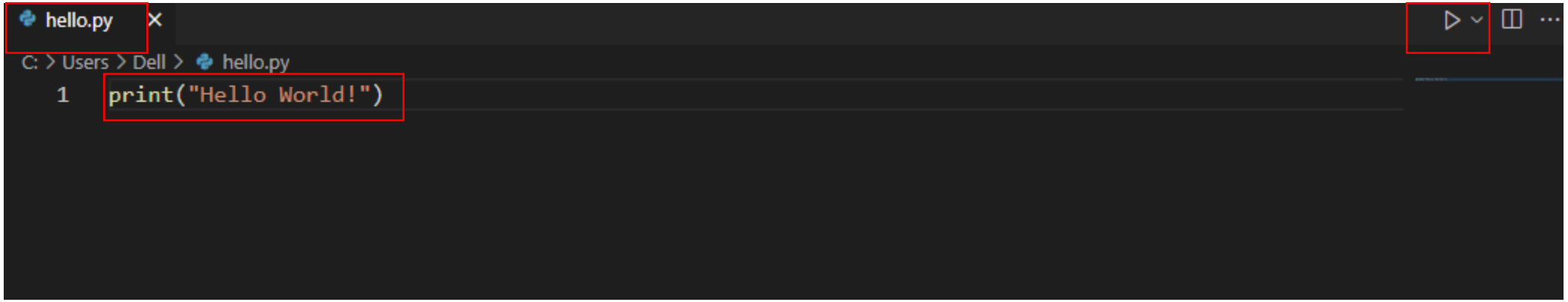


Example : Hello World

- Click on new file
- Save as “ Hello.py ”
  - print(”Hello World!”)

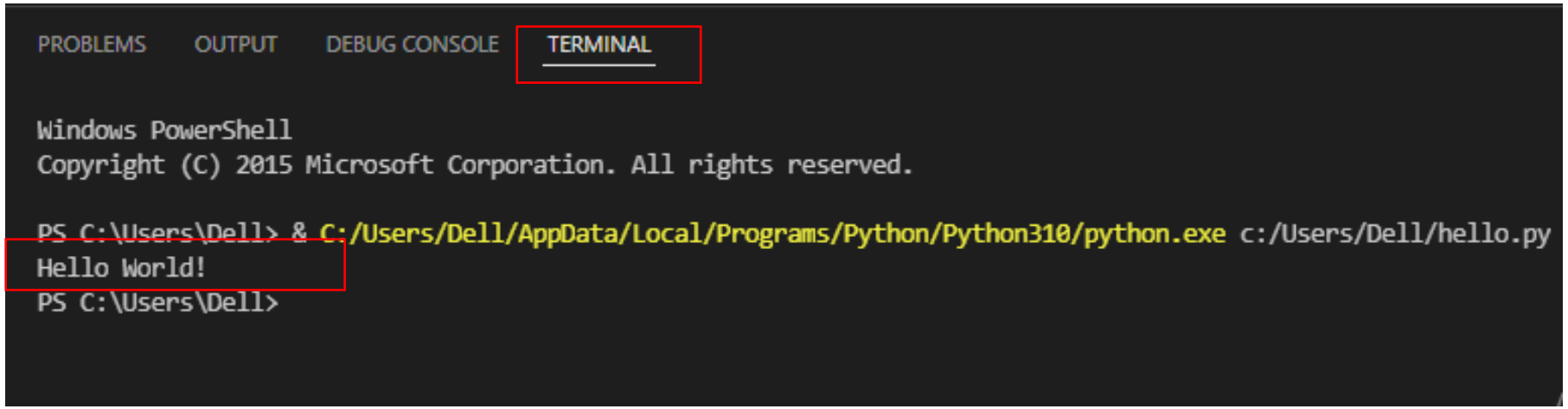


- Press the run button on the top right to run python code



Example : Hello World

- Hello World! appears in the terminal below .
- Congratulations, you have created your first Python program.



Research Participation

- Download the folders in the following link.

