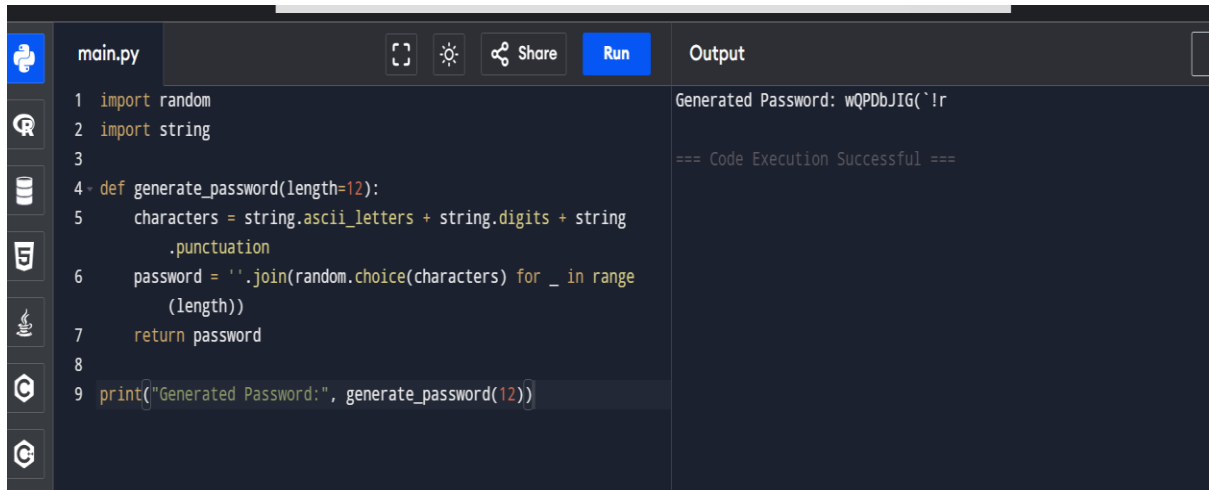


PYTHON BASIC PROJECTS

Python Program for random password generators

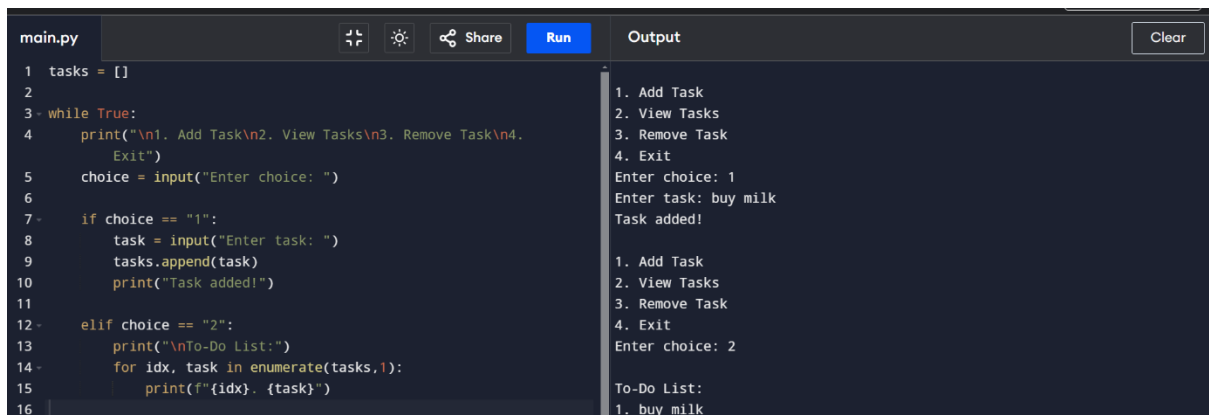


The screenshot shows a Python IDE with a file named 'main.py'. The code defines a function 'generate_password' that takes a length parameter (default 12) and returns a random password. The password is generated by joining random choices from a set of characters (ASCII letters, digits, and punctuation). The main program calls this function and prints the generated password. The output shows the generated password 'wQPdbJIG('!r' and a success message.

```
1 import random
2 import string
3
4 def generate_password(length=12):
5     characters = string.ascii_letters + string.digits + string
        .punctuation
6     password = ''.join(random.choice(characters) for _ in range
        (length))
7     return password
8
9 print("Generated Password:", generate_password(12))
```

