

# AI ASSISTED CODING

## LAB EXAM-4

NAME : K.SAI KARTHIK

HT.no : 2403A52043

BATCH : 03

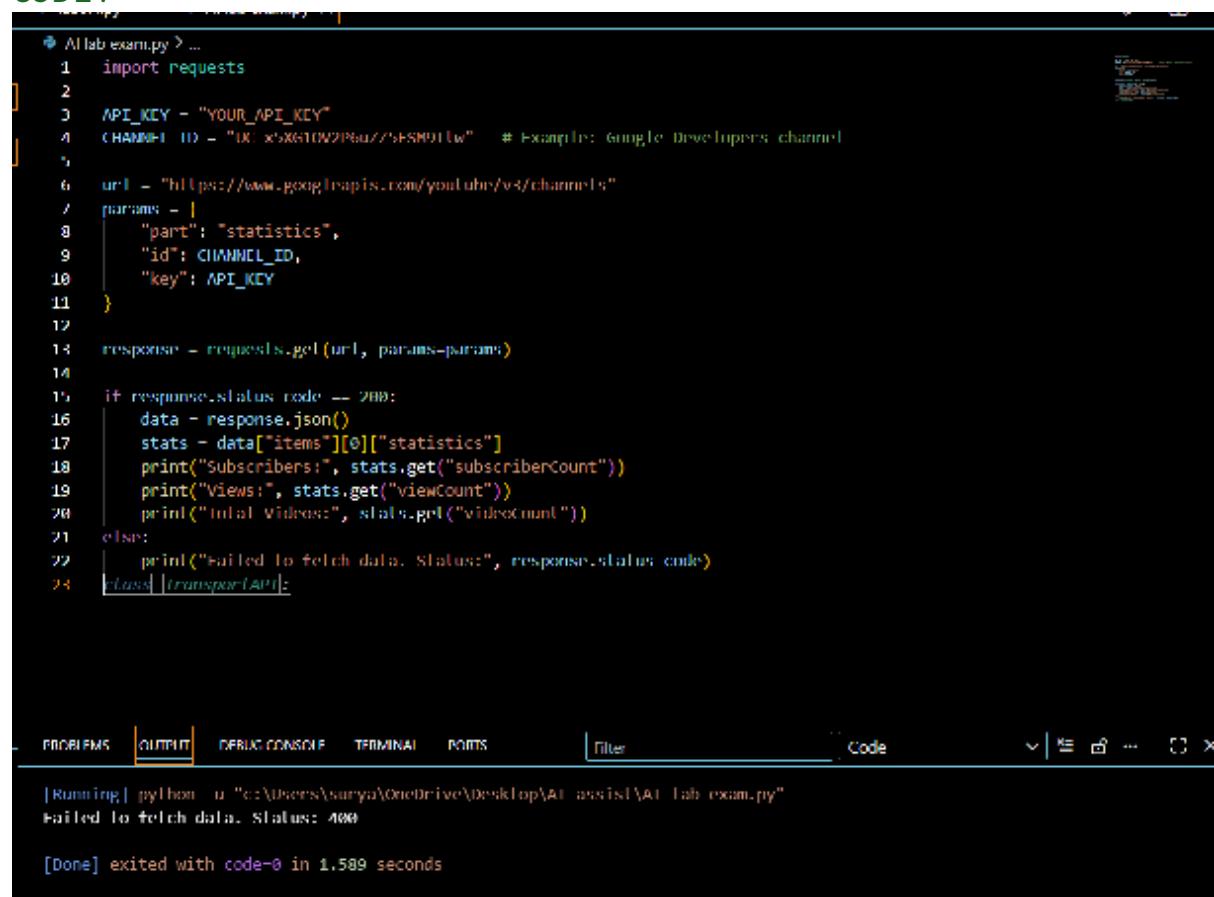
### Q1: (API Integration)

