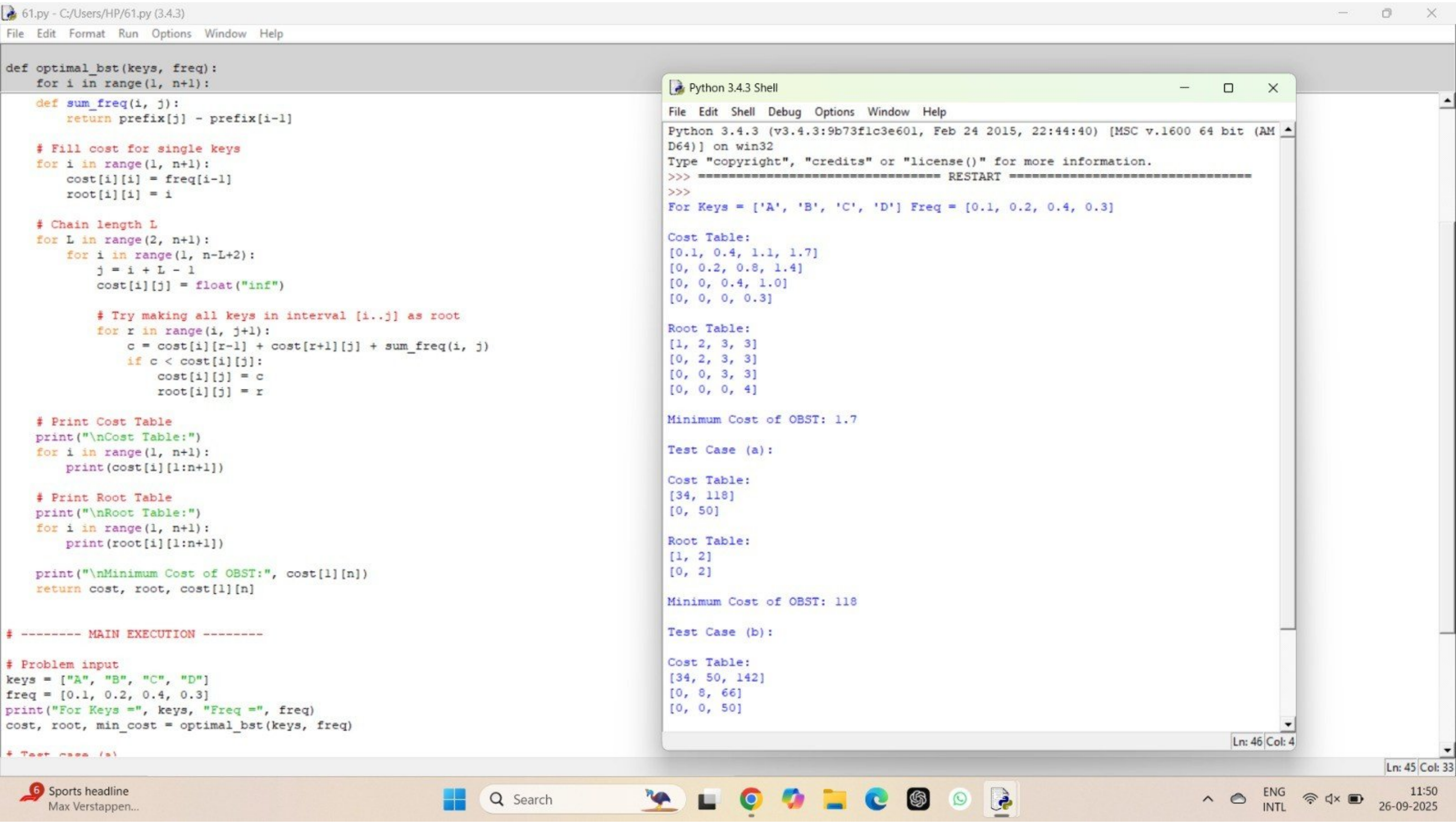
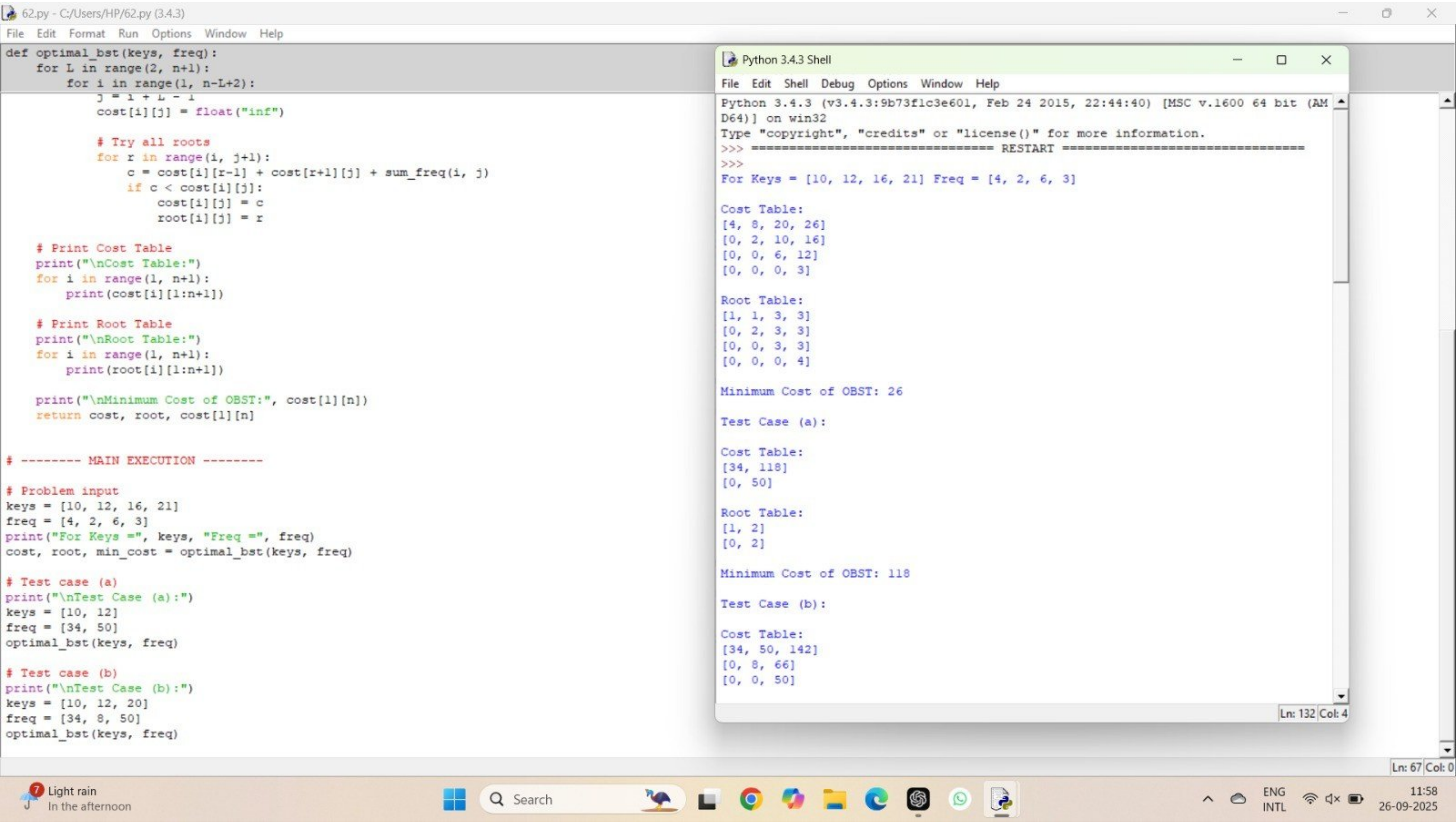

