AI ASSISTED CODING

LAB EXAM-4

NAME : K.SAI KARTHIK

HT.no : 2403A52043

BATCH : 03

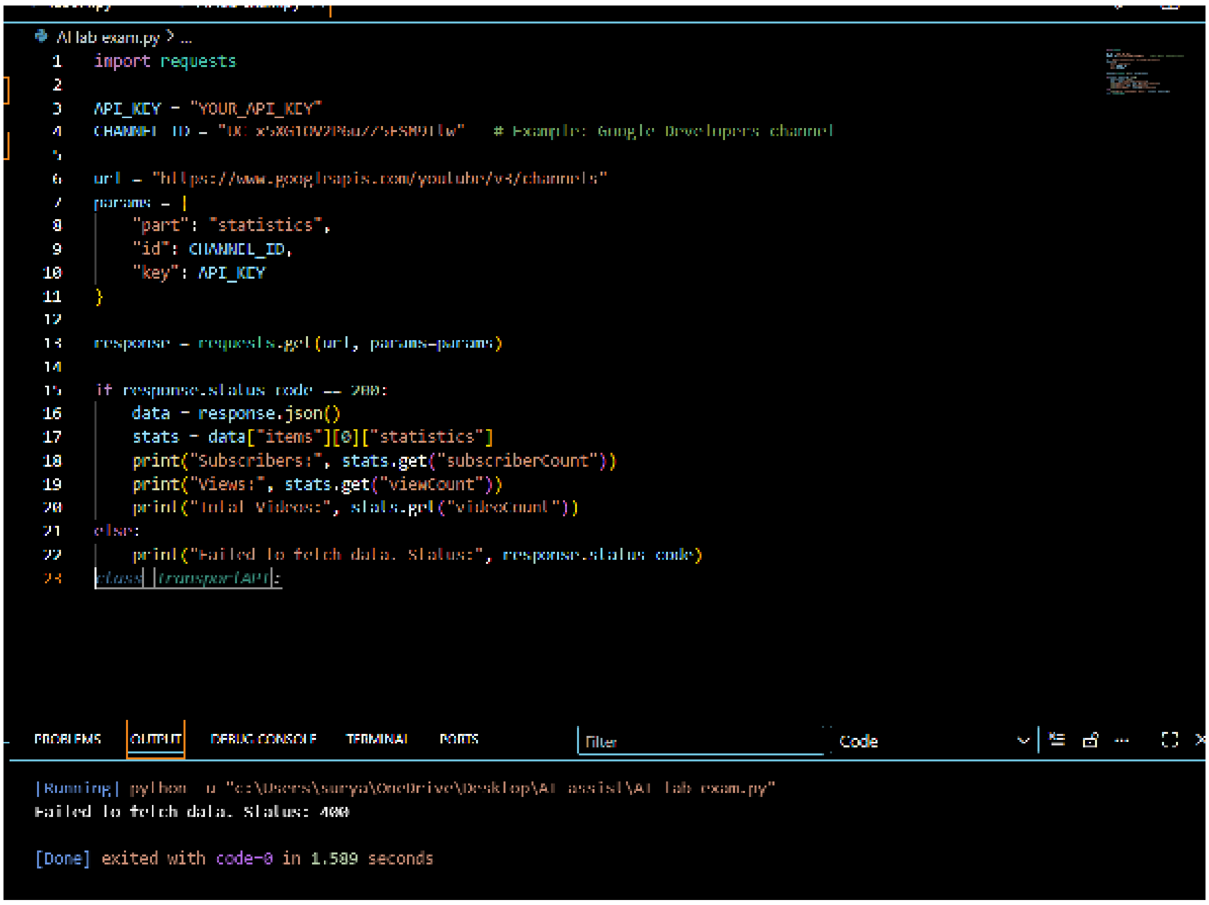
Q1: (API Integration)

(a) Connect to YouTube Data API to fetch channel statistics.

Prompt:

Write Python code that connects to the YouTube Data API (v3) to fetch statistics of a given YouTube channel (subscribers, views, video count). Use API key authentication.

