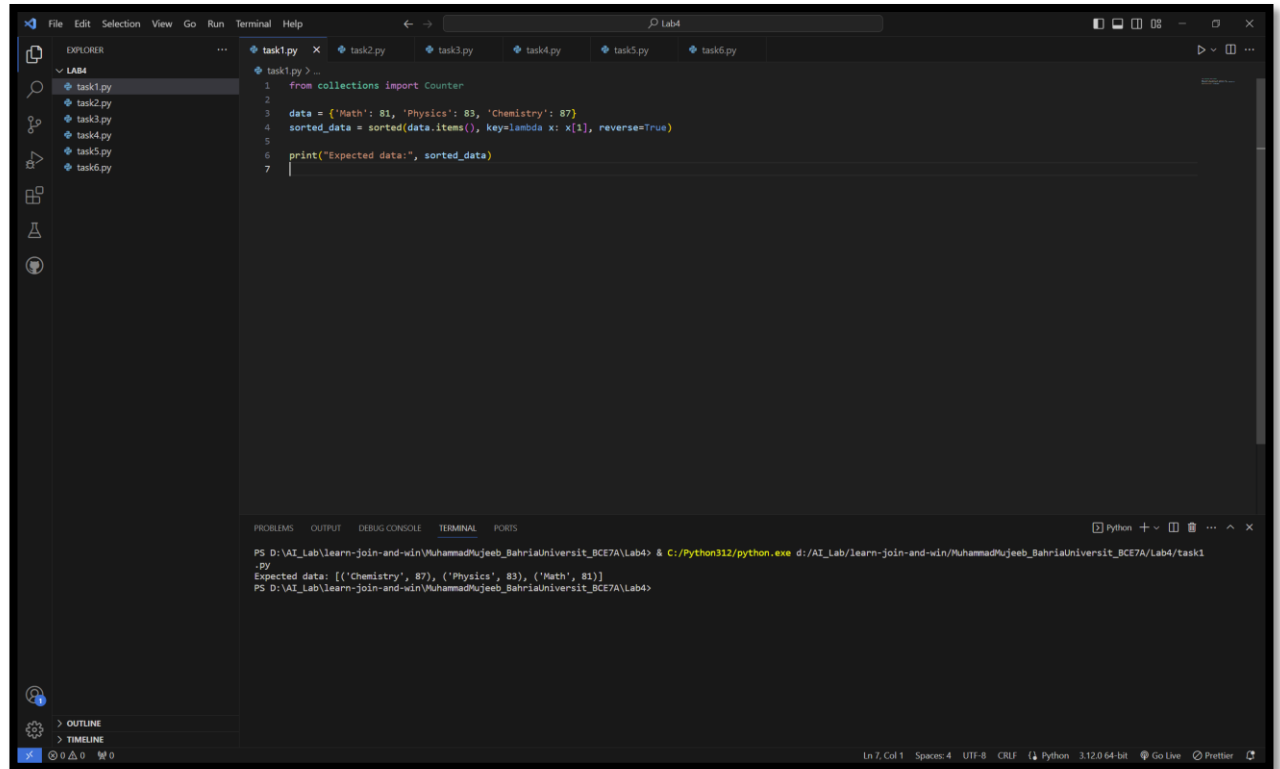


## Lab # 4

1. Write a Python program to sort Counter by value.

Sample data: {'Math':81, 'Physics':83, 'Chemistry':87}

Expected data: [('Chemistry', 87), ('Physics', 83), ('Math', 81)]



The screenshot shows a Visual Studio Code editor window with a Python file named `task1.py`. The code in the file is as follows:

```
1 from collections import Counter
2
3 data = {'Math': 81, 'Physics': 83, 'Chemistry': 87}
4 sorted_data = sorted(data.items(), key=lambda x: x[1], reverse=True)
5
6 print("Expected data:", sorted_data)
7
```

The terminal output at the bottom of the editor shows the command to run the script and the resulting output:

```
PS D:\VAI_Lab\learn-join-and-win\MuhammadUjeeb_BahriaUniversit_BCE7A\Lab4> & C:/Python312/python.exe d:/AI_Lab/learn-join-and-win/MuhammadUjeeb_BahriaUniversit_BCE7A/Lab4/task1
.py
Expected data: [('Chemistry', 87), ('Physics', 83), ('Math', 81)]
PS D:\VAI_Lab\learn-join-and-win\MuhammadUjeeb_BahriaUniversit_BCE7A\Lab4>
```

