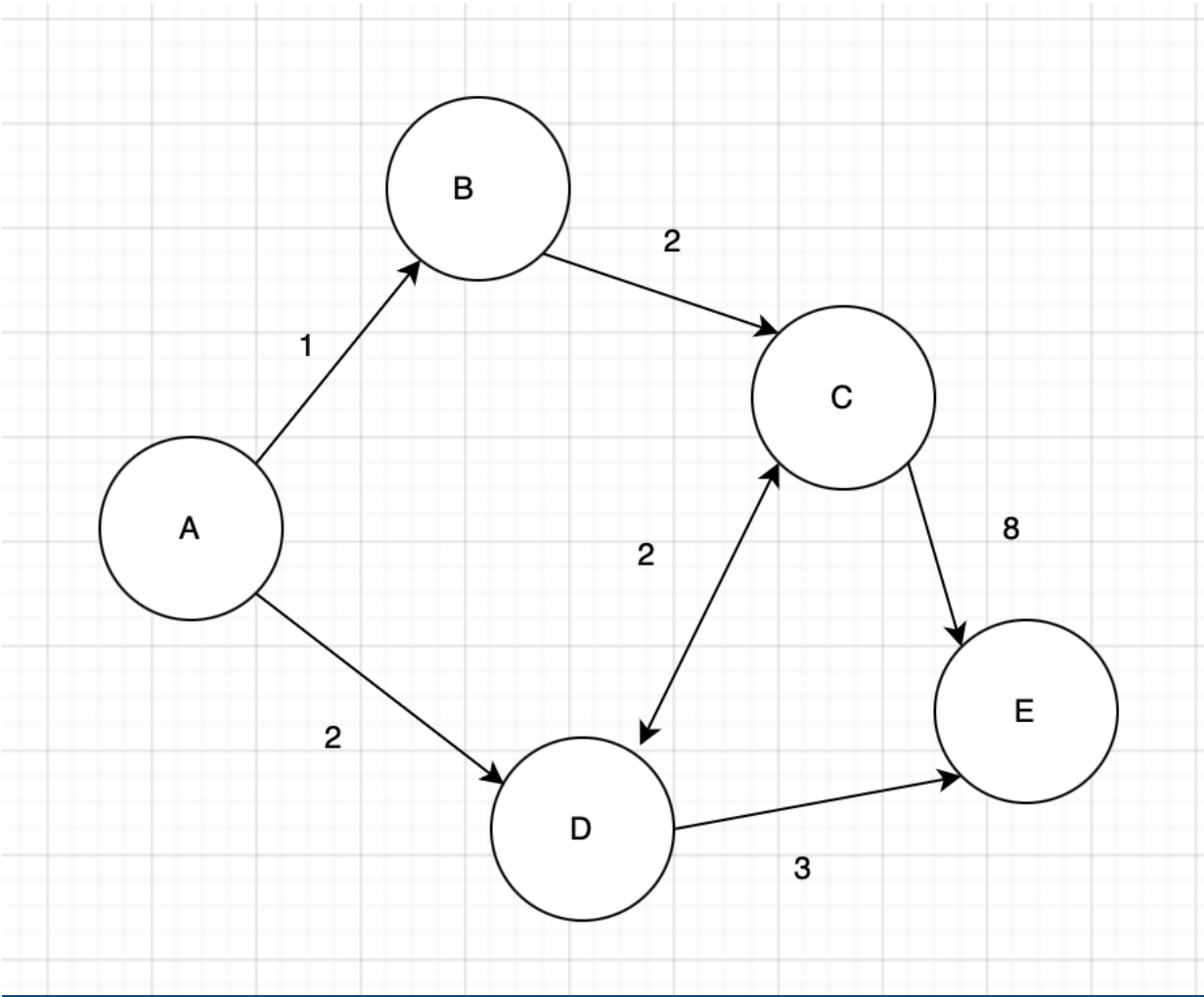


Week6_Q1



Intial	NEXT NODE A	NEXT NODE B	NEXT NODE D	NEXT NODE D
A=∞	A=0	A=0	A=0	A=0
B=∞	B=∞	B=1	B=1	B=1
C=∞	C=∞	C=∞	C=3	C=3
D=∞	D=∞	D=2	D=2	D=2
E=∞	E=∞	E=∞	E=∞	E=5

```
import heapq;
```

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

```

# Step 1: Create the graph
graph = {}
for u, v, w in times:
    if u not in graph:
        graph[u] = []
    graph[u].append((v, w))

# Step 2: Perform Dijkstra's Algorithm
distances = [float('inf')] * (n + 1)
distances[k] = 0
heap = [(0, k)]

while heap:
    distance, node = heapq.heappop(heap)

    # If the current distance is greater than the stored distance, skip the node
    if distance > distances[node]:
        continue

    if node in graph:
        for neighbor, weight in graph[node]:
            new_distance = distance + weight
            if new_distance < distances[neighbor]:
                distances[neighbor] = new_distance
                heapq.heappush(heap, (new_distance, neighbor))

# Step 3: Find the maximum delay time
max_delay = max(distances[1:])

if max_delay == float('inf'):
    return -1
else:
    return max_delay

# Test the code with the given input
times = [[2, 1, 1], [2, 3, 1], [3, 4, 1]]
n = 4
k = 2

output = networkDelayTime(times, n, k)
print(output)

```

The screenshot shows a VS Code editor window with a Python file named `week6_Q1.py`. The code implements Dijkstra's algorithm to find the shortest path in a graph. The terminal shows the command to run the script, which executes successfully.

Code Editor Content:

```

1  import heapq
2
3
4  def networkDelayTime(times, n, k):
5      # Step 1: Create the graph
6      graph = {}
7      for u, v, w in times:
8          if u not in graph:
9              graph[u] = []
10             graph[u].append((v, w))
11
12     # Step 2: Perform Dijkstra's Algorithm
13     distances = [float('inf')] * (n + 1)
14     distances[k] = 0
15     heap = [(0, k)]
16

```

Terminal Content:

```

/usr/local/bin/python3 /Users/rakesh_kasha/Desktop/week6_Q1.py
(base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ % /usr/local/bin/python3 /Users/rakesh_kasha/Desktop/week6_Q1.py
2
1
-1
(base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ %

```

The screenshot shows a Mac desktop with a code editor window titled "week6_Q1.py". The code in the editor is as follows:

```

40 # Test the code with the given input
41 times = [[2, 1, 1], [2, 3, 1], [3, 4, 1]]
42 n = 4
43 k = 2
44
45 output = networkDelayTime(times, n, k)
46 print(output)
47
48 times1 = [[1,2,1]]
49 n = 2
50 k = 1
51 output = networkDelayTime(times1, n, k)
52 print(output)
53
54 times2 = [[1,2,1]]
55 n = 2

```

Below the code editor, there is a "TERMINAL" tab. The terminal shows the following commands and output:

```

/usr/local/bin/python3 /Users/rakesh_kasha/Desktop/week6_Q1.py
(base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ % /usr/local/bin/python3 /Users/rakesh_kasha/Desktop/week6_Q1.py
2
1
-1
(base) rakesh_kasha@Rakesh-kashas-MacBook-Pro ~ %

```

The Mac's top status bar shows the time as "Sat Jul 1 5:20 PM". The bottom of the screen shows a dock with various application icons and a system status bar with network, volume, and battery indicators.