

Name : Husnain

Intern ID : TN/IN01/PY/009

Email ID : husnain45605@gmail.com

Internship Domain : python

Week 6 task

Instructor Name : Hassan ALI

• Task 1:

Use math & statistics libraries to get square roots and average.

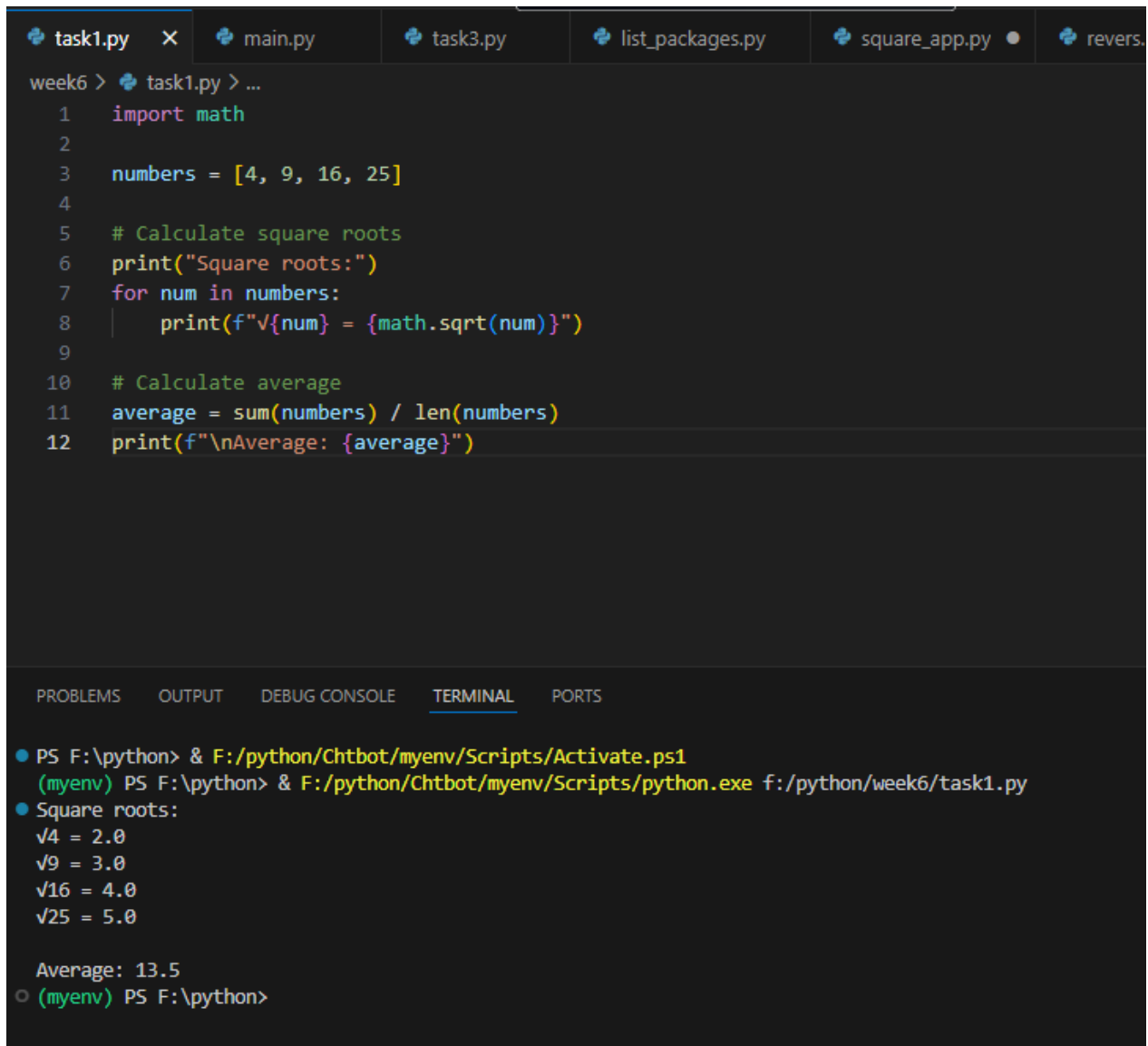
Description:

This Python script uses the built-in `math` and `statistics` libraries to:

- Calculate the square roots of a list of numbers
- Find the average (mean) of those numbers

It prints the original numbers, their square roots, and the average to the screen.