61.py - C:/Users/HP/61.py (3.4.3)
File Edit Format Run Options Window Help
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Ln: 46 Col: 4
Ln: 45 Col: 33
Sports headline
Max Verstappen...
Search
ENG
INTL
11:50
26-09-2025

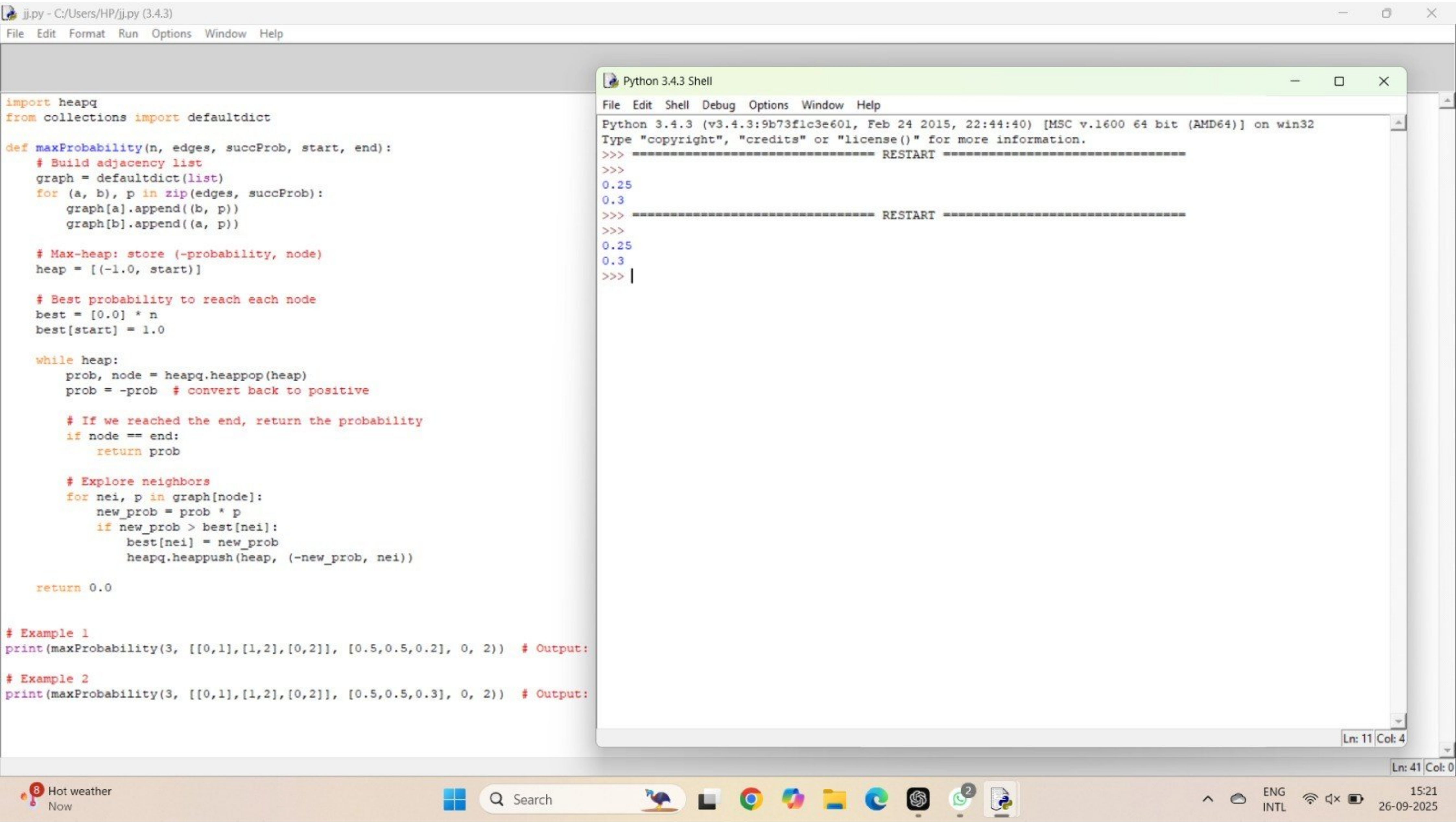
```python
def optimal_bst(keys, freq):
    for i in range(1, n+1):
    def sum_freq(i, j):
        return prefix[j] - prefix[i-1]

    # Fill cost for single keys
    for i in range(1, n+1):
        cost[i][i] = freq[i-1]
        root[i][i] = i

    # Chain length L
    for L in range(2, n+1):
        for i in range(1, n-L+2):
            j = i + L - 1
            cost[i][j] = float("inf")

            # Try making all keys in interval [i..j] as root
            for r in range(i, j+1):
                c = cost[i][r-1] + cost[r+1][j] + sum_freq(i, j)
                if c < cost[i][j]:
                    cost[i][j] = c
                    root[i][j] = r

    # Print Cost Table
    print("\nCost Table:")
    for i in range(1, n+1):
        print(cost[i][1:n+1])

    # Print Root Table
    print("\nRoot Table:")
    for i in range(1, n+1):
        print(root[i][1:n+1])

    print("\nMinimum Cost of OBST:", cost[1][n])
    return cost, root, cost[1][n]


# -------- MAIN EXECUTION --------

# Problem input
keys = ["A", "B", "C", "D"]
freq = [0.1, 0.2, 0.4, 0.3]
print("For Keys =", keys, "Freq =", freq)
cost, root, min_cost = optimal_bst(keys, freq)
```

