

## Lab 5: Ethical Foundations – Responsible AI Coding Practices

Week 3 – Monday

Name: B. Tejas Kumar

Hall Ticket No: 2303A51489

### Lab Objectives

- To understand ethical risks involved in AI-generated code.
- To identify issues related to privacy, security, and transparency.
- To analyze the responsibility of developers when using AI tools.
- To promote responsible and ethical AI coding practices.

### Lab Outcomes

After completing this lab, students will be able to:

- Identify insecure coding patterns generated by AI tools.
- Analyze privacy and security risks in AI-generated programs.
- Understand the importance of transparency and explainability.
- Recognize the role of human responsibility in ethical AI coding.

## Task Description #1: Privacy in API Usage

### Objective:

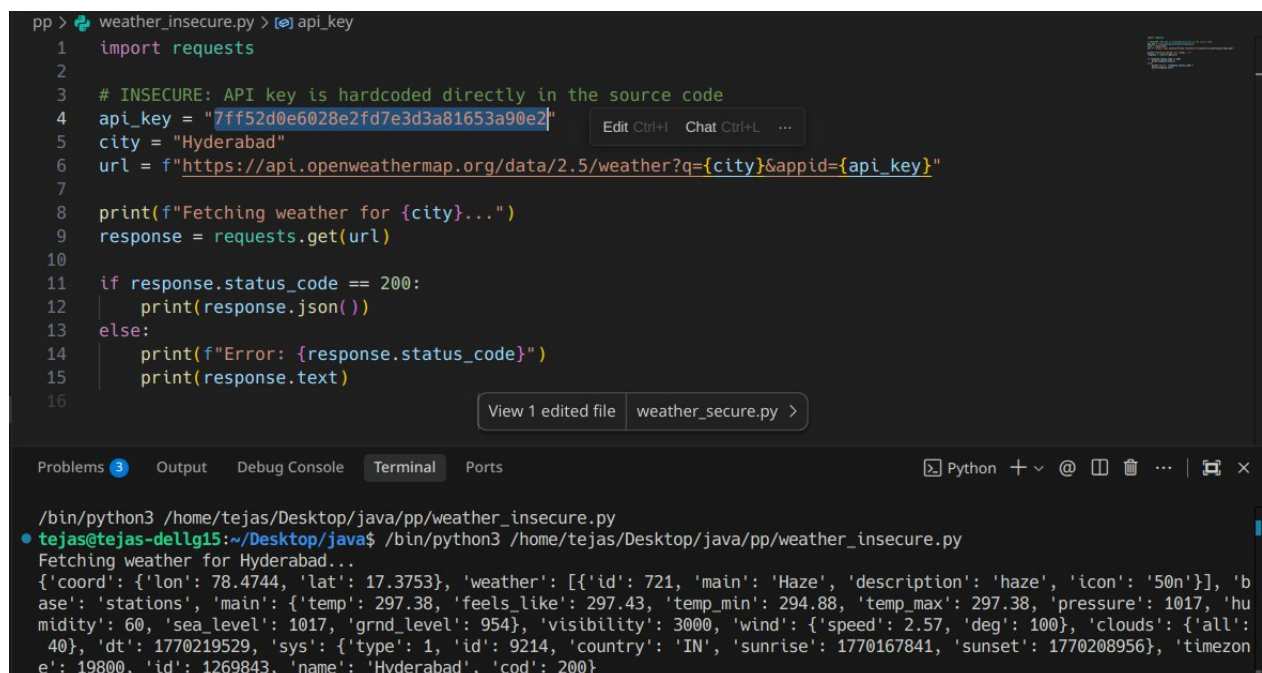
To generate a Python program that fetches weather data securely without exposing API keys.

### Risk Analysis:

AI-generated code may hardcode API keys directly in the program. This is unsafe and may lead to security breaches.

### Conclusion:

Using environment variables protects sensitive credentials and follows ethical security practices.



```
pp > weather_insecure.py > api_key
1 import requests
2
3 # INSECURE: API key is hardcoded directly in the source code
4 api_key = "7ff52d0e6028e2fd7e3d3a81653a90e2"
5 city = "Hyderabad"
6 url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}"
7
8 print(f"Fetching weather for {city}...")
9 response = requests.get(url)
10
11 if response.status_code == 200:
12     print(response.json())
13 else:
14     print(f"Error: {response.status_code}")
15     print(response.text)
16
```

```
/bin/python3 /home/tejas/Desktop/java/pp/weather_insecure.py
tejas@tejas-dellg15:~/Desktop/java$ /bin/python3 /home/tejas/Desktop/java/pp/weather_insecure.py
Fetching weather for Hyderabad...
{'coord': {'lon': 78.4744, 'lat': 17.3753}, 'weather': [{'id': 721, 'main': 'Haze', 'description': 'haze', 'icon': '50n'}], 'base': 'stations', 'main': {'temp': 297.38, 'feels_like': 297.43, 'temp_min': 294.88, 'temp_max': 297.38, 'pressure': 1017, 'humidity': 60, 'sea_level': 1017, 'grnd_level': 954}, 'visibility': 3000, 'wind': {'speed': 2.57, 'deg': 100}, 'clouds': {'all': 40}, 'dt': 1770219529, 'sys': {'type': 1, 'id': 9214, 'country': 'IN', 'sunrise': 1770167841, 'sunset': 1770208956, 'timezone': 19800, 'id': 1269843, 'name': 'Hyderabad', 'cod': 200}}
```