Output



The image shows a VS Code editor window with several tabs: task1.py, main.py, task3.py, list_packages.py, square_app.py, and revers. The active tab is task1.py, which contains the following Python code:

```
week6 > task1.py > ...
1  import math
2
3  numbers = [4, 9, 16, 25]
4
5  # Calculate square roots
6  print("Square roots:")
7  for num in numbers:
8      print(f"√{num} = {math.sqrt(num)}")
9
10 # Calculate average
11 average = sum(numbers) / len(numbers)
12 print(f"\nAverage: {average}")
```

The bottom panel shows the TERMINAL output:

```
● PS F:\python> & F:/python/Chtbot/myenv/Scripts/Activate.ps1
(myenv) PS F:\python> & F:/python/Chtbot/myenv/Scripts/python.exe f:/python/week6/task1.py
● Square roots:
√4 = 2.0
√9 = 3.0
√16 = 4.0
√25 = 5.0

Average: 13.5
○ (myenv) PS F:\python>
```

- **Task 2:**

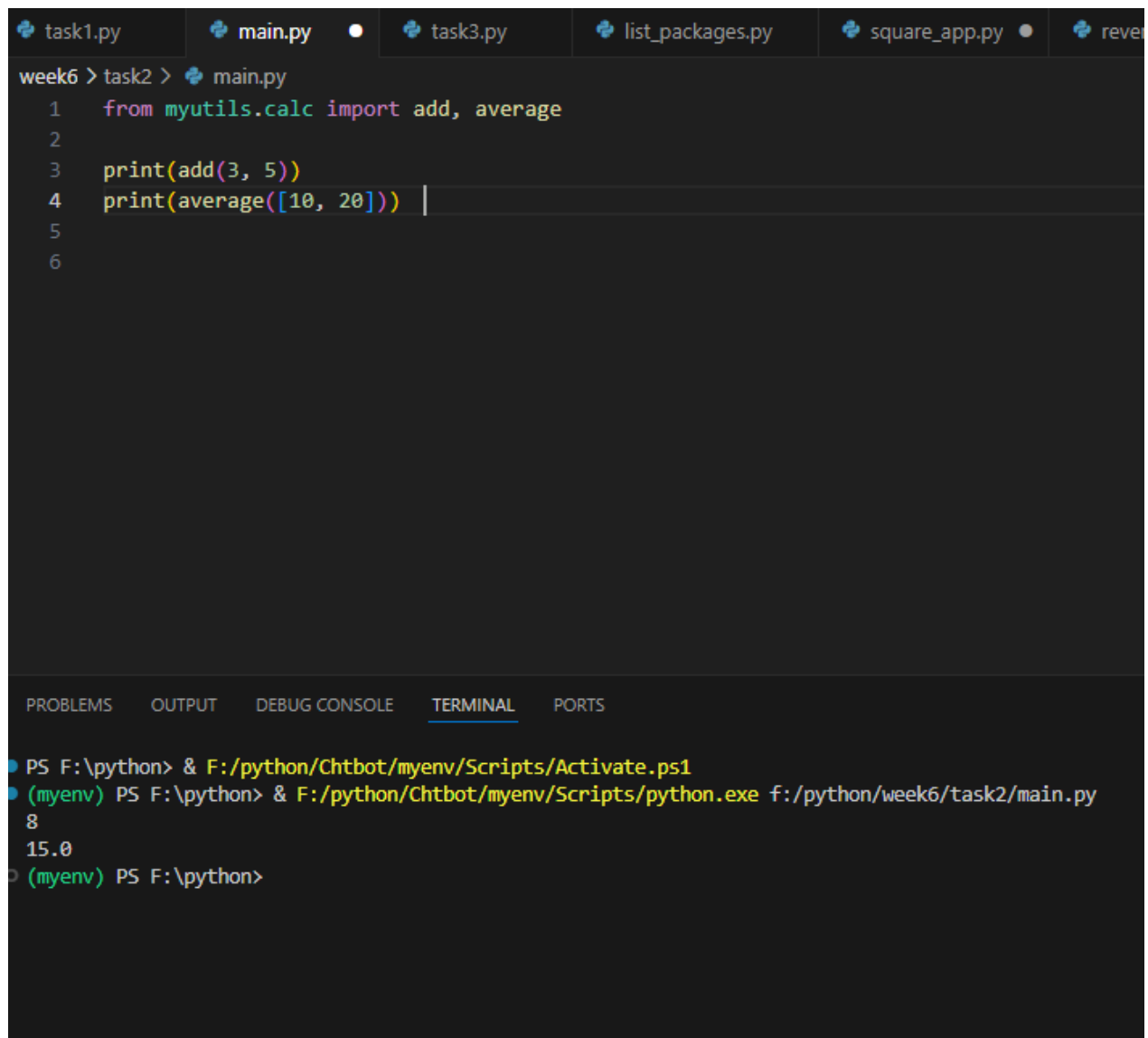
Create a custom package and import it in another script.

Description:

This task shows how to create a custom Python package named `mypackage` with a module `tools.py` that contains a `multiply()` function.

We then import this package in another script (`main.py`, placed outside the package folder) and use the function to perform multiplication and print the result.