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
For Keys = ['A', 'B', 'C', 'D'] Freq = [0.1, 0.2, 0.4, 0.3]

Cost Table:
[0.1, 0.4, 1.1, 1.7]
[0, 0.2, 0.8, 1.4]
[0, 0, 0.4, 1.0]
[0, 0, 0, 0.3]

Root Table:
[1, 2, 3, 3]
[0, 2, 3, 3]
[0, 0, 3, 3]
[0, 0, 0, 4]

Minimum Cost of OBST: 1.7

Test Case (a):

Cost Table:
[34, 118]
[0, 50]

Root Table:
[1, 2]
[0, 2]

Minimum Cost of OBST: 118

Test Case (b):

Cost Table:
[34, 50, 142]
[0, 8, 66]
[0, 0, 50]
```


62.py - C:/Users/HP/62.py (3.4.3)
File Edit Format Run Options Window Help
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Ln: 132 Col: 4
Ln: 67 Col: 0
Light rain
In the afternoon
Search
ENG
INTL
11:58
26-09-2025

```python
def optimal_bst(keys, freq):
    for L in range(2, n+1):
        for i in range(1, n-L+2):
            j = i + L - 1
            cost[i][j] = float("inf")

            # Try all roots
            for r in range(i, j+1):
                c = cost[i][r-1] + cost[r+1][j] + sum_freq(i, j)
                if c < cost[i][j]:
                    cost[i][j] = c
                    root[i][j] = r

    # Print Cost Table
    print("\nCost Table:")
    for i in range(1, n+1):
        print(cost[i][1:n+1])

    # Print Root Table
    print("\nRoot Table:")
    for i in range(1, n+1):
        print(root[i][1:n+1])

    print("\nMinimum Cost of OBST:", cost[1][n])
    return cost, root, cost[1][n]


# -------- MAIN EXECUTION --------

# Problem input
keys = [10, 12, 16, 21]
freq = [4, 2, 6, 3]
print("For Keys =", keys, "Freq =", freq)
cost, root, min_cost = optimal_bst(keys, freq)

# Test case (a)
print("\nTest Case (a):")
keys = [10, 12]
freq = [34, 50]
optimal_bst(keys, freq)

# Test case (b)
print("\nTest Case (b):")
keys = [10, 12, 20]
freq = [34, 8, 50]
optimal_bst(keys, freq)
```

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
For Keys = [10, 12, 16, 21] Freq = [4, 2, 6, 3]

Cost Table:
[4, 8, 20, 26]
[0, 2, 10, 16]
[0, 0, 6, 12]
[0, 0, 0, 3]

Root Table:
[1, 1, 3, 3]
[0, 2, 3, 3]
[0, 0, 3, 3]
[0, 0, 0, 4]

Minimum Cost of OBST: 26

Test Case (a):

Cost Table:
[34, 118]
[0, 50]

Root Table:
[1, 2]
[0, 2]

Minimum Cost of OBST: 118

Test Case (b):

Cost Table:
[34, 50, 142]
[0, 8, 66]
[0, 0, 50]
```



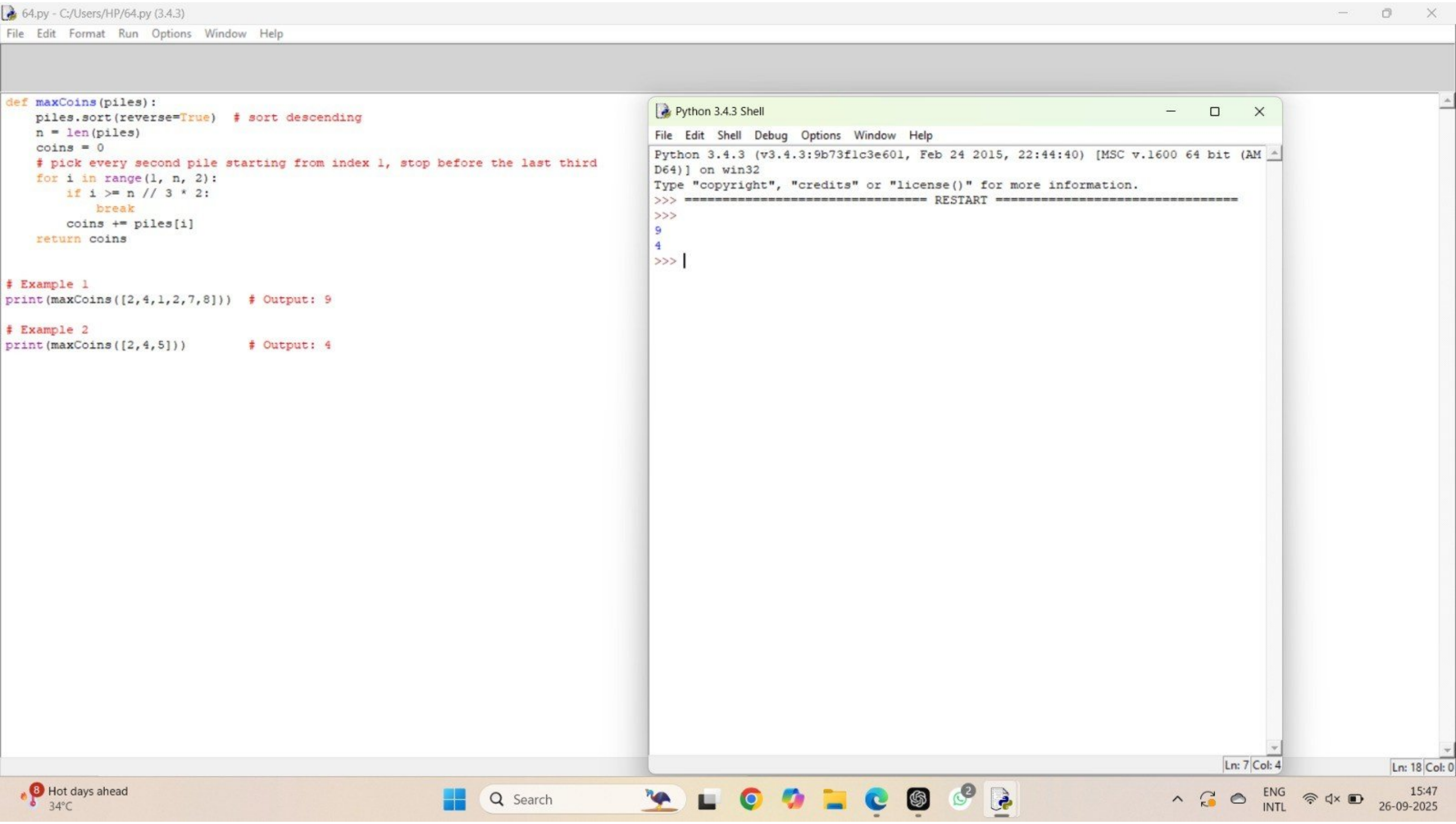
jj.py - C:/Users/HP/jj.py (3.4.3)

```python
import heapq
from collections import defaultdict

def maxProbability(n, edges, succProb, start, end):
    # Build adjacency list
    graph = defaultdict(list)
    for (a, b), p in zip(edges, succProb):
        graph[a].append((b, p))
        graph[b].append((a, p))

    # Max-heap: store (-probability, node)
    heap = [(-1.0, start)]

    # Best probability to reach each node
    best = [0.0] * n
    best[start] = 1.0

    while heap:
        prob, node = heapq.heappop(heap)
        prob = -prob  # convert back to positive

        # If we reached the end, return the probability
        if node == end:
            return prob

        # Explore neighbors
        for nei, p in graph[node]:
            new_prob = prob * p
            if new_prob > best[nei]:
                best[nei] = new_prob
                heapq.heappush(heap, (-new_prob, nei))

    return 0.0


# Example 1
print(maxProbability(3, [[0,1],[1,2],[0,2]], [0.5,0.5,0.2], 0, 2))  # Output:

# Example 2
print(maxProbability(3, [[0,1],[1,2],[0,2]], [0.5,0.5,0.3], 0, 2))  # Output:
```