CODE :



AI lab exam.py > ...

```python
import requests

API_KEY = "YOUR_API_KEY"
CHANNEL_ID = "UC_x5XG1OV2P6uZZ5FSM9Ttw"    # Example: Google Developers channel

url = "https://www.googleapis.com/youtube/v3/channels"
params = {
    "part": "statistics",
    "id": CHANNEL_ID,
    "key": API_KEY
}

response = requests.get(url, params=params)

if response.status_code == 200:
    data = response.json()
    stats = data["items"][0]["statistics"]
    print("Subscribers:", stats.get("subscriberCount"))
    print("Views:", stats.get("viewCount"))
    print("Total Videos:", stats.get("videoCount"))
else:
    print("Failed to fetch data. Status:", response.status_code)
class transportAPI:
```

```
[Running] python -u "c:\Users\surya\OneDrive\Desktop\AI assist\AI lab exam.py"
Failed to fetch data. Status: 400

[Done] exited with code=0 in 1.589 seconds
```

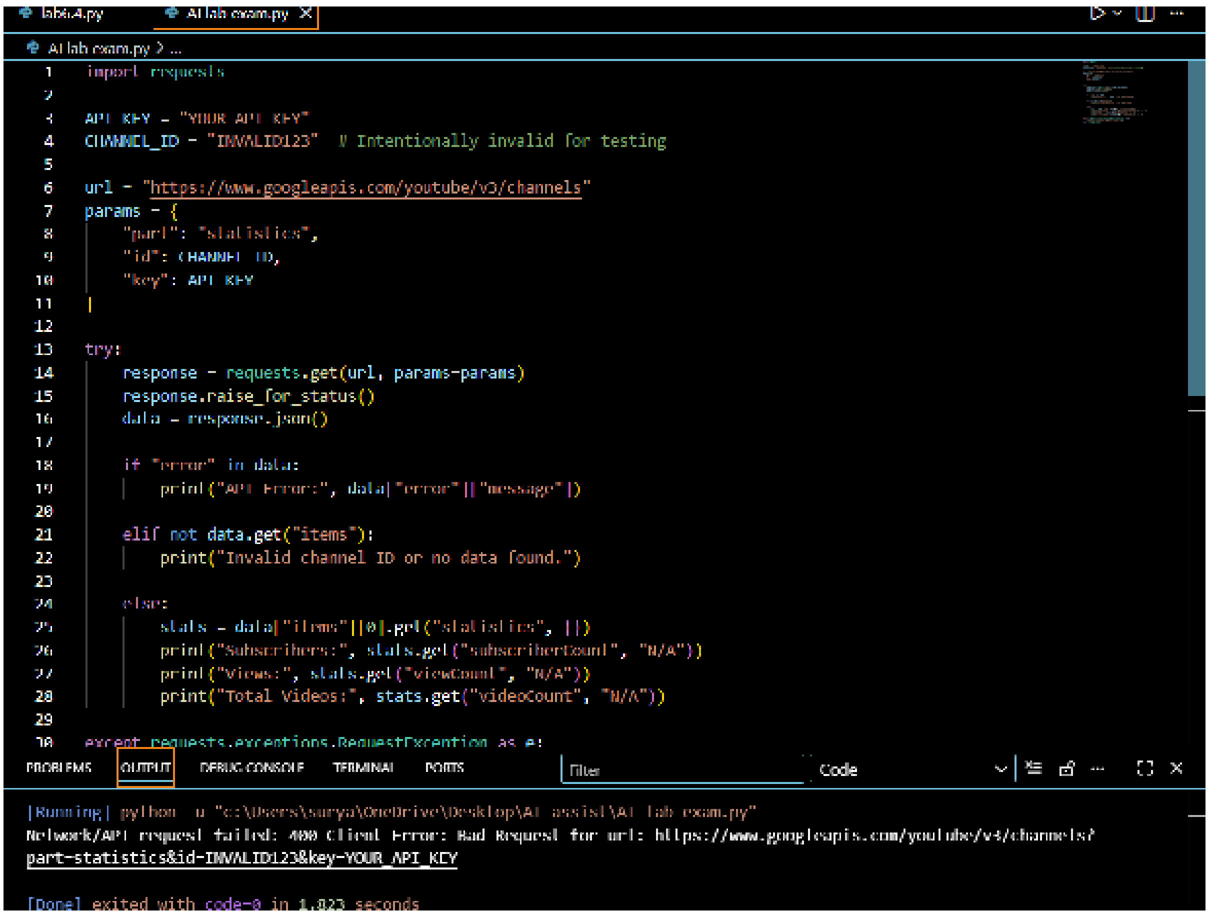
Observation :