  Python_CWP_Libraries

  https://drive.google.com/file/d/1qi9aNMvZrkfPZFOz3caL53sRIpwNdWkj/view?usp=share_link

  Python_CWP_UI

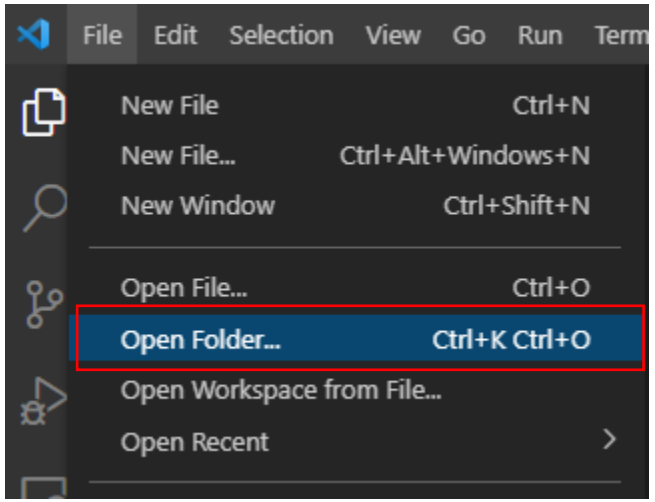
  https://drive.google.com/file/d/1wh7I7GDw-y5K4fcadymwp8QW6AaYLiXw/view?usp=share_link

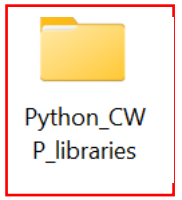
- You can see in **details** how to solve the code writing problem in next slides.

1. Download the folder in the following link

[Python_CWP_Libraries]

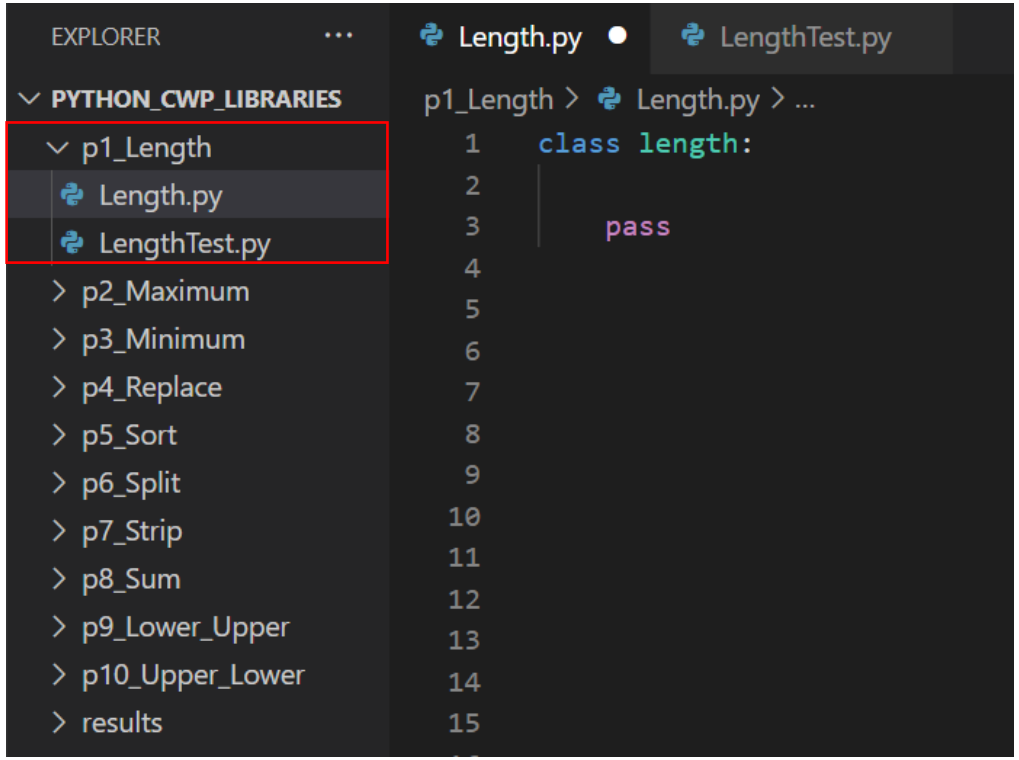
https://drive.google.com/file/d/1qi9aNMvZrkfPZFOz3caL53sRIpwNdWkj/view?usp=share_link





2. Open the downloaded folder in VS code.

- In each folder, there are **source code, test code** python files.