## Task Description #2: Privacy & Security in File Handling

### Objective:

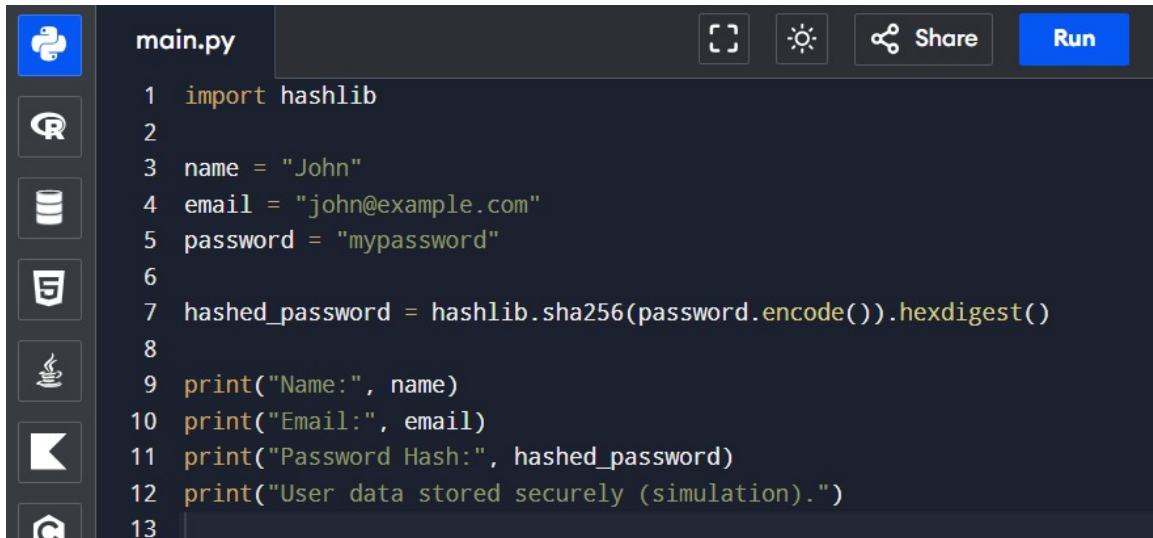
To analyze how AI-generated code stores user data and improve its security.

### Privacy Risk Identified:

Storing passwords in plain text can compromise user accounts.

### Conclusion:

Hashing passwords ensures data privacy and security.



```
main.py
1 import hashlib
2
3 name = "John"
4 email = "john@example.com"
5 password = "mypassword"
6
7 hashed_password = hashlib.sha256(password.encode()).hexdigest()
8
9 print("Name:", name)
10 print("Email:", email)
11 print("Password Hash:", hashed_password)
12 print("User data stored securely (simulation).")
13
```

Output	Clear
Name: John Email: john@example.com Password Hash: 89e01536ac207279409d4de1e5253e01f4a1769e696db0d6062ca9b8f56767c8 User data stored securely (simulation).	

### Task Description #3: Transparency in Algorithm Design

Objective:

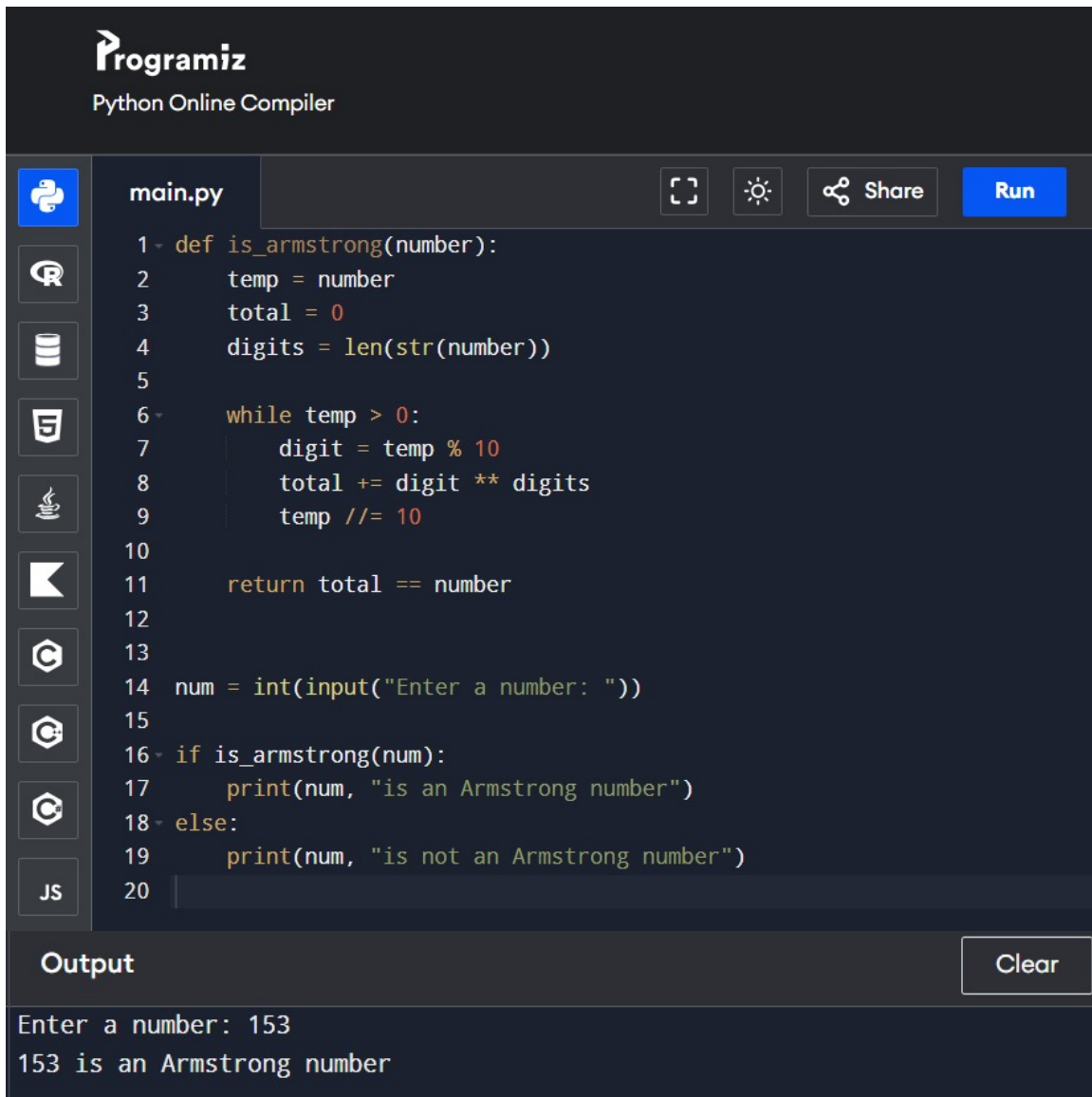
To create an Armstrong number checking program with clear explanation.

Explanation:

The program checks whether the sum of digits raised to the power of total digits equals the original number.

Conclusion:

The logic is simple, transparent, and easy to understand.



### Task Description #4: Transparency in Algorithm Comparison

Objective:

To implement and compare Bubble Sort and Quick Sort algorithms.

Explanation:

Bubble Sort is easy to understand but slow, whereas Quick Sort is faster and efficient for large datasets.

### Conclusion:

Choosing the right algorithm improves performance and ethical decision-making.

Programiz

Python Online Compiler

main.py

Share

Run

1 def bubble\_sort(arr):

2 n = len(arr)

3 for i in range(n):

4 for j in range(0, n - i - 1):

5 if arr[j] > arr[j + 1]:

6 arr[j], arr[j + 1] = arr[j + 1], arr[j]

7 return arr

8

9

10 def quick\_sort(arr):

11 if len(arr) <= 1:

12 return arr

13

14 pivot = arr[len(arr) // 2]

15 left = [x for x in arr if x < pivot]

16 middle = [x for x in arr if x == pivot]

17 right = [x for x in arr if x > pivot]

18

19 return quick\_sort(left) + middle + quick\_sort(right)

20

21

22 arr = [5, 2, 9, 1, 7]

23

24 print("Original Array:", arr)

25 print("Bubble Sort Output:", bubble\_sort(arr.copy()))

26 print("Quick Sort Output:", quick\_sort(arr))

Output

Original Array: [5, 2, 9, 1, 7]  
Bubble Sort Output: [1, 2, 5, 7, 9]  
Quick Sort Output: [1, 2, 5, 7, 9]

## Task Description #5: Transparency in AI Recommendations

Objective:

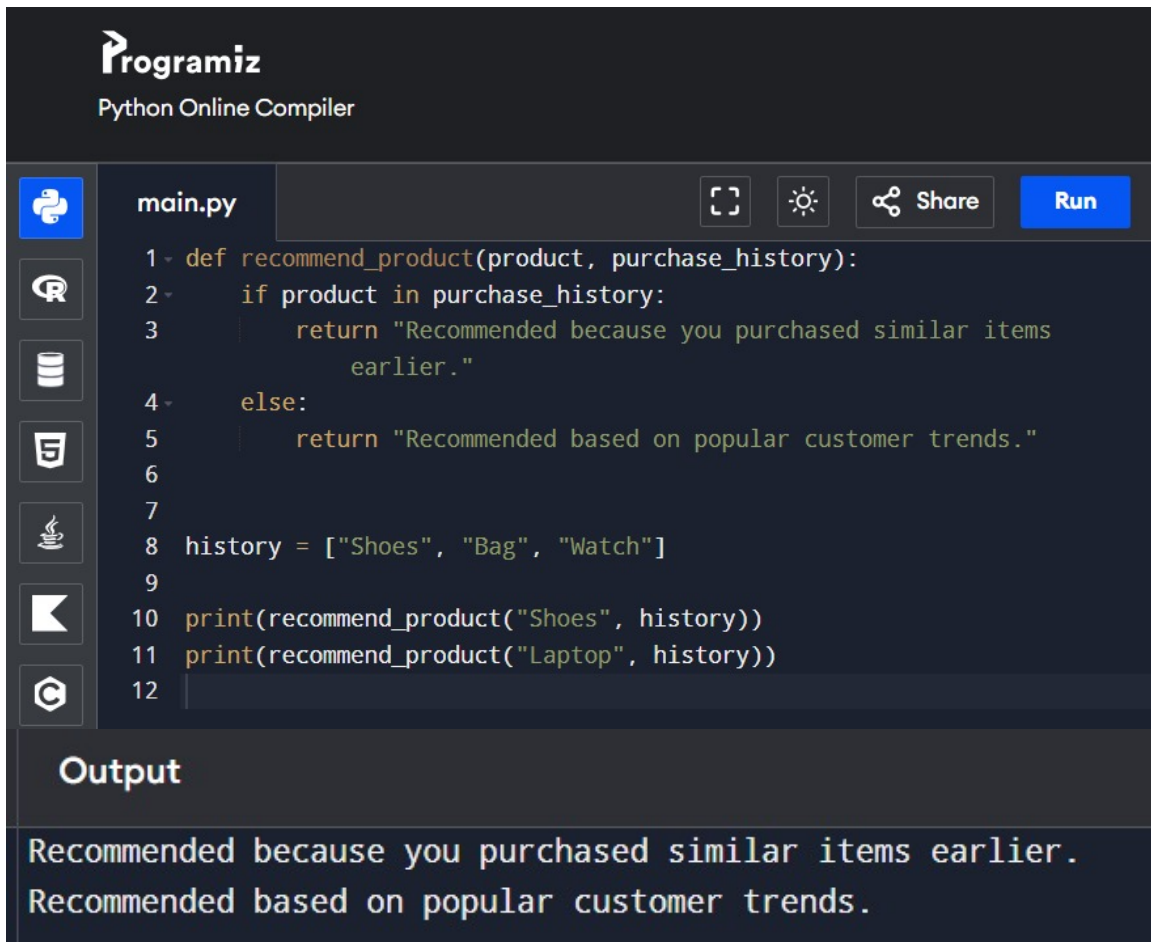
To build a recommendation system that explains why items are suggested.

Explanation:

Providing reasons for recommendations improves transparency and trust.

Conclusion:

Explainable AI systems are more ethical and user-friendly.



The screenshot displays the Programiz Python Online Compiler interface. At the top, the logo 'Programiz' and the text 'Python Online Compiler' are visible. Below the header, there is a sidebar with icons for Python, REPL, Database, File Manager, Shell, and Settings. The main area shows a file named 'main.py' with the following Python code:

```
1 def recommend_product(product, purchase_history):
2     if product in purchase_history:
3         return "Recommended because you purchased similar items
4             earlier."
5     else:
6         return "Recommended based on popular customer trends."
7
8 history = ["Shoes", "Bag", "Watch"]
9
10 print(recommend_product("Shoes", history))
11 print(recommend_product("Laptop", history))
12
```

Below the code editor, the 'Output' section shows the results of the execution:

```
Recommended because you purchased similar items earlier.
Recommended based on popular customer trends.
```

### Overall Reflection

This lab emphasizes that AI tools assist developers, but ethical responsibility lies with humans. Ensuring security, privacy, and transparency is essential in responsible AI coding.