- The code sends a GET request to youtube/v3/channels.
- Returns key statistics (subscribers, views, video count).
- Requires a valid API key.
- JSON response structure is nested; statistics are inside items   statistics.

(b) Handle API quota or invalid channel ID errors.

Prompt : Modify the code to gracefully handle YouTube API quota errors, invalid channel IDs, or missing response fields.

CODE :



```python
import requests

API_KEY = "YOUR_API_KEY"
CHANNEL_ID = "INVALID123"  # Intentionally invalid for testing

url = "https://www.googleapis.com/youtube/v3/channels"
params = {
    "part": "statistics",
    "id": CHANNEL_ID,
    "key": API_KEY
}

try:
    response = requests.get(url, params=params)
    response.raise_for_status()
    data = response.json()

    if "error" in data:
        print("API Error:", data["error"]["message"])

    elif not data.get("items"):
        print("Invalid channel ID or no data found.")

    else:
        stats = data["items"][0].get("statistics", {})
        print("Subscribers:", stats.get("subscriberCount", "N/A"))
        print("Views:", stats.get("viewCount", "N/A"))
        print("Total Videos:", stats.get("videoCount", "N/A"))

except requests.exceptions.RequestException as e:
```

```
[Running] python -u "c:\Users\surya\OneDrive\Desktop\AI assist\AI lab exam.py"
Network/API request failed: 400 Client Error: Bad Request for url: https://www.googleapis.com/youtube/v3/channels?
part=statistics&id=INVALID123&key=YOUR_API_KEY

[Done] exited with code=0 in 1.023 seconds
```

Observation:

- Handles quota exceeded (data["error"]).
- Detects invalid channel ID (empty items list).
- Uses exception handling to catch network errors.

- Prevents runtime crashes due to missing fields.