Source Code

p1_Length > Length.py > ...

```python
class length:

    pass

    # using len()
    def str_length1(x):
        result = len(x)
        return result

    # without using len()
    def str_length2(string):
        string_length = 0
        for i in string:
            string_length += 1
        return (string_length)
```

Test Code

Length.py ● LengthTest.py ×

p1_Length > LengthTest.py > main

```python
#  A library function of length
import unittest,sys,os
import inspect
from Length import length

class Test_StrLength(unittest.TestCase):
    # use len()
    def test_str_length1(self):
        source_code = inspect.getsource(length.str_length1)
        self.assertEqual(length.str_length1('Hello'), 5)
        self.assertIn("len(", source_code)

    # without use len()
    def test_str_length2(self):
        source_code = inspect.getsource(length.str_length2)
        self.assertEqual(length.str_length2('Hello'), 5)
        self.assertNotIn("len(", source_code)
```

```python
# Output the result file
def main(out=sys.stderr, verbosity=2):
    loader = unittest.TestLoader()
    suite = loader.loadTestsFromModule(sys.modules[__name__])
    unittest.TextTestRunner(out, verbosity=verbosity).run(suite)

if __name__ == '__main__':
    path1 = os.getcwd()
    path2 = "results"
    completeName = os.path.join(path1, path2, "p1.result")
    with open(completeName, 'w') as f:
        main(f)

# run the unittest
if __name__ == '__main__':
    unittest.main()
```

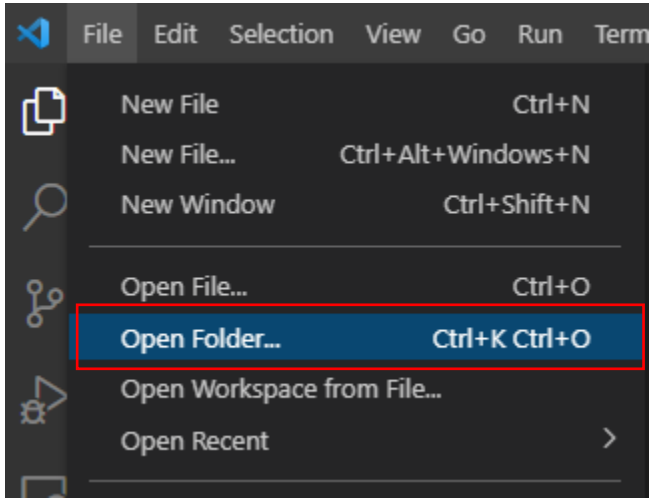
- See the **Test Code** then write Source code under pass keyword.
- Each program mentions to write the source code with **using python library or not**.

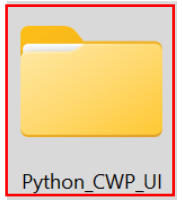
Output the result file and # run the unittest are not considerable in writing source code

1. Download the folders in the following link

[Python_CWP_UI]