- (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 ...  
1 import requests  
2  
3 API_KEY = "YOUR_API_KEY"  
4 CHANNEL_ID = "UC_x5XGtOw2P6u7sESM9Ibw" # Example: Google Developers channel  
5  
6 url = "https://www.googleapis.com/youtube/v3/channels"  
7 params = {  
8     "part": "statistics",  
9     "id": CHANNEL_ID,  
10    "key": API_KEY  
11 }  
12  
13 response = requests.get(url, params=params)  
14  
15 if response.status_code == 200:  
16     data = response.json()  
17     stats = data["items"][0]["statistics"]  
18     print("Subscribers:", stats.get("subscriberCount"))  
19     print("Views:", stats.get("viewCount"))  
20     print("Total Videos:", stats.get("videoCount"))  
21 else:  
22     print("Failed to fetch data. Status:", response.status_code)  
23 class TransportAPI:
```

The screenshot shows a code editor window with the file name 'AI lab exam.py'. The code is a Python script that uses the YouTube Data API (v3) to fetch channel statistics. It imports the 'requests' module, defines API keys and channel IDs, constructs a URL, and sends a GET request to the API. It then checks the status code and prints the subscriber count, view count, and total video count if the request is successful. If it fails, it prints the error status code. The code editor interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, POINTS, and a search bar labeled 'Filter'. Below the code editor is a terminal window showing the execution of the script and its output.

[Running] python -u "c:\Users\surya\OneDrive\Desktop\AI assisted\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 :

```
AI lab exam.py
```

```
AI lab exam.py > ...
```

```
1 import requests
2
3 API_KEY = "YOUR_API_KEY"
4 CHANNEL_ID = "INVALID123" # Intentionally invalid for testing
5
6 url = "https://www.googleapis.com/youtube/v3/channels"
7 params = {
8     "part": "statistics",
9     "id": CHANNEL_ID,
10    "key": API_KEY
11 }
12
13 try:
14     response = requests.get(url, params=params)
15     response.raise_for_status()
16     data = response.json()
17
18     if "error" in data:
19         print("API Error:", data["error"]["message"])
20
21     elif not data.get("items"):
22         print("Invalid channel ID or no data found.")
23
24     else:
25         stats = data["items"][0].get("statistics", {})
26         print("Subscribers:", stats.get("subscriberCount", "N/A"))
27         print("Views:", stats.get("viewCount", "N/A"))
28         print("Total Videos:", stats.get("videoCount", "N/A"))
29
30 except requests.exceptions.RequestException as e:
31     print(e)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter Code

```
[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 > ...
1  class Student:
2      def __init__(self, name, marks):
3          self.name = name
4          self.marks = marks
5
6      def display(self):
7          return f"Name: {self.name}, Marks: {self.marks}"
8
9  # Create object and call method
10 s = Student("Rahul", 85)
11 print(s.display())
12
```

PROBLEMS    **OUTPUT**    DEBUG CONSOLE    TERMINAL    PORTS    Filter    Code

[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 :