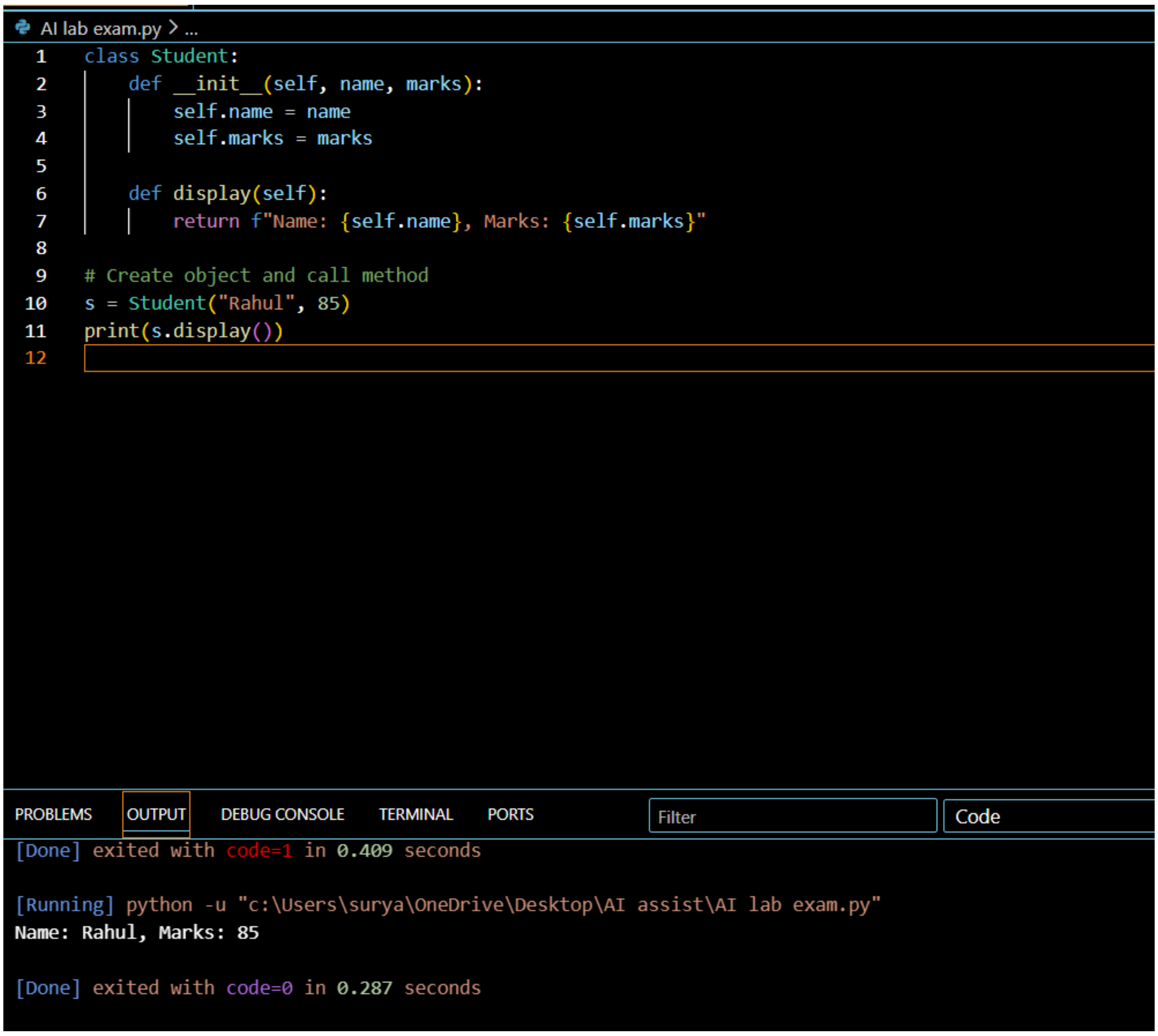
Q2. (Code Translation)

(a) Translate a Python class into Kotlin.

Prompt:

Translate the given Python class into Kotlin while preserving functionality.

CODE :



AI lab exam.py > ...

```python
class Student:
    def __init__(self, name, marks):
        self.name = name
        self.marks = marks

    def display(self):
        return f"Name: {self.name}, Marks: {self.marks}"

# Create object and call method
s = Student("Rahul", 85)
print(s.display())
```

```
[Done] exited with code=1 in 0.409 seconds

[Running] python -u "c:\Users\surya\OneDrive\Desktop\AI assist\AI lab exam.py"
Name: Rahul, Marks: 85

[Done] exited with code=0 in 0.287 seconds
```

Observation :