Python 3.4.3 Shell

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
0.25
0.3
>>> ================================ RESTART ================================
>>>
0.25
0.3
>>>
```

File Edit Format Run Options Window Help
File Edit Shell Debug Options Window Help
Ln: 11 Col: 4
Ln: 41 Col: 0
Hot weather
Now
Search
ENG
INTL
15:21
26-09-2025

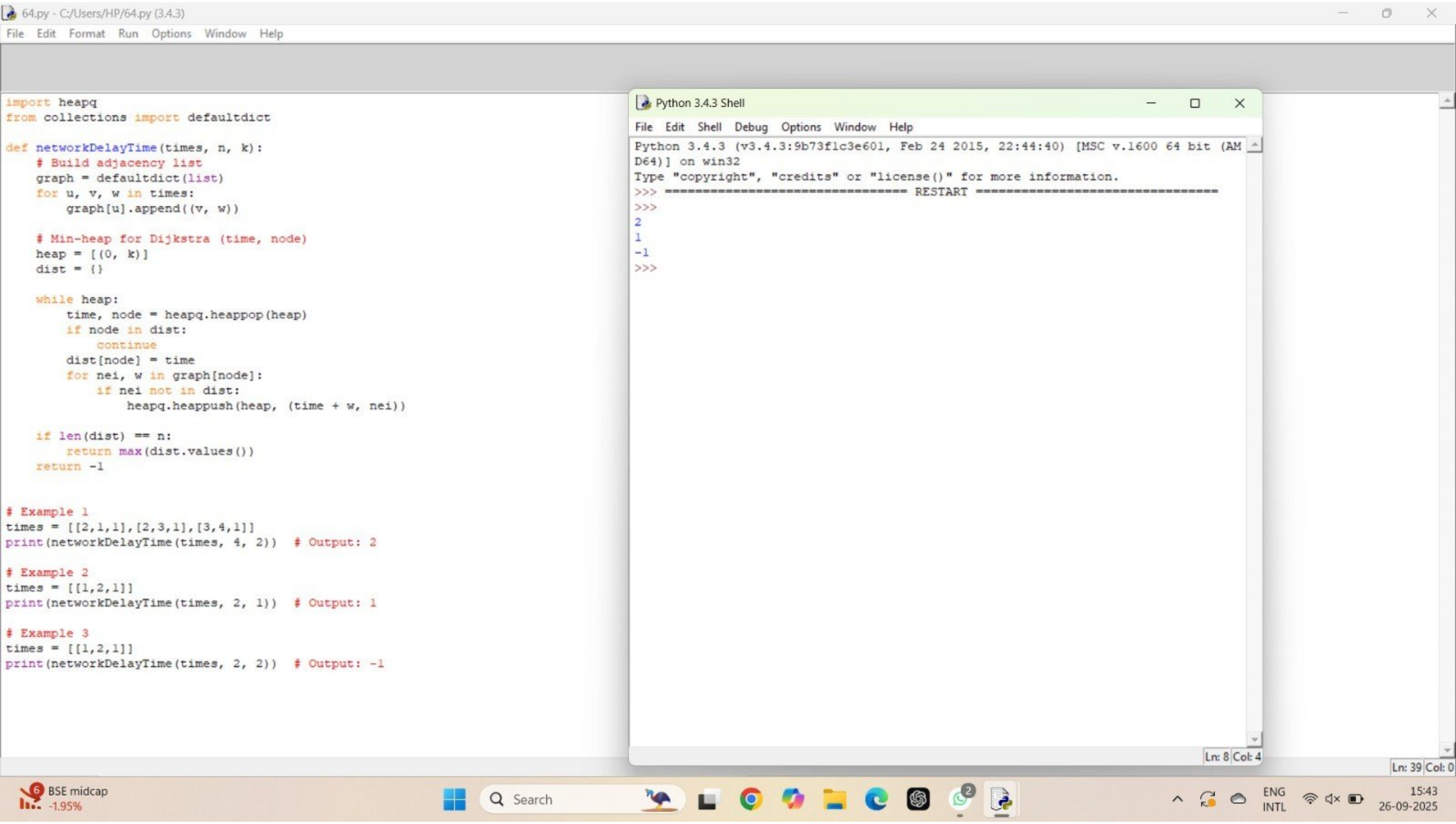


64.py - C:/Users/HP/64.py (3.4.3)