2. Write a Python program to store dictionary data in a JSON file.

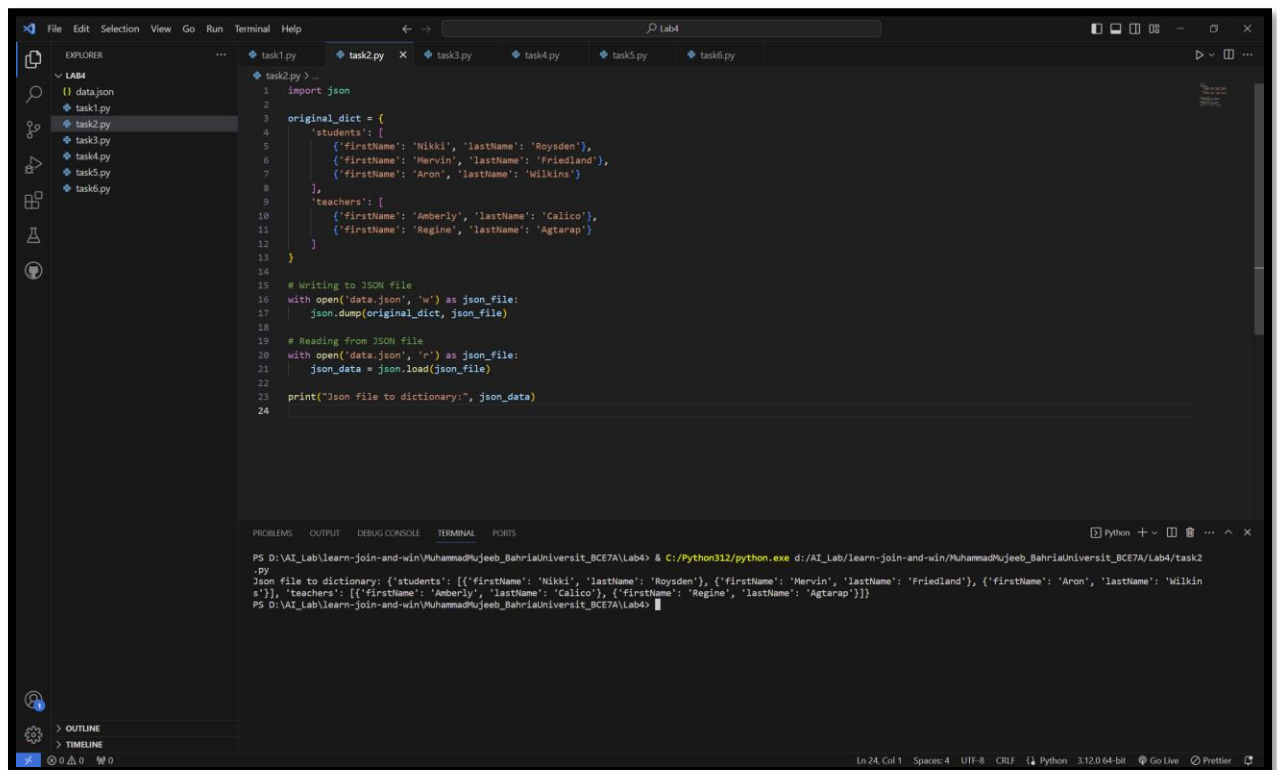
Original dictionary:

```
{'students': [{ 'firstName': 'Nikki', 'lastName': 'Roysden'}, { 'firstName': 'Mervin', 'lastName': 'Friedland'}, { 'firstName': 'Aron ', 'lastName': 'Wilkins'}], 'teachers': [{ 'firstName': 'Amberly', 'lastName': 'Calico'}, { 'firstName': 'Regine', 'lastName': 'Agtarap'}]}
```