Output



The image shows a VS Code editor window with a dark theme. At the top, there are tabs for several Python files: `task1.py`, `main.py` (which is the active file), `task3.py`, `list_packages.py`, `square_app.py`, and `rever`. The `main.py` file contains the following code:

```
1  from myutils.calc import add, average
2
3  print(add(3, 5))
4  print(average([10, 20]))
5
6
```

Below the editor, the **TERMINAL** panel is open, showing the execution of the script. The terminal output is as follows:

```
PS F:\python> & F:/python/Chtbot/myenv/Scripts/Activate.ps1
(myenv) PS F:\python> & F:/python/Chtbot/myenv/Scripts/python.exe f:/python/week6/task2/main.py
8
15.0
(myenv) PS F:\python>
```

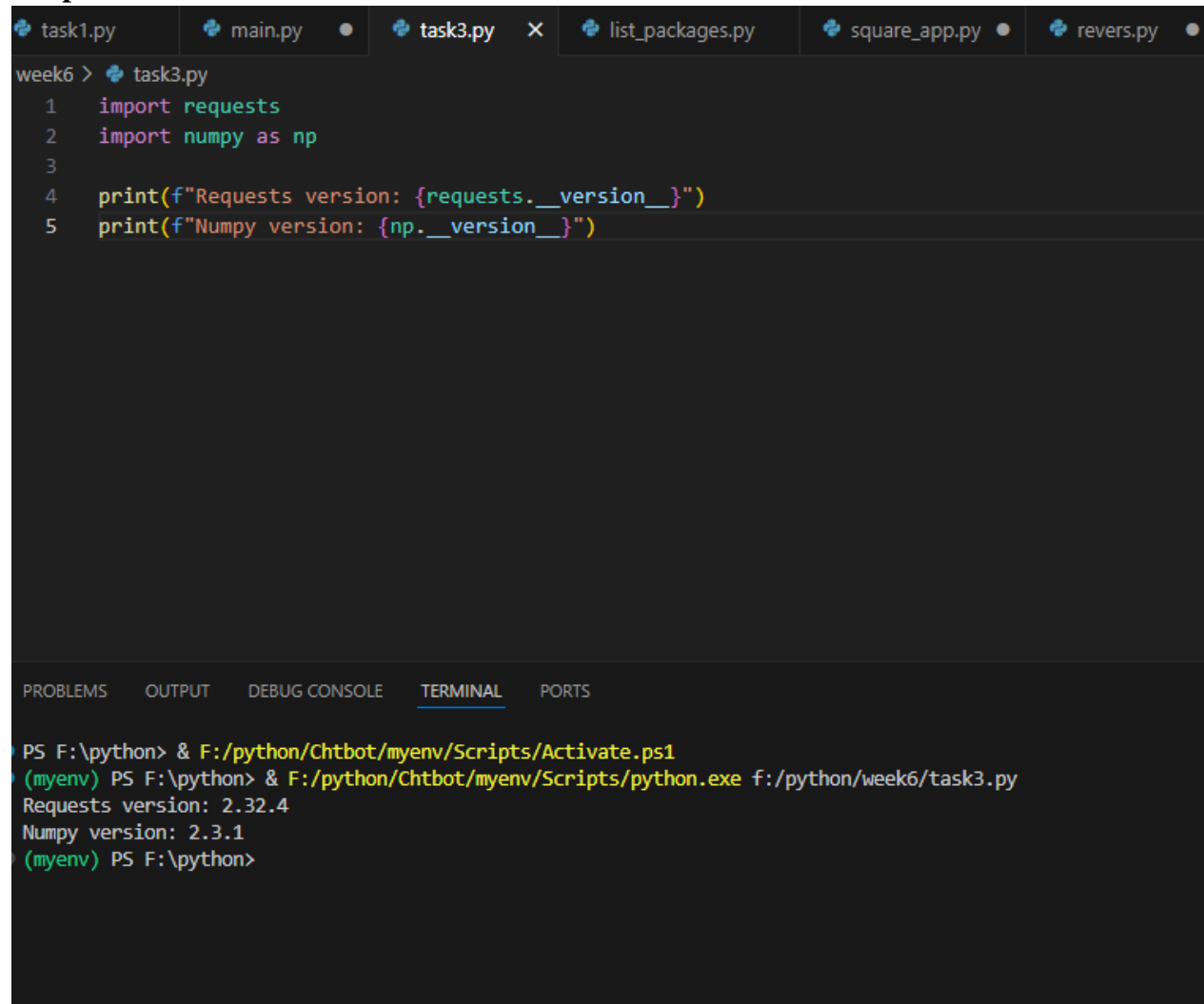
- **Task 3:**

Create a virtual environment, install requests & numpy, and print their versions.

description:

This task demonstrates how to set up a Python virtual environment, install external libraries (requests and numpy), and check their installed versions using a simple Python script.