Output: Generated Password: wQPdbJIG('!r
=== Code Execution Successful ===

Python Todolist Program

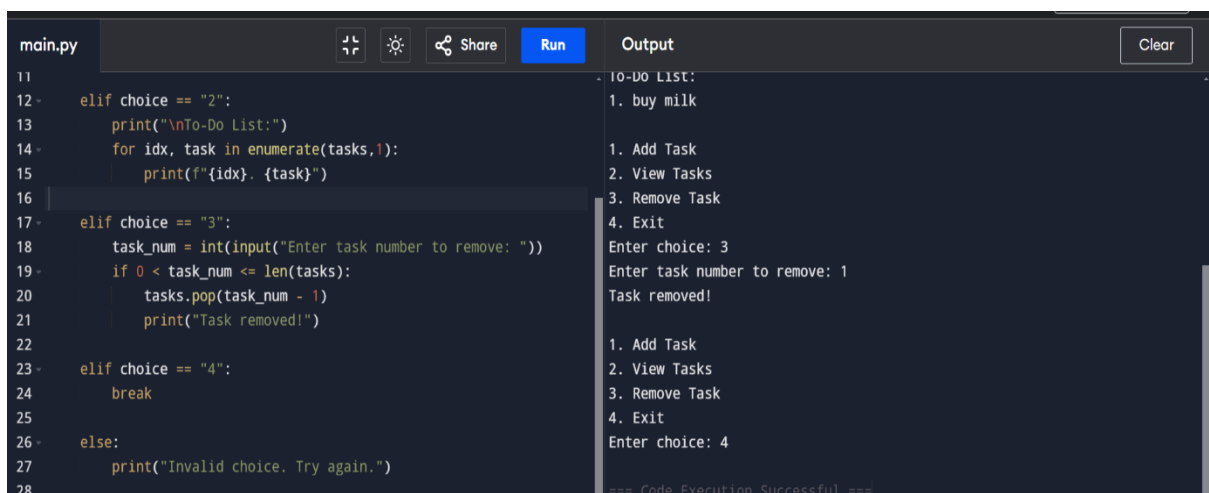


The screenshot shows a Python IDE with a file named 'main.py'. The code implements a to-do list program with a menu-driven interface. The menu options are: 1. Add Task, 2. View Tasks, 3. Remove Task, and 4. Exit. The program allows adding tasks, viewing the list, and removing tasks. The output shows the menu, the user adding a task 'buy milk', and the updated list.

```
1 tasks = []
2
3 while True:
4     print("\n1. Add Task\n2. View Tasks\n3. Remove Task\n4.
        Exit")
5     choice = input("Enter choice: ")
6
7     if choice == "1":
8         task = input("Enter task: ")
9         tasks.append(task)
10        print("Task added!")
11
12    elif choice == "2":
13        print("\nTo-Do List:")
14        for idx, task in enumerate(tasks,1):
15            print(f"{idx}. {task}")
16
```

Output: 1. Add Task
2. View Tasks
3. Remove Task
4. Exit
Enter choice: 1
Enter task: buy milk
Task added!

1. Add Task
2. View Tasks
3. Remove Task
4. Exit
Enter choice: 2
To-Do List:
1. buy milk



This screenshot shows the continuation of the to-do list program. It includes the logic for removing a task and the final success message. The output shows the user removing the task 'buy milk' and the updated list.