<class 'dict'>

Json file to dictionary:

```
{'students': [{ 'firstName': 'Nikki', 'lastName': 'Roysden'}, { 'firstName': 'Mervin', 'lastName': 'Friedland'}, { 'firstName': 'Aron ', 'lastName': 'Wilkins'}], 'teachers': [{ 'firstName': 'Amberly', 'lastName': 'Calico'}, { 'firstName': 'Regine', 'lastName': 'Agtarap'}]}
```



```
1 import json
2
3 original_dict = {
4     'students': [
5         {'firstName': 'Nikki', 'lastName': 'Roysden'},
6         {'firstName': 'Mervin', 'lastName': 'Friedland'},
7         {'firstName': 'Aron ', 'lastName': 'Wilkins'}
8     ],
9     'teachers': [
10        {'firstName': 'Amberly', 'lastName': 'Calico'},
11        {'firstName': 'Regine', 'lastName': 'Agtarap'}
12    ]
13 }
14
15 # Writing to JSON file
16 with open('data.json', 'w') as json_file:
17     json.dump(original_dict, json_file)
18
19 # Reading from JSON file
20 with open('data.json', 'r') as json_file:
21     json_data = json.load(json_file)
22
23 print("Json file to dictionary:", json_data)
24
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\AI_Lab\learn-join-and-win\MuhammadMujeeb_BahriaIniversit_BCE7A\Lab4> & C:\Python312\python.exe d:\AI_Lab\learn-join-and-win\MuhammadMujeeb_BahriaIniversit_BCE7A\Lab4\task2
.py
Json file to dictionary: {'students': [{'firstName': 'Nikki', 'lastName': 'Roysden'}, {'firstName': 'Mervin', 'lastName': 'Friedland'}, {'firstName': 'Aron', 'lastName': 'Wilkin
s'}], 'teachers': [{'firstName': 'Amberly', 'lastName': 'Calico'}, {'firstName': 'Regine', 'lastName': 'Agtarap'}]}
```

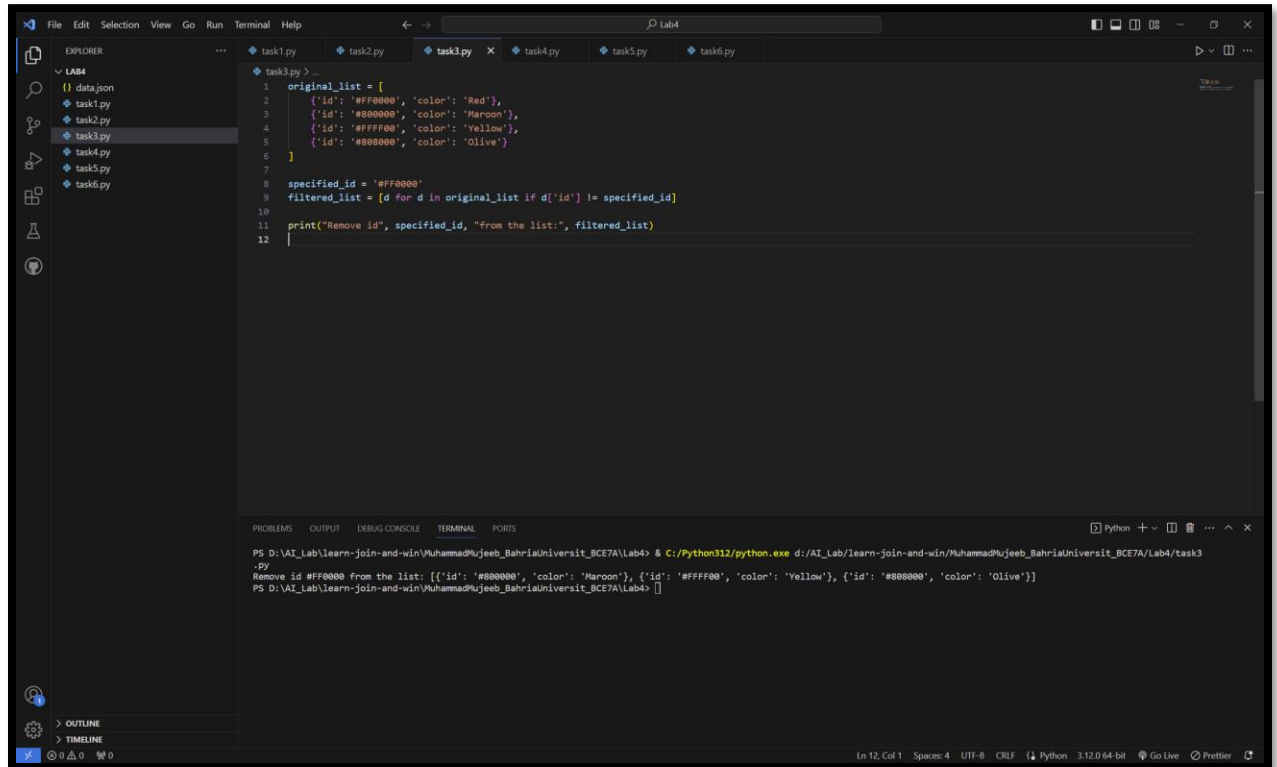
3. . Write a Python program to remove a specified dictionary from a given list.