Output



The screenshot shows a VS Code editor with several tabs: task1.py, main.py, task3.py (active), list_packages.py, square_app.py, and revers.py. The active file, task3.py, contains the following Python code:

```
1 import requests
2 import numpy as np
3
4 print(f"Requests version: {requests.__version__}")
5 print(f"Numpy version: {np.__version__}")
```

Below the editor, the TERMINAL tab is active, showing the execution of the script. The terminal output is as follows:

```
PS F:\python> & F:/python/Chtbot/myenv/Scripts/Activate.ps1
(myenv) PS F:\python> & F:/python/Chtbot/myenv/Scripts/python.exe f:/python/week6/task3.py
Requests version: 2.32.4
Numpy version: 2.3.1
(myenv) PS F:\python>
```

- **Task 4:**

Print list of all installed pip packages from Python code. description:

This script uses the `setuptools` module from `setuptools` to list all installed pip packages and their versions in the current Python environment.

Output

```
task1.py  main.py  task3.py  list_packages.py X  squar

week6 > list_packages.py > ...
1  import importlib.metadata
2
3  # Get all installed packages
4  installed_packages = importlib.metadata.distributions()
5
6  # Print name and version
7  for pkg in installed_packages:
8      print(f"{pkg.metadata['Name']}=={pkg.version}")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

tomlkit==0.13.3
tqdm==4.67.1
typer==0.16.0
typing_extensions==4.14.1
typing-inspection==0.4.1
tzdata==2025.2
urllib3==2.5.0
uvicorn==0.35.0
websockets==15.0.1
Werkzeug==3.1.3
zstandard==0.23.0
(myenv) PS F:\python>
```

- **Task 5:**

Create Gradio app that takes a number and returns its square.

description:

This is a simple Gradio web app that asks the user to enter a number, then calculates and displays its square when the “**Calculate**” button is clicked.

- ✓ Built with Gradio’s `Blocks` layout for a clean and modern interface.
- ✓ Includes a title and instructions to guide the user.
- ✓ Shows the result neatly in a textbox below.
- ✓ Uses a soft theme to make the interface look smooth and user-friendly.

Output

```
square_app.py
1  import gradio as gr
2  # Use one of these proper theme names:
3  gr.Interface(
4      lambda x: x*x,
5      gr.Number(),
6      "number",
7
8  ).launch()
```

x	output
<input type="text" value="0"/>	<input type="text" value="0"/>
<input type="button" value="Clear"/>	<input type="button" value="Flag"/>
<input type="button" value="Submit"/>	

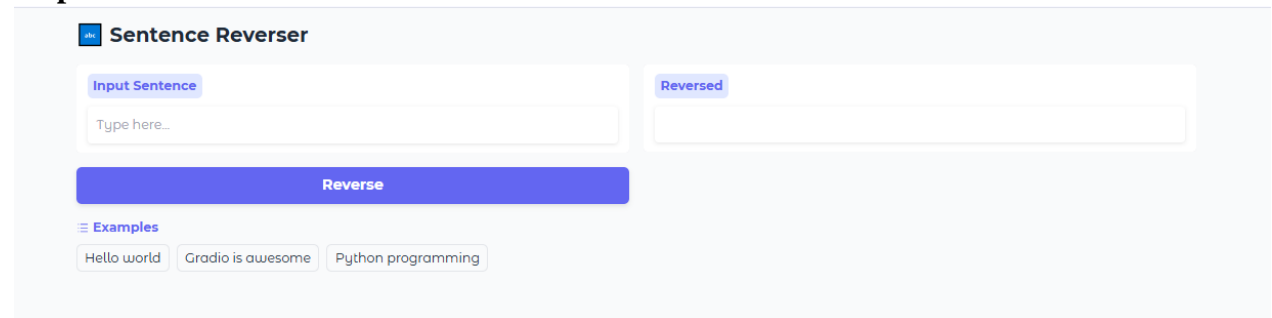
- **Task 6:**

Create Gradio interface that takes a sentence and returns it reversed. description:

This Gradio app takes a sentence typed by the user and returns it reversed when the “**Reverse**” button is clicked.

- ✓ Built using Gradio's `Blocks` layout for a clean and modern look.
- ✓ Includes a textbox for input, a button to trigger the action, and an output box to display the reversed sentence.
- ✓ Uses a soft theme to keep the interface simple and user-friendly.

output



The screenshot shows a web application titled "Sentence Reverser" with a blue header bar. Below the header, there is a light gray container. Inside this container, on the left, is a text input field with a blue label "Input Sentence" and a placeholder "Type here...". To the right of the input field is a blue button labeled "Reverse". To the right of the button is a text output field with a blue label "Reversed". Below the input field and button, there is a section titled "Examples" with a hamburger menu icon. Under "Examples", there are three buttons: "Hello world", "Gradio is awesome", and "Python programming".