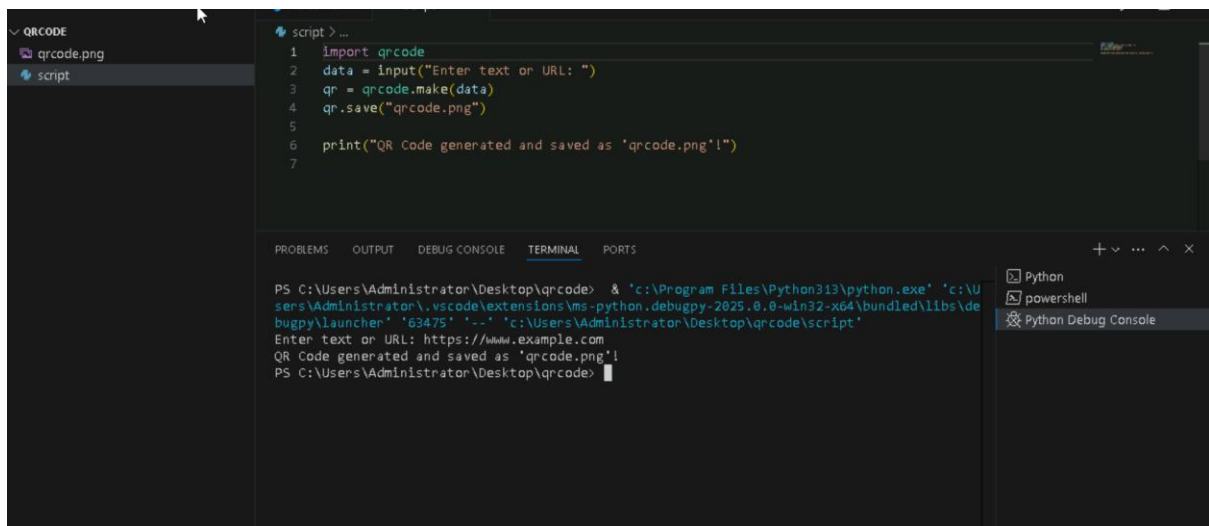
```
17
18    elif choice == "3":
19        task_num = int(input("Enter task number to remove: "))
20        if 0 < task_num <= len(tasks):
21            tasks.pop(task_num - 1)
22            print("Task removed!")
23
24    elif choice == "4":
25        break
26
27    else:
28        print("Invalid choice. Try again.")
```