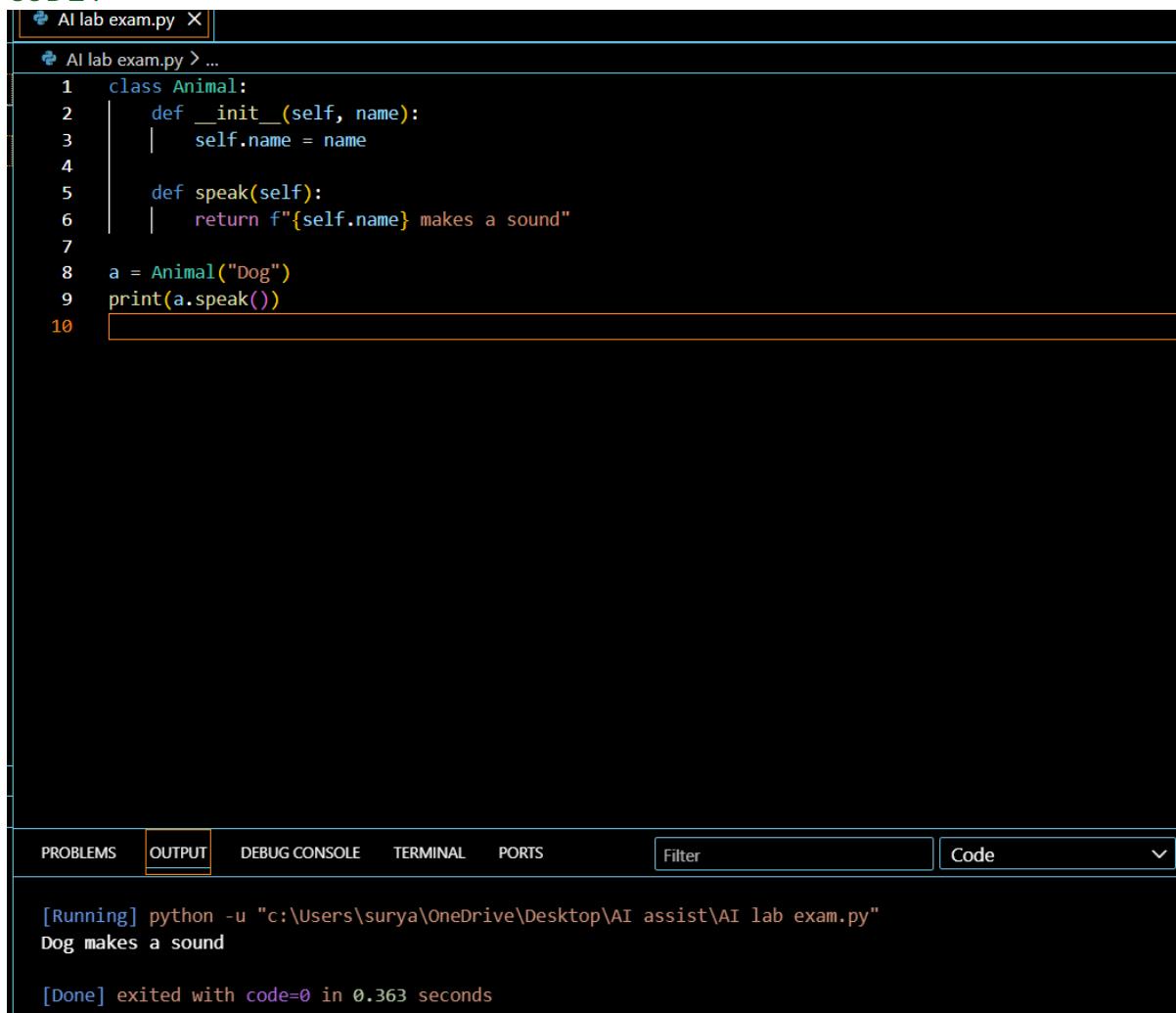
- Python uses `__init__` while Kotlin uses a primary constructor.
- Kotlin requires explicit typing (String, Int), whereas Python is dynamically typed.
- Both use classes, objects, and methods similarly, but Kotlin enforces stronger type safety.

### (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 :



The screenshot shows a code editor window with a dark theme. A file named "AI lab exam.py" is open, containing the following Python code:

```
1  class Animal:
2      def __init__(self, name):
3          self.name = name
4
5      def speak(self):
6          return f"{self.name} makes a sound"
7
8  a = Animal("Dog")
9  print(a.speak())
10
```

The code defines a class `Animal` with an `__init__` method that initializes the `name` attribute. It also has a `speak` method that returns a string indicating the animal makes a sound. An instance `a` is created with the name "Dog", and its `speak` method is called, printing the result. The code editor interface includes tabs for PROBLEMS, OUTPUT (which is selected), DEBUG CONSOLE, TERMINAL, and PORTS, along with a Filter and Code dropdown menu. Below the code area, the terminal output shows the execution of the script and its output.

[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.