```python
def maxCoins(piles):
    piles.sort(reverse=True)  # sort descending
    n = len(piles)
    coins = 0
    # pick every second pile starting from index 1, stop before the last third
    for i in range(1, n, 2):
        if i >= n // 3 * 2:
            break
        coins += piles[i]
    return coins


# Example 1
print(maxCoins([2,4,1,2,7,8]))  # Output: 9

# Example 2
print(maxCoins([2,4,5]))        # Output: 4
```

Python 3.4.3 Shell

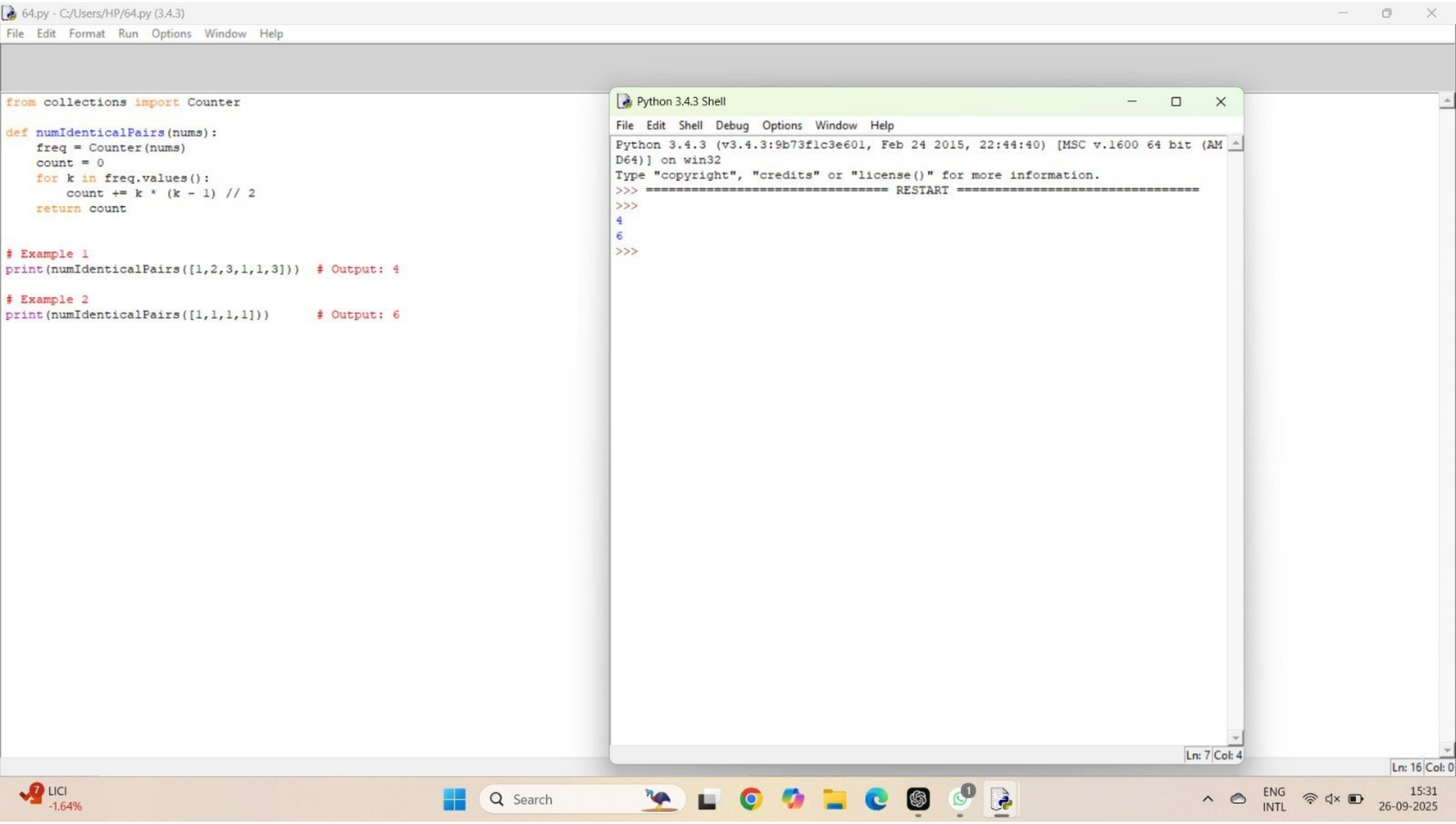
```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
9
4
>>> 
```



64.py - C:/Users/HP/64.py (3.4.3)
File Edit Format Run Options Window Help

```python
import heapq
from collections import defaultdict

def networkDelayTime(times, n, k):
    # Build adjacency list
    graph = defaultdict(list)
    for u, v, w in times:
        graph[u].append((v, w))

    # Min-heap for Dijkstra (time, node)
    heap = [(0, k)]
    dist = {}

    while heap:
        time, node = heapq.heappop(heap)
        if node in dist:
            continue
        dist[node] = time
        for nei, w in graph[node]:
            if nei not in dist:
                heapq.heappush(heap, (time + w, nei))