Output: To-Do List:
1. buy milk

1. Add Task
2. View Tasks
3. Remove Task
4. Exit
Enter choice: 3
Enter task number to remove: 1
Task removed!

1. Add Task
2. View Tasks
3. Remove Task
4. Exit
Enter choice: 4
=== Code Execution Successful ===

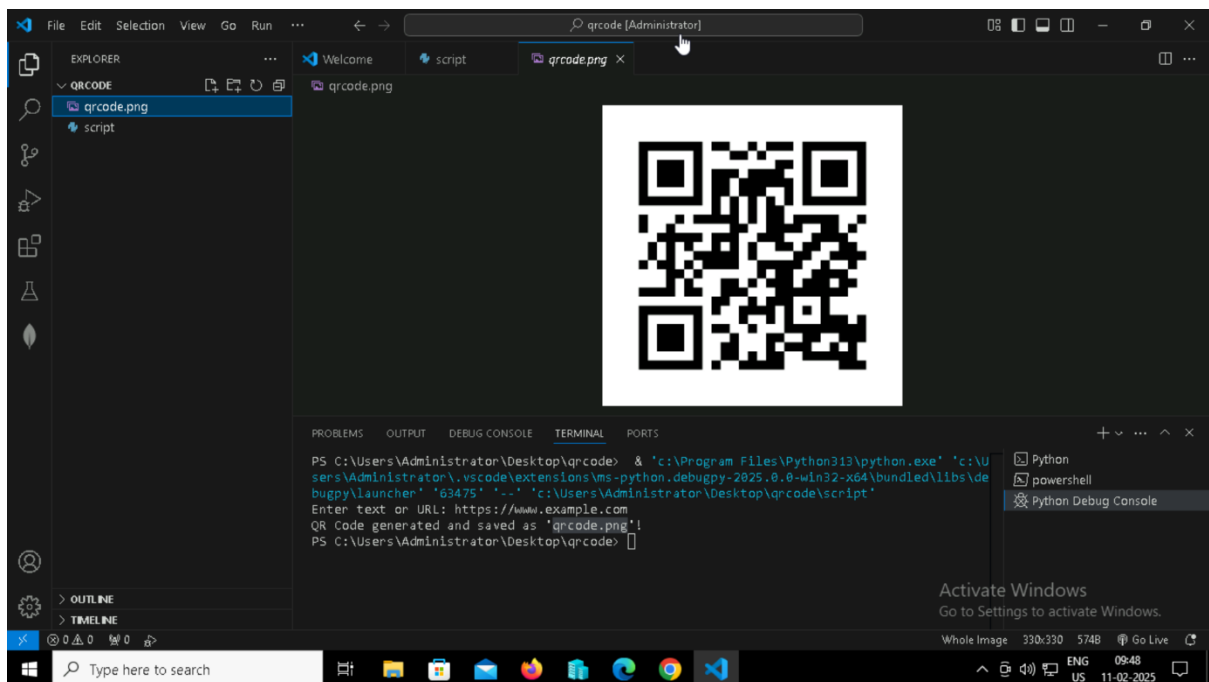
QR Code Genarator Program



This screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a folder named 'QRCODE' containing 'qrcode.png' and 'script'. The 'script' file is open in the editor, showing a Python script that takes user input and generates a QR code. The terminal shows the command to run the script and its output.

```
script > ...
1 import qrcode
2 data = input("Enter text or URL: ")
3 qr = qrcode.make(data)
4 qr.save("qrcode.png")
5
6 print("QR Code generated and saved as 'qrcode.png'!")
7
```

```
PS C:\Users\Administrator\Desktop\qrcode> & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2025.0.0-win32-x64\bundle\libs\debugpy\launcher' '63475' '--' 'c:\Users\Administrator\Desktop\qrcode\script'
Enter text or URL: https://www.example.com
QR Code generated and saved as 'qrcode.png'!
PS C:\Users\Administrator\Desktop\qrcode>
```



