

Cycle 1

0	1	3			
A	B	C	D	E	
0	1	3	2		
A	B	C	D	E	
0	1	3	2 5		
A	B	C	D	E	
0	1	3	2 5		
A	B	C	D	E	

Cycle 2

0	1	3	2	5
A	B	C	D	E

Chat gpt code

```
def networkDelayTime(times, n, k):

    # Step 2: Initialize distance array
    distances = [float('inf')] * n
    distances[k-1] = 0 # Set distance to source node as 0

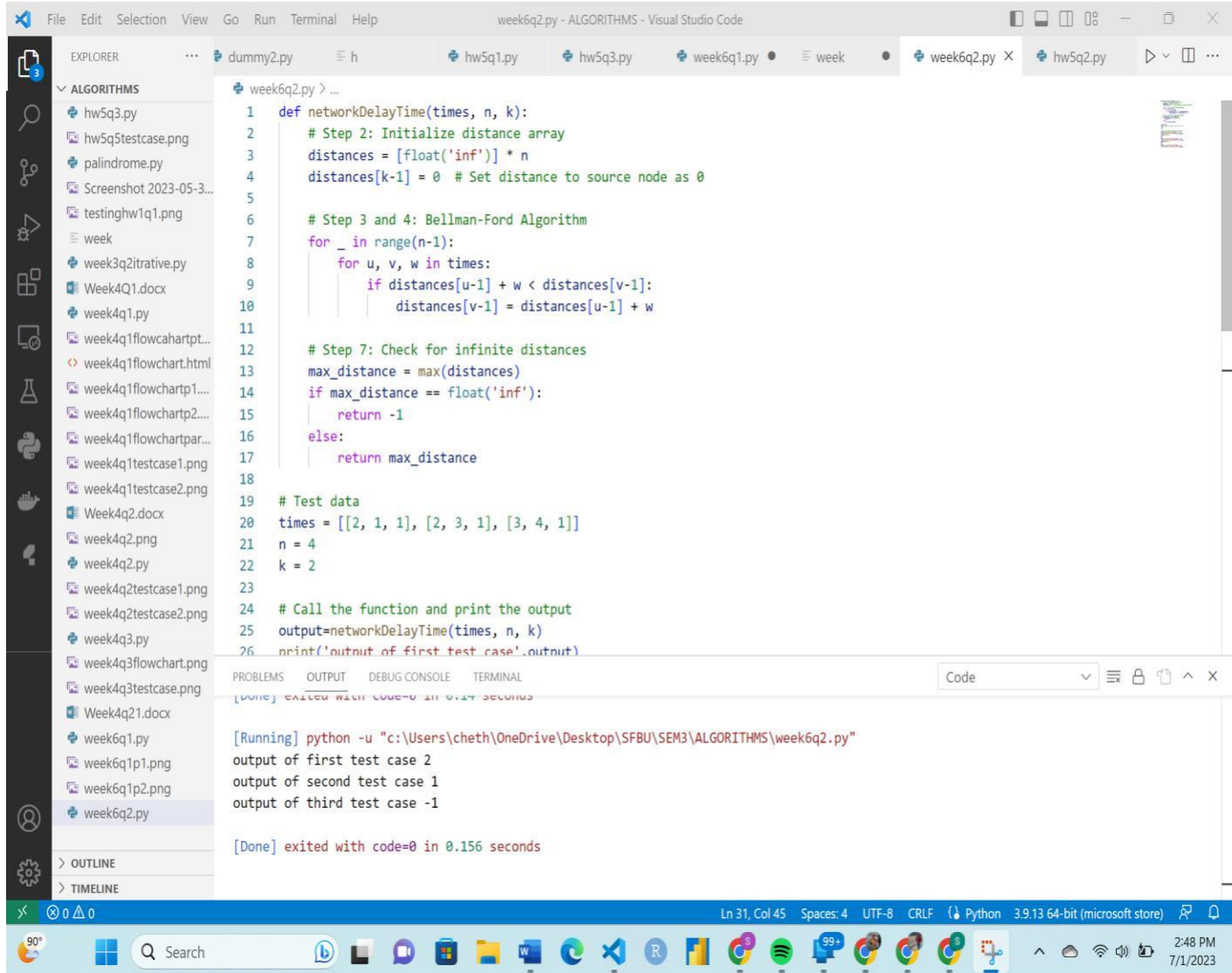
    # Step 3 and 4: Bellman-Ford Algorithm
    for _ in range(n-1):
        for u, v, w in times:
            if distances[u-1] + w < distances[v-1]:
                distances[v-1] = distances[u-1] + w

    # Step 7: Check for infinite distances
    max_distance = max(distances)
    if max_distance == float('inf'):
        return -1
    else:
        return max_distance

# Test data
times = [[2, 1, 1], [2, 3, 1], [3, 4, 1]]
n = 4
k = 2

# Call the function and print the output
print(networkDelayTime(times, n, k)) # Output: 2
```

Test case



The image shows a Visual Studio Code window with a Python file named `week6q2.py` open. The file contains a function `networkDelayTime` that implements the Bellman-Ford algorithm. The function takes a list of edges `times`, the number of nodes `n`, and the number of intermediate stops `k`. It initializes a distance array, iterates through the edges `k` times, and returns the maximum distance or -1 if the destination is unreachable.

```
1 def networkDelayTime(times, n, k):
2     # Step 2: Initialize distance array
3     distances = [float('inf')] * n
4     distances[k-1] = 0 # Set distance to source node as 0
5
6     # Step 3 and 4: Bellman-Ford Algorithm
7     for _ in range(n-1):
8         for u, v, w in times:
9             if distances[u-1] + w < distances[v-1]:
10                 distances[v-1] = distances[u-1] + w
11
12     # Step 7: Check for infinite distances
13     max_distance = max(distances)
14     if max_distance == float('inf'):
15         return -1
16     else:
17         return max_distance
18
19 # Test data
20 times = [[2, 1, 1], [2, 3, 1], [3, 4, 1]]
21 n = 4
22 k = 2
23
24 # Call the function and print the output
25 output=networkDelayTime(times, n, k)
26 print('output of first test case'.output)
```

The terminal output shows the execution of the script:

```
[Running] python -u "c:\Users\cheth\OneDrive\Desktop\SFBUS\SEM3\ALGORITHMS\week6q2.py"
output of first test case 2
output of second test case 1
output of third test case -1

[Done] exited with code=0 in 0.156 seconds
```

The status bar at the bottom indicates the file is at line 31, column 45, with 4 spaces, UTF-8 encoding, CRLF line endings, and Python 3.9.13 64-bit (microsoft store) interpreter.