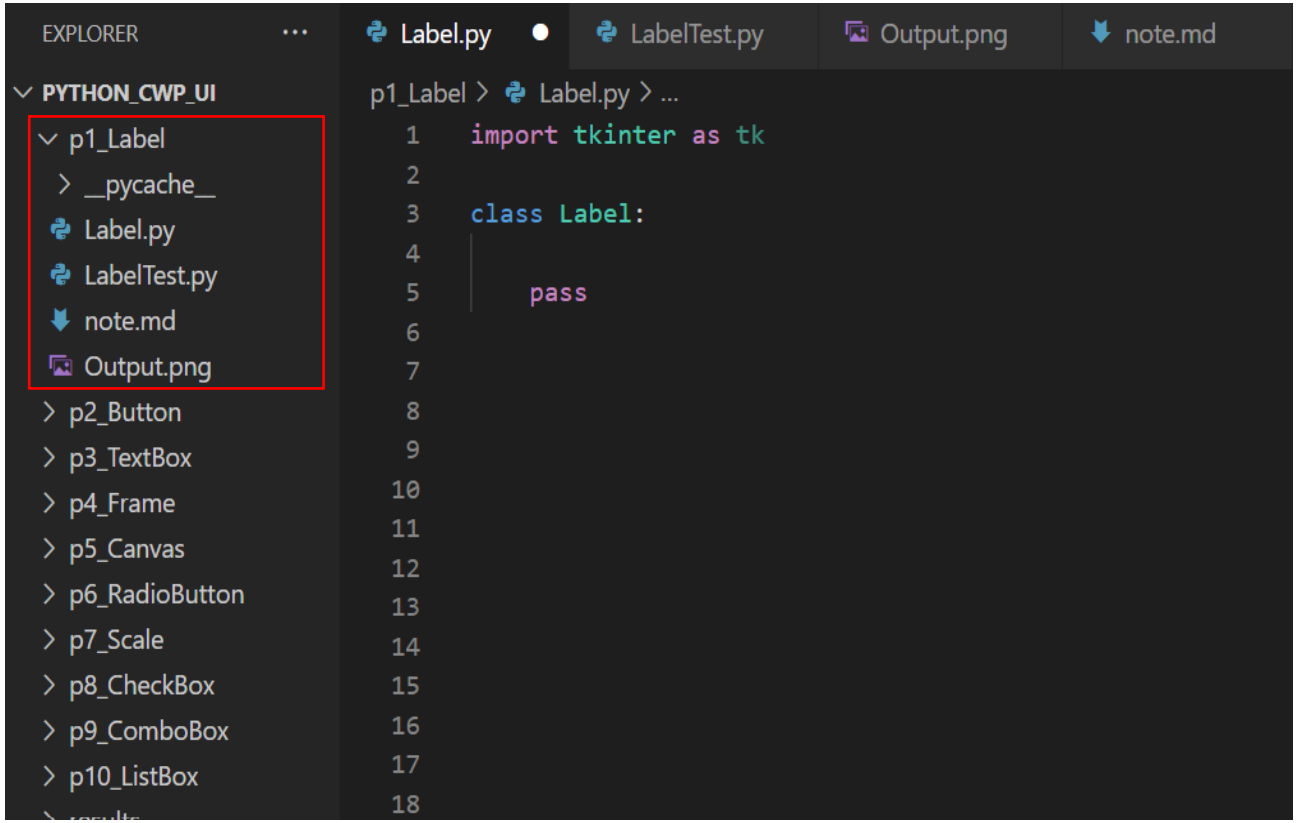
https://drive.google.com/file/d/1wh7I7GDw-y5K4fcadymwp8QW6AaYLiXw/view?usp=share_link





2. Open the downloaded folder in VS code.

- In each folder, there are **source code, test code** python files, **output.png** and reference **note**.



Source Code

p1_Label > Label.py > ...

```python
import tkinter as tk

class Label:

    pass

    def label_program():

        root = tk.Tk()
        root.title("Hello World")

        # Create a label with red text color
        label = tk.Label(root, text="Hello, World!", fg="red")
        label.pack()

        return root, label
```

Test Code

p1_Label > LabelTest.py > main

```python
import unittest,sys,os
import tkinter as tk
from Label import Label

class Label_Program(unittest.TestCase):
    def test_Label_Program(self):
        root, label = Label.label_program()

        # test root window properties
        self.assertIsInstance(root, tk.Tk)
        self.assertEqual(root.winfo_exists(), 1)
        self.assertEqual(root.title(), "Hello World")

        # test label properties
        self.assertIsInstance(label, tk.Label)
        self.assertEqual(label['text'], "Hello, World!")
        self.assertEqual(label['fg'], "red")
```

- See the **Test Code** then write Source code under pass keyword.
- See the **output.png** file to easily to understand program output.
- See **note** file for Reference and Program Description.