Original list of dictionary:

```
[{'id': '#FF0000', 'color': 'Red'}, {'id': '#800000', 'color': 'Maroon'}, {'id': '#FFFF00', 'color': 'Yellow'}, {'id': '#808000', 'color': 'Olive'}]
```

Remove id #FF0000 from the said list of dictionary:

```
[{'id': '#800000', 'color': 'Maroon'}, {'id': '#FFFF00', 'color': 'Yellow'}, {'id': '#808000', 'color': 'Olive'}]
```

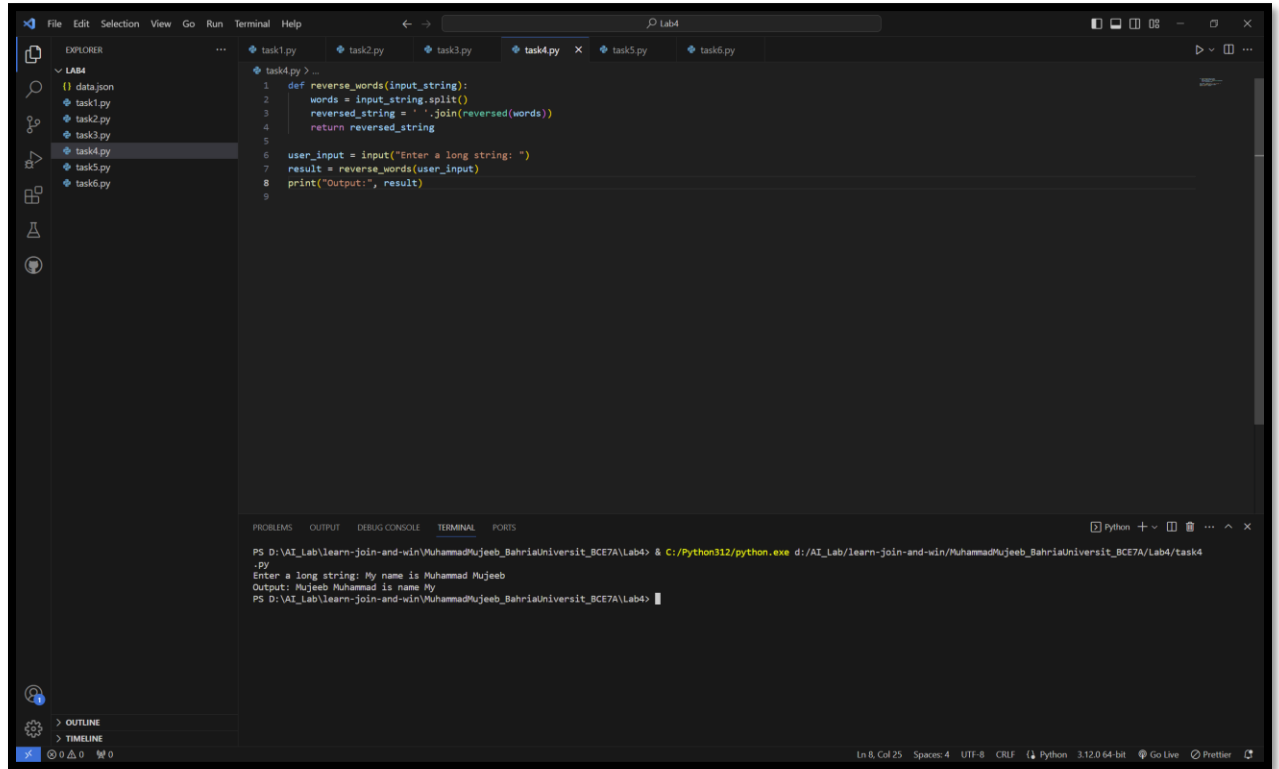


The screenshot shows a Visual Studio Code editor window with a Python file named 'task3.py'. The code defines an 'original\_list' of dictionaries, specifies an 'id' to be removed, and uses a list comprehension to create a 'filtered\_list' that excludes the specified dictionary. The terminal output shows the execution of the script, confirming the removal of the dictionary with id '#FF0000'.

```
task3.py > ...
1 original_list = [
2     {'id': '#FF0000', 'color': 'Red'},
3     {'id': '#800000', 'color': 'Maroon'},
4     {'id': '#FFFF00', 'color': 'Yellow'},
5     {'id': '#808000', 'color': 'Olive'}
6 ]
7
8 specified_id = '#FF0000'
9 filtered_list = [d for d in original_list if d['id'] != specified_id]
10
11 print("Remove id", specified_id, "from the list:", filtered_list)
12
```

```
PS D:\AI_Lab\learn-join-and-win\MuhammadUjeeb_BahriaUniversit_BCE7A\Lab4> & C:/Python312/python.exe d:/AI_Lab/learn-join-and-win/MuhammadUjeeb_BahriaUniversit_BCE7A/Lab4/task3
.PY
Remove id #FF0000 from the list: [{'id': '#800000', 'color': 'Maroon'}, {'id': '#FFFF00', 'color': 'Yellow'}, {'id': '#808000', 'color': 'Olive'}]
PS D:\AI_Lab\learn-join-and-win\MuhammadUjeeb_BahriaUniversit_BCE7A\Lab4>
```

4. Write a program (using functions!) that asks the user for a long string containing multiple words. Print back to the user the same string, except with the words in backwards order. For example, Input: I live in Pakistan. Output: Pakistan in live I.



The screenshot shows a Visual Studio Code editor window with a Python script named `task4.py` and its execution output in the terminal.

**Python Script (`task4.py`):**

```
1 def reverse_words(input_string):
2     words = input_string.split()
3     reversed_string = ' '.join(reversed(words))
4     return reversed_string
5
6 user_input = input("Enter a long string: ")
7 result = reverse_words(user_input)
8 print("Output:", result)
9
```