    if len(dist) == n:
        return max(dist.values())
    return -1


# Example 1
times = [[2,1,1],[2,3,1],[3,4,1]]
print(networkDelayTime(times, 4, 2))  # Output: 2

# Example 2
times = [[1,2,1]]
print(networkDelayTime(times, 2, 1))  # Output: 1

# Example 3
times = [[1,2,1]]
print(networkDelayTime(times, 2, 2))  # Output: -1
```

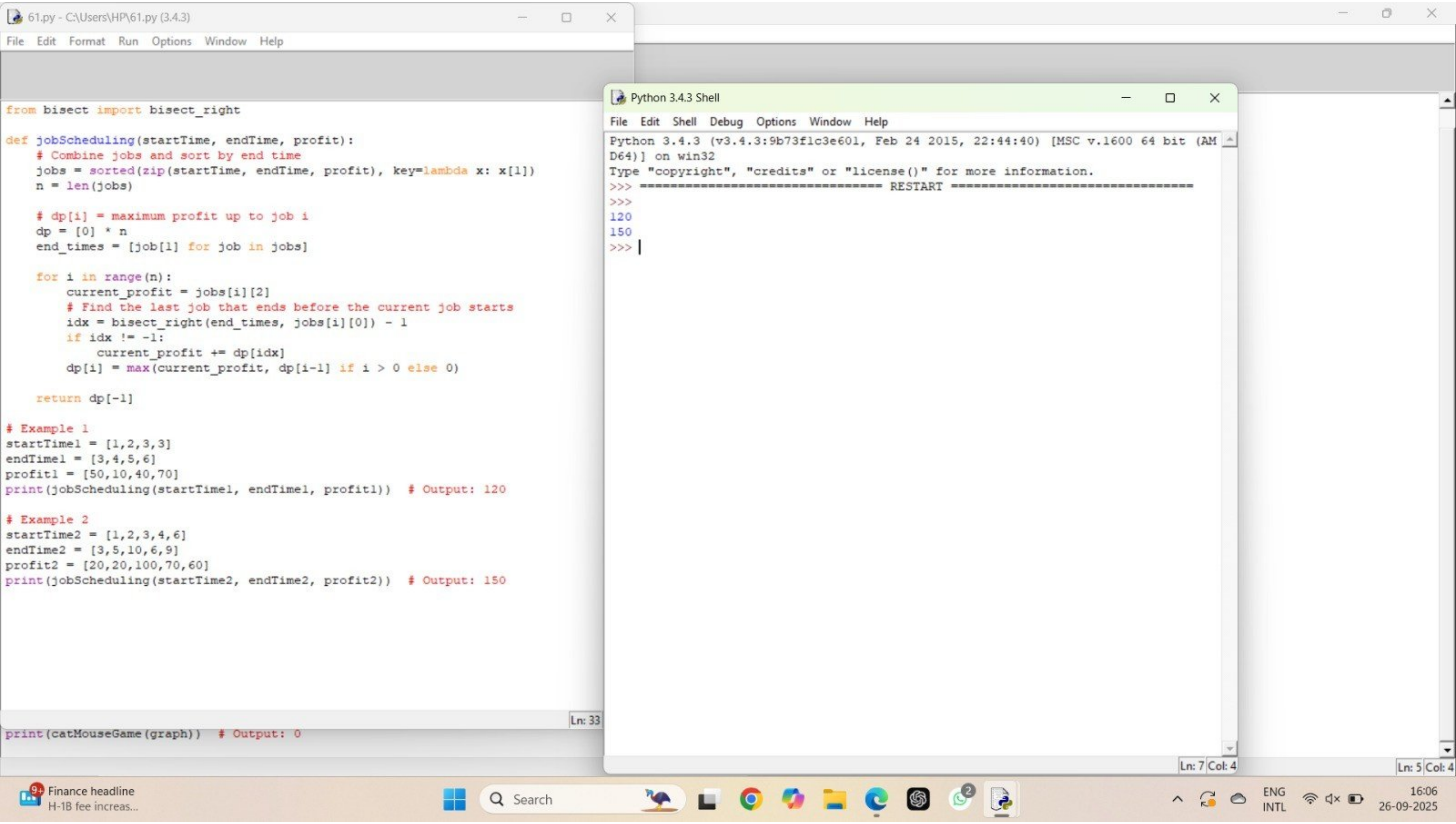
```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
2
1
-1
>>>
```


64.py - C:/Users/HP/64.py (3.4.3)
File Edit Format Run Options Window Help

```
from collections import Counter

def numIdenticalPairs(nums):
    freq = Counter(nums)
    count = 0
    for k in freq.values():
        count += k * (k - 1) // 2
    return count


# Example 1
print(numIdenticalPairs([1,2,3,1,1,3]))  # Output: 4

# Example 2
print(numIdenticalPairs([1,1,1,1]))      # Output: 6
```

Python 3.4.3 Shell
File Edit Shell Debug Options Window Help

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
4
6
>>>
```