Number guessing game in python



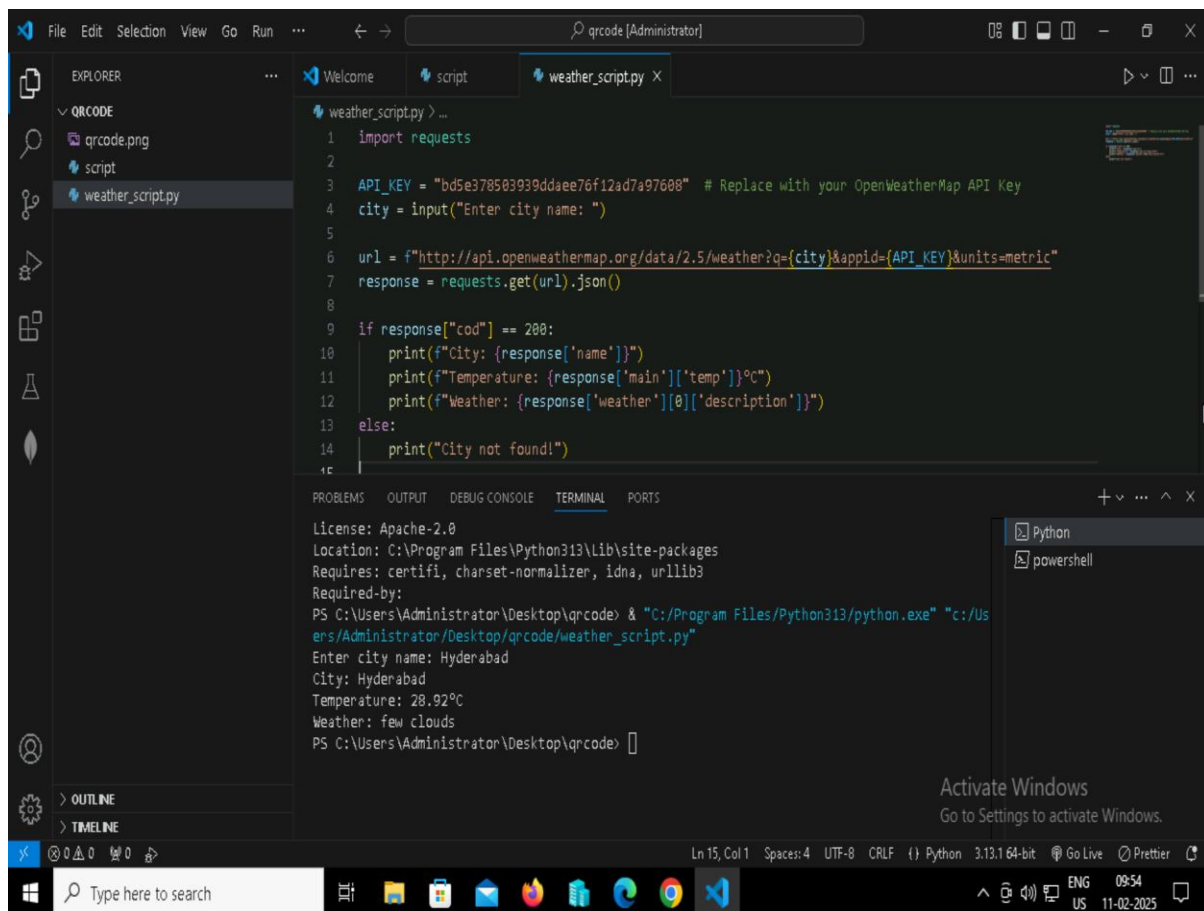
The screenshot shows a Python IDE with a file named `main.py`. The code is a number guessing game that generates a random number between 1 and 100 and prompts the user to guess it. The game provides feedback on whether the guess is too low, too high, or correct. The output window shows the game's execution with several guesses and feedback messages.

```
1 import random
2
3 number = random.randint(1, 100)
4
5 while True:
6     guess = int(input("Guess the number (1-100): "))
7     if guess < number:
8         print("Too low! Try again.")
9     elif guess > number:
10        print("Too high! Try again.")
11    else:
12        print("Congratulations! You guessed it right.")
13        break
14
```

Output:

```
Guess the number (1-100): 777
Too high! Try again.
Guess the number (1-100): -1
Too low! Try again.
Guess the number (1-100): 65
Too low! Try again.
Guess the number (1-100): 44
Too low! Try again.
Guess the number (1-100): 56
Too low! Try again.
Guess the number (1-100): 34
Too low! Try again.
Guess the number (1-100): 55
Too low! Try again.
```

Weather App (API-based)



The screenshot shows a Python IDE with a file named `weather_script.py`. The code is a weather app that uses the OpenWeatherMap API to fetch weather data for a given city. The app prompts the user to enter a city name and displays the city name, temperature, and weather description. The output window shows the app's execution, including the API call and the resulting weather data for Hyderabad.

```
1 import requests
2
3 API_KEY = "bd5e378503939ddaee76f12ad7a97608" # Replace with your OpenWeatherMap API Key
4 city = input("Enter city name: ")
5
6 url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={API_KEY}&units=metric"
7 response = requests.get(url).json()
8
9 if response["cod"] == 200:
10    print(f"City: {response['name']}")
11    print(f"Temperature: {response['main']['temp']}°C")
12    print(f"Weather: {response['weather'][0]['description']}")
13 else:
14    print("City not found!")
15
```

Output:

```
License: Apache-2.0
Location: C:\Program Files\Python313\Lib\site-packages
Requires: certifi, charset-normalizer, idna, urllib3
Required-by:
PS C:\Users\Administrator\Desktop\qrcode> & "C:/Program Files/Python313/python.exe" "c:/Users/Administrator/Desktop/qrcode/weather_script.py"
Enter city name: Hyderabad
City: Hyderabad
Temperature: 28.92°C
Weather: few clouds
PS C:\Users\Administrator\Desktop\qrcode>
```