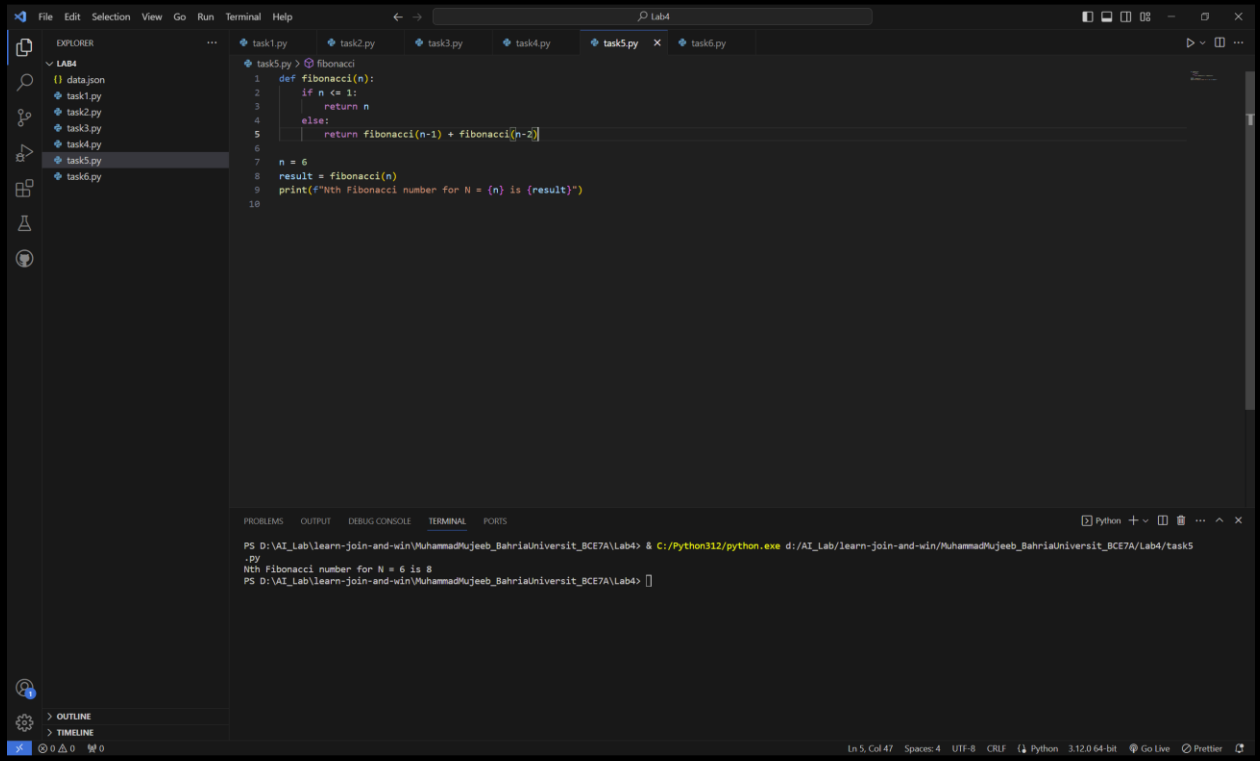
**Terminal Output:**

```
PS D:\AI_Lab\learn-join-and-win\MuhammadMujeeb_BahriaUniversity_BCE7A\Lab4> & C:/Python312/python.exe d:/AI_Lab/learn-join-and-win/MuhammadMujeeb_BahriaUniversity_BCE7A/Lab4/task4
.py
Enter a long string: My name is Muhammad Mujeeb
Output: Mujeeb Muhammad is name My
PS D:\AI_Lab\learn-join-and-win\MuhammadMujeeb_BahriaUniversity_BCE7A\Lab4>
```

The terminal output demonstrates the program's functionality: it prompts the user to enter a long string, processes it using the `reverse_words` function, and prints the result with the words in reverse order.

5. Write a recursive function to compute Nth Fibonacci number. Test and trace for  $N = 6$  is 8. We remember that a Fibonacci number can be recursively defined as:

$$F(n) = F(n - 1) + F(n - 2) \text{ for } n \geq 2, \text{ where } F(0) = 0, F(1) = 1.$$