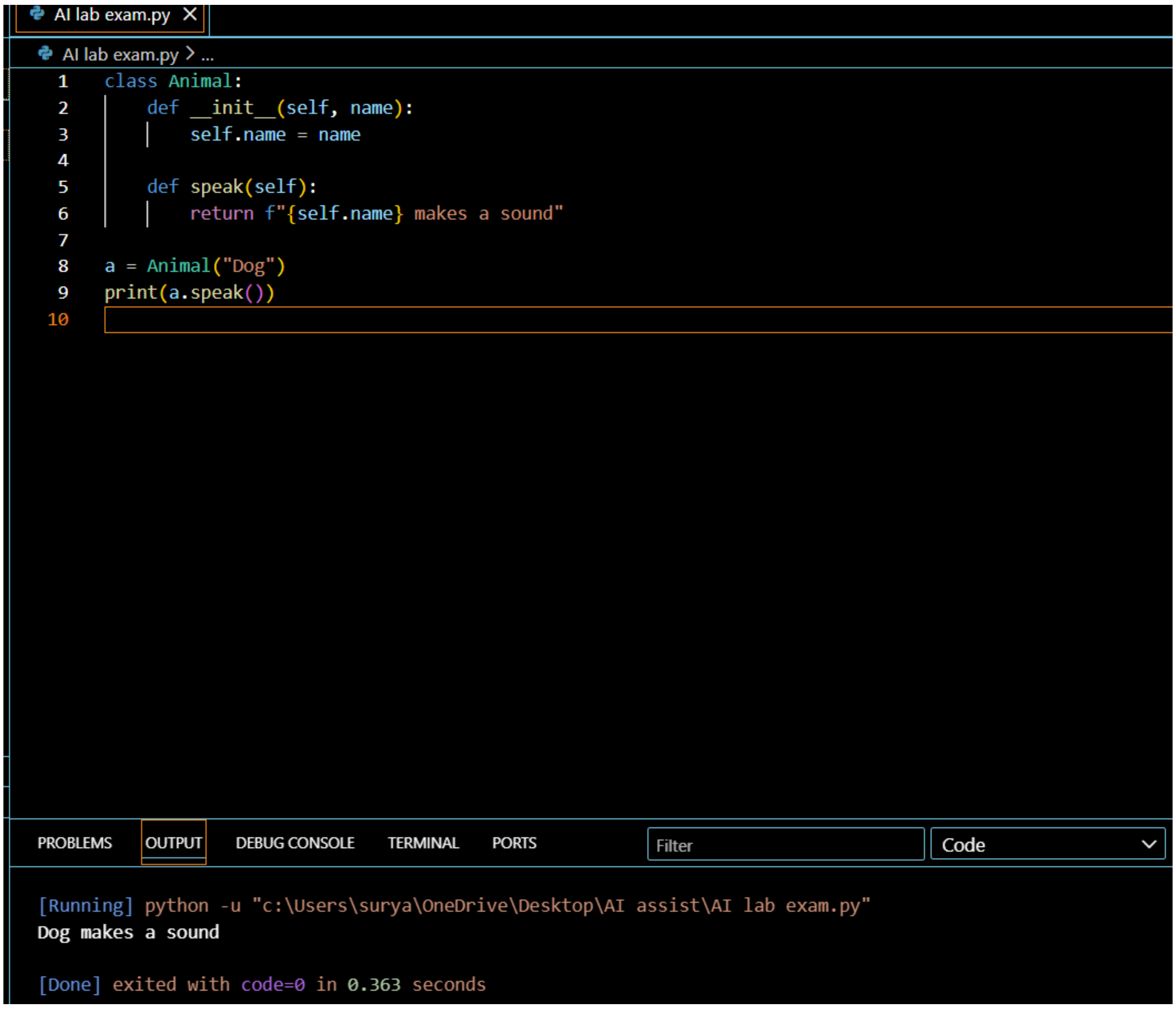
- Python uses __init__ while Kotlin uses a primary constructor.
- Kotlin requires explicit typing (String, Int), whereas Python is dynamically typed.
- Both use classes, objects, and methods similarly, but Kotlin enforces stronger type safety.

(b) Compare object-oriented features of both languages.

Prompt:

Compare the object-oriented programming features of Python and Kotlin using a simple class example in each language. Show code, output, and a short observation.

CODE :



```python
class Animal:
    def __init__(self, name):
        self.name = name

    def speak(self):
        return f"{self.name} makes a sound"

a = Animal("Dog")
print(a.speak())
```

```
[Running] python -u "c:\Users\surya\OneDrive\Desktop\AI assist\AI lab exam.py"
Dog makes a sound

[Done] exited with code=0 in 0.363 seconds
```

Observation :

- Kotlin offers a more structured, type-safe, and strict OOP model, while Python provides a more flexible and dynamic approach. Both support encapsulation, inheritance, and polymorphism, but Kotlin enforces correctness at compile time.