AI ASSISTED CODING

EXAM -1

2403A51266

Batch – 12

1. Create a Python function that converts an amount from one currency to another using exchange rates stored in a dictionary. Use GitHub Copilot along with VS Code. Use Few shot prompting.
2. Write a Python program to extract all email addresses from a block of text using regular expressions. GitHub Copilot along with VS Code. Use zero shot prompting.
3. Given a list of movies with their genres, Write a Python function that recommends movies based on a user's preferred genre. Use the Cursor AI tool. Use few shot prompting.
4. Write Python code that reads a CSV file containing student names and marks in 3 subjects. Calculate the total and average marks for each student. Use the Cursor AI tool.

1. create a Python function that converts an amount from one currency to another using exchange stored in a dictionary. Use GitHub Copilot along with VS Code. Use Few shot prompting

✅ **Code:**

```
def convert_currency(amount, from_currency, to_currency, exchange_rates):
    if from_currency not in exchange_rates or to_currency not in exchange_rates:
        raise ValueError("Currency not found in exchange rates.")
    base_amount = amount / exchange_rates[from_currency]
    converted_amount = base_amount * exchange_rates[to_currency]
```

```
    return converted_amount
```

✅ **Output:**

100 USD in EUR: 92.0

100 EUR in JPY: 15923.91304347826

PROMPT:

The first two functions provide examples of simple, clear function definitions.

The third function header with a brief description sets the stage for Copilot to understand what kind of function you want.

When you write the above in VS Code with Copilot enabled, it will typically autocomplete convert_currency correctly based on the pattern.

2. Write a Python program to extract all email addresses from a block of text using regular expressions. GitHub Copilot along with VS Code. Use zero shot prompting.

✅ **Code**

```
import re


def extract_emails(text):
    pattern = r'[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}'
```

```
    return re.findall(pattern, text)


sample_text = """
Hello John, please contact jane.doe@example.com or support@company.co.uk
for further information. Also, cc it to admin123@my-site.org.
"""


emails = extract_emails(sample_text)
print(emails)
```

✅ Output

```
['jane.doe@example.com', 'support@company.co.uk', 'admin123@my-site.org']
```

PROMPT:

Write a Python function that extracts all email addresses from a given block of text. Use regular expressions to identify the email addresses, and return them as a list. Then, provide an example showing how the function works with a sample text containing multiple email addresses.

3. Given a list of movies with their genres, Write a Python function that recommends movies based on a user's preferred genre. Use the Cursor Al tool. Use few shot prompting.

✅ Code

```
def recommend_movies(genre, movie_list):
    return [title for title, g in movie_list if g == genre]

movies = [
    ("Inception", "Sci-Fi"),
    ("Die Hard", "Action"),
    ("The Matrix", "Sci-Fi"),
    ("John Wick", "Action"),
    ("Titanic", "Romance"),
    ("Interstellar", "Sci-Fi")
]

print(recommend_movies("Sci-Fi", movies))
print(recommend_movies("Action", movies))
print(recommend_movies("Romance", movies))
```

✅ Output

```
['Inception', 'The Matrix', 'Interstellar']
['Die Hard', 'John Wick']
['Titanic']
```

Prompt:

Given a list of movies where each movie is represented as a dictionary with keys "title" and "genre", write a Python function that recommends movies based on a user's preferred genre. Return a list of movie titles that match the user's genre.

4. Write Python code that reads a CSV file containing student names and marks in 3 subjects. Calculate the total and average marks for each student. Use the Cursor Al tool.

✅ **Sample CSV (students.csv)**

```
Name,Math,Science,English
Alice,85,90,88
Bob,78,82,80
Charlie,92,88,95
```

✅ **Python Code**

```
import csv


with open('students.csv', newline='') as csvfile:
    reader = csv.DictReader(csvfile)
    for row in reader:
        name = row['Name']
        marks = [int(row['Math']), int(row['Science']),
int(row['English'])]
```

```
    total = sum(marks)
    average = total / len(marks)
    print(f"{name}: Total = {total}, Average = {average:.2f}")
```

✅ **Output**

Alice: Total = 263, Average = 87.67

Bob: Total = 240, Average = 80.00

Charlie: Total = 275, Average = 91.67

Prompt:

Here’s a complete solution that:

Reads a CSV file containing student names and marks in 3 subjects

Calculates the total and average marks for each student

Prints the result

Includes a prompt suitable for GitHub Copilot (few-shot style)