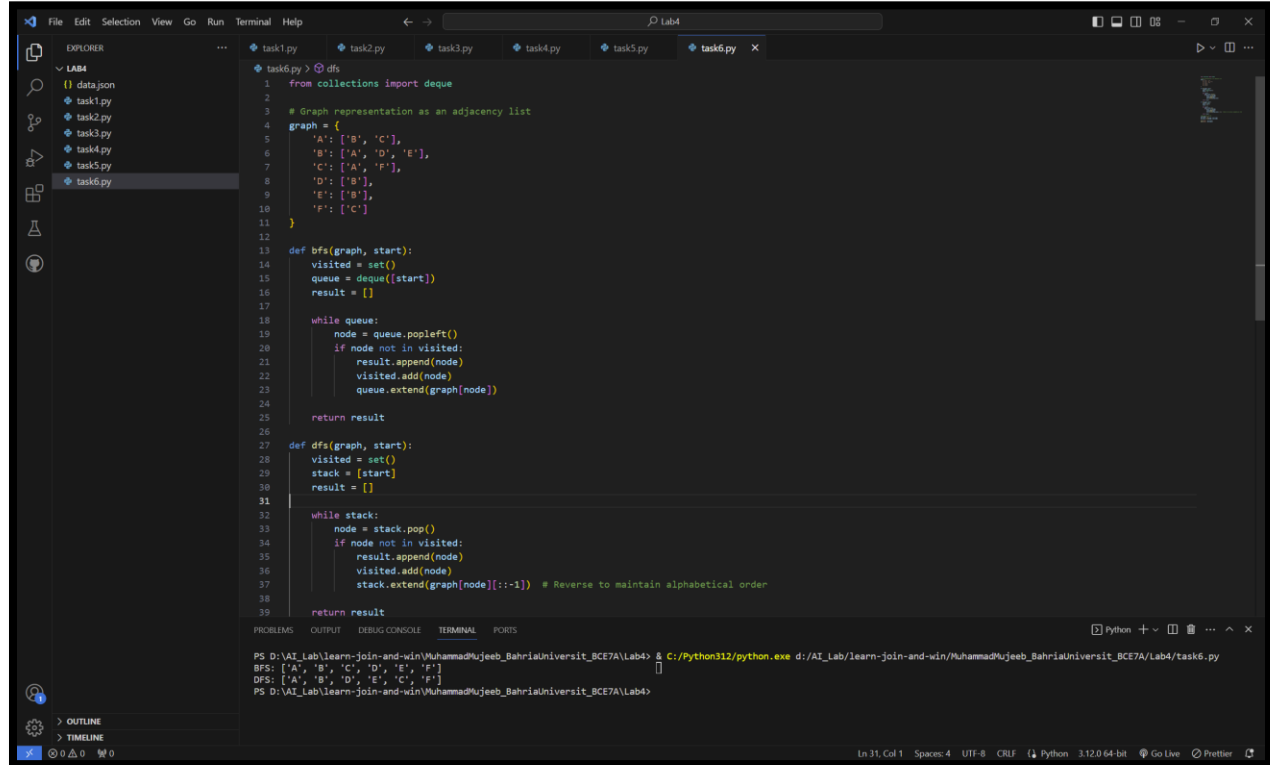
The screenshot shows a Visual Studio Code editor with a Python file named `task5.py` open. The code defines a recursive function `fibonacci(n)` and tests it for  $N = 6$ . The terminal output shows the command to run the script and the resulting output: "Nth Fibonacci number for N = 6 is 8".

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     else:
5         return fibonacci(n-1) + fibonacci(n-2)
6
7 n = 6
8 result = fibonacci(n)
9 print(f"Nth Fibonacci number for N = {n} is {result}")
10
```

Terminal Output:

```
PS D:\AI_Lab\learn-join-and-win\MuhammadMujeeb_BahriaUniversity_BCE7A\Lab4> & C:/Python312/python.exe d:/AI_Lab/learn-join-and-win/MuhammadMujeeb_BahriaUniversity_BCE7A/Lab4/task5.py
Nth Fibonacci number for N = 6 is 8
PS D:\AI_Lab\learn-join-and-win\MuhammadMujeeb_BahriaUniversity_BCE7A\Lab4>
```

6. Implement BFS & DFS in python as describes in the class.



```
1 from collections import deque
2
3 # Graph representation as an adjacency list
4 graph = {
5     'A': ['B', 'C'],
6     'B': ['A', 'D', 'E'],
7     'C': ['A', 'F'],
8     'D': ['B'],
9     'E': ['B'],
10    'F': ['C']
11 }
12