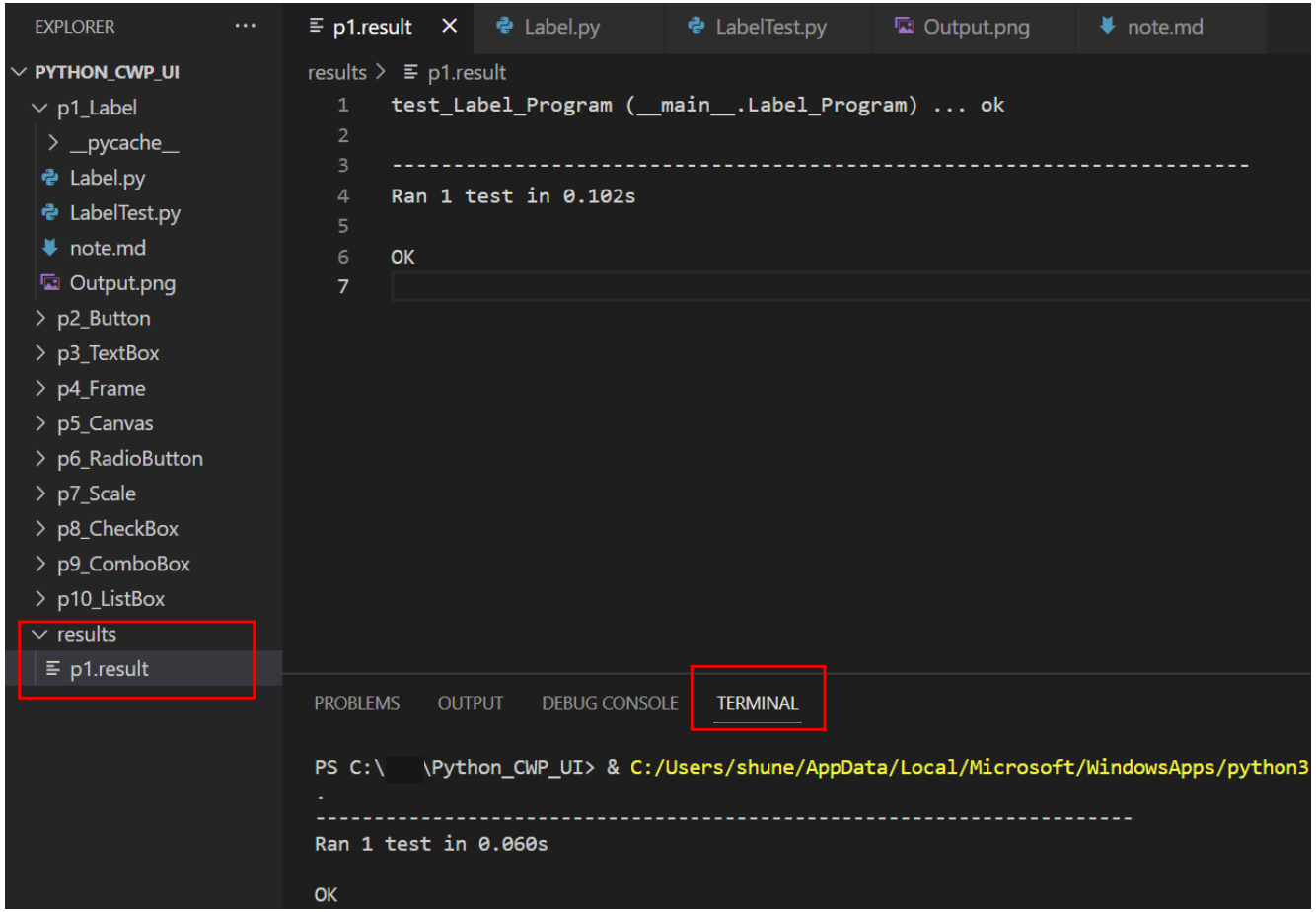
```python
# Output the result file
def main(out=sys.stderr, verbosity=2):
    loader = unittest.TestLoader()
    suite = loader.loadTestsFromModule(sys.modules[__name__])
    unittest.TextTestRunner(out, verbosity=verbosity).run(suite)

if __name__ == '__main__':
    path1 = os.getcwd()
    path2 = "results"
    completeName = os.path.join(path1, path2, "p1.result")
    with open(completeName, 'w') as f:
        main(f)

# run the unittest
if __name__ == '__main__':
    unittest.main()
```

Output the result file and # run the unittest are not considerable in writing source code

After finishing the source code, you can **check source code** by **running the test code.**
See the results in the terminal or “results” folder.



After completing ,

- Copy all source code files.
- Copy the ‘results’ folder.

- Send email to …
- **p1kl27uw@s.okayama-u.ac.jp**