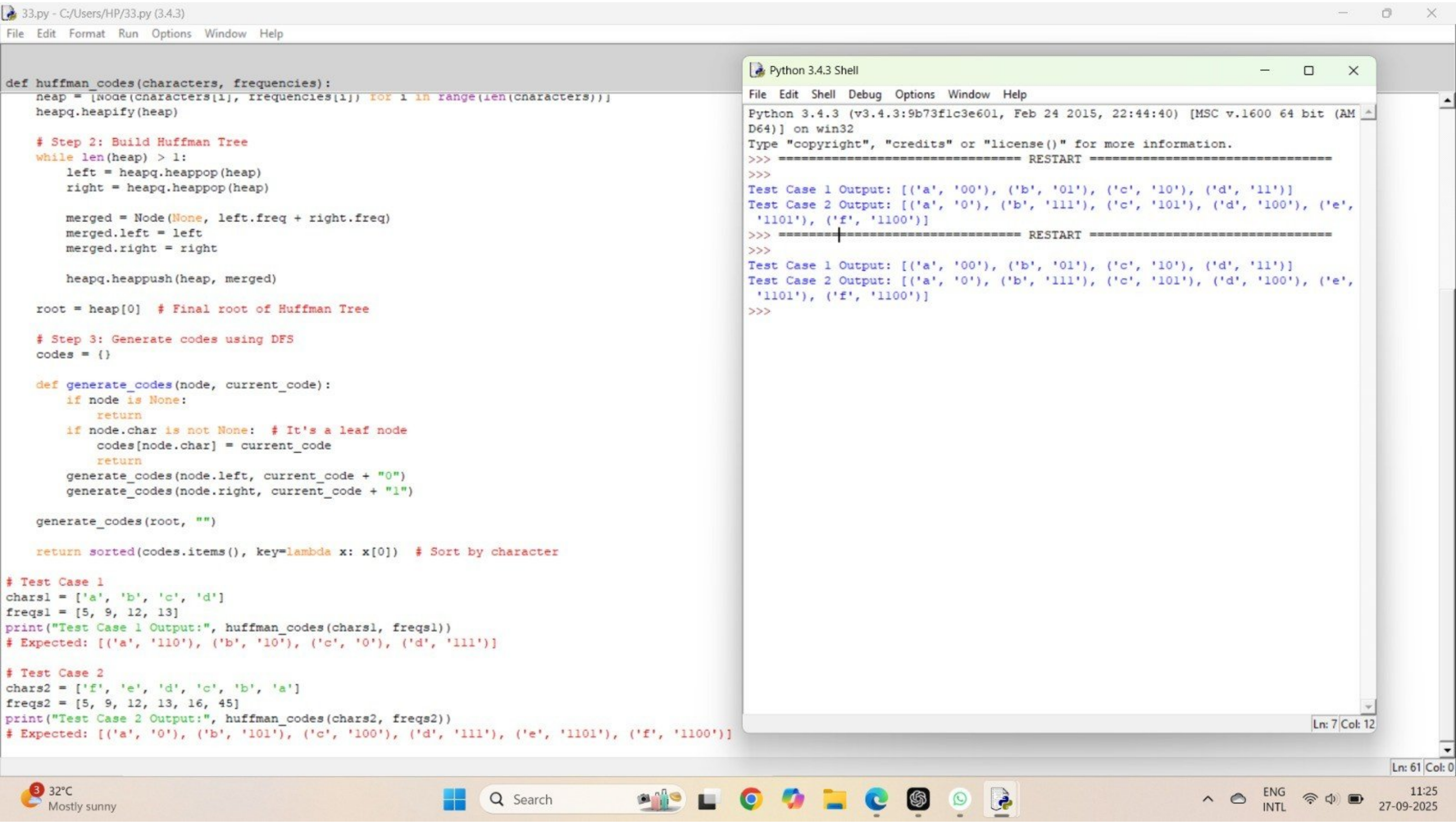


61.py - C:\Users\HP\61.py (3.4.3)
File Edit Format Run Options Window Help
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help

```
from bisect import bisect_right

def jobScheduling(startTime, endTime, profit):
    # Combine jobs and sort by end time
    jobs = sorted(zip(startTime, endTime, profit), key=lambda x: x[1])
    n = len(jobs)

    # dp[i] = maximum profit up to job i
    dp = [0] * n
    end_times = [job[1] for job in jobs]

    for i in range(n):
        current_profit = jobs[i][2]
        # Find the last job that ends before the current job starts
        idx = bisect_right(end_times, jobs[i][0]) - 1
        if idx != -1:
            current_profit += dp[idx]
        dp[i] = max(current_profit, dp[i-1] if i > 0 else 0)

    return dp[-1]

# Example 1
startTime1 = [1,2,3,3]
endTime1 = [3,4,5,6]
profit1 = [50,10,40,70]
print(jobScheduling(startTime1, endTime1, profit1))  # Output: 120

# Example 2
startTime2 = [1,2,3,4,6]
endTime2 = [3,5,10,6,9]
profit2 = [20,20,100,70,60]
print(jobScheduling(startTime2, endTime2, profit2))  # Output: 150
```

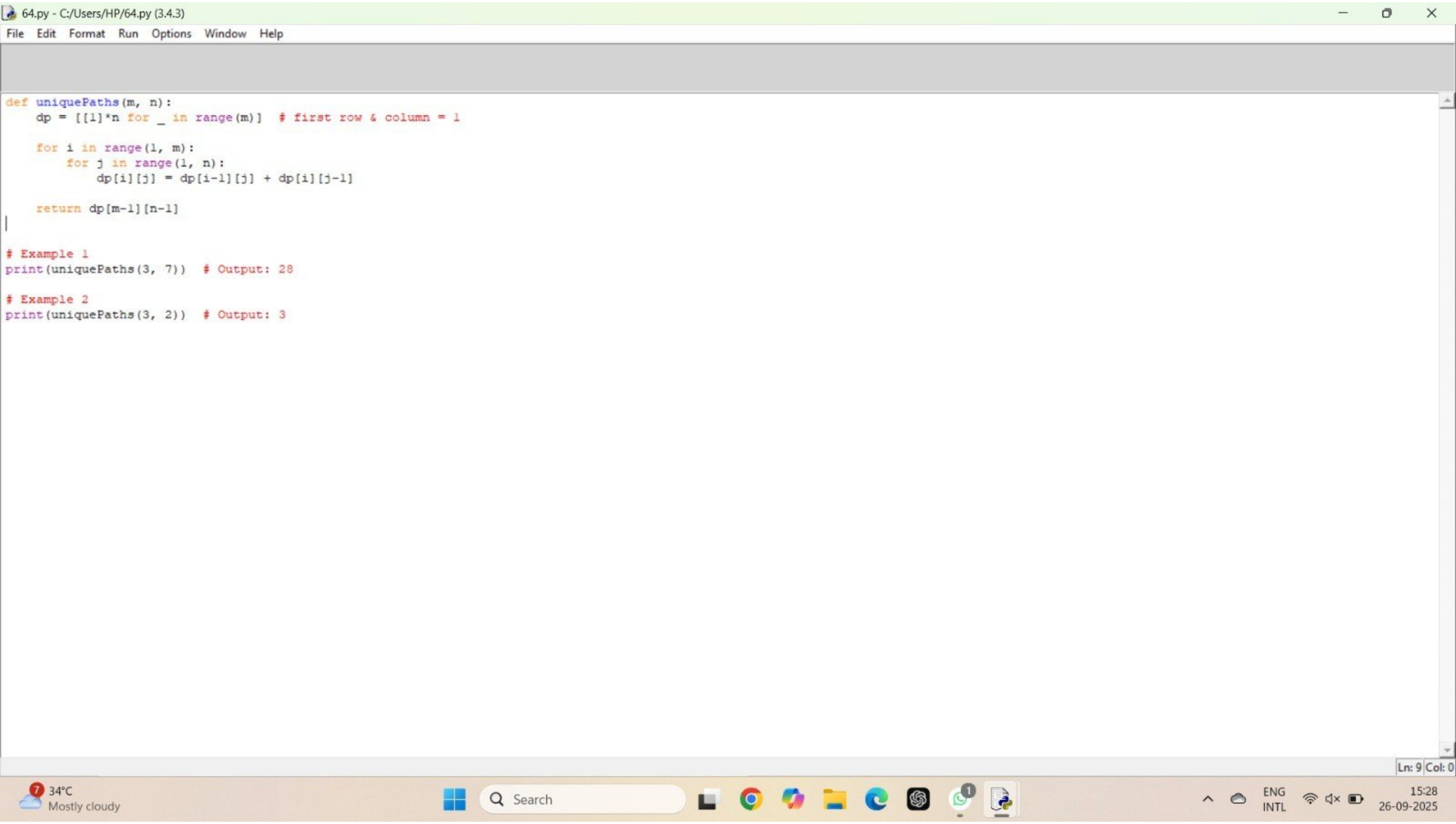
```
print(catMouseGame(graph))  # Output: 0
```

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
120
150
>>>
```

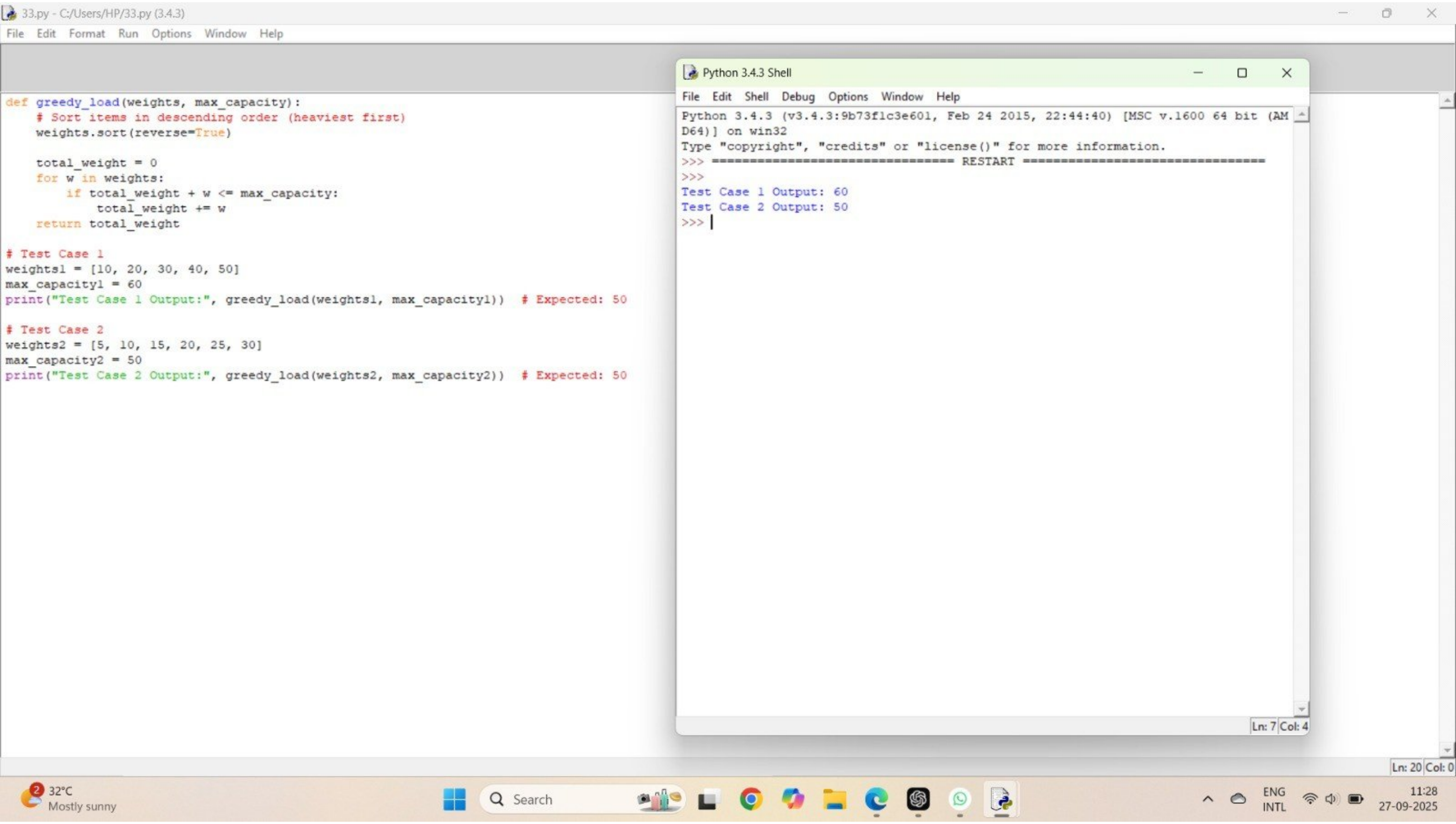

33.py - C:/Users/HP/33.py (3.4.3)
File Edit Format Run Options Window Help
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Ln: 7 Col: 12
Ln: 61 Col: 0
32°C Mostly sunny
Search
ENG INTL
11:25 27-09-2025

```python
def huffman_codes(characters, frequencies):
    heap = [Node(characters[i], frequencies[i]) for i in range(len(characters))]
    heapq.heapify(heap)

    # Step 2: Build Huffman Tree
    while len(heap) > 1:
        left = heapq.heappop(heap)
        right = heapq.heappop(heap)

        merged = Node(None, left.freq + right.freq)
        merged.left = left
        merged.right = right

        heapq.heappush(heap, merged)

    root = heap[0]  # Final root of Huffman Tree

    # Step 3: Generate codes using DFS
    codes = {}

    def generate_codes(node, current_code):
        if node is None:
            return
        if node.char is not None:  # It's a leaf node
            codes[node.char] = current_code
            return
        generate_codes(node.left, current_code + "0")
        generate_codes(node.right, current_code + "1")

    generate_codes(root, "")

    return sorted(codes.items(), key=lambda x: x[0])  # Sort by character

# Test Case 1
chars1 = ['a', 'b', 'c', 'd']
freqs1 = [5, 9, 12, 13]
print("Test Case 1 Output:", huffman_codes(chars1, freqs1))
# Expected: [('a', '110'), ('b', '10'), ('c', '0'), ('d', '111')]

# Test Case 2
chars2 = ['f', 'e', 'd', 'c', 'b', 'a']
freqs2 = [5, 9, 12, 13, 16, 45]
print("Test Case 2 Output:", huffman_codes(chars2, freqs2))
# Expected: [('a', '0'), ('b', '101'), ('c', '100'), ('d', '111'), ('e', '1101'), ('f', '1100')]
```

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Test Case 1 Output: [('a', '00'), ('b', '01'), ('c', '10'), ('d', '11')]
Test Case 2 Output: [('a', '0'), ('b', '111'), ('c', '101'), ('d', '100'), ('e',
 '1101'), ('f', '1100')]
>>> ================================ RESTART ================================
>>>
Test Case 1 Output: [('a', '00'), ('b', '01'), ('c', '10'), ('d', '11')]
Test Case 2 Output: [('a', '0'), ('b', '111'), ('c', '101'), ('d', '100'), ('e',
 '1101'), ('f', '1100')]
>>>
```


64.py - C:/Users/HP/64.py (3.4.3)
File Edit Format Run Options Window Help
Ln: 9 Col: 0
34°C
Mostly cloudy
Search
ENG
INTL
15:28
26-09-2025

```
def uniquePaths(m, n):
    dp = [[1]*n for _ in range(m)]  