13 def bfs(graph, start):
14     visited = set()
15     queue = deque([start])
16     result = []
17
18     while queue:
19         node = queue.popleft()
20         if node not in visited:
21             result.append(node)
22             visited.add(node)
23             queue.extend(graph[node])
24
25     return result
26
27 def dfs(graph, start):
28     visited = set()
29     stack = [start]
30     result = []
31
32     while stack:
33         node = stack.pop()
34         if node not in visited:
35             result.append(node)
36             visited.add(node)
37             stack.extend(graph[node][::-1]) # Reverse to maintain alphabetical order
38
39     return result
40
41 if __name__ == '__main__':
42     start = 'A'
43     print("BFS: ", bfs(graph, start))
44     print("DFS: ", dfs(graph, start))
```

PS D:\AI\_Lab\learn-join-and-win\MuhammadUjeeb\_BahriaUniversity\_BCE7A\Lab4> & C:/Python312/python.exe d:/AI\_Lab/learn-join-and-win/MuhammadUjeeb\_BahriaUniversity\_BCE7A/Lab4/task6.py

BFS: ['A', 'B', 'C', 'D', 'E', 'F']  
DFS: ['A', 'B', 'D', 'E', 'C', 'F']

PS D:\AI\_Lab\learn-join-and-win\MuhammadUjeeb\_BahriaUniversity\_BCE7A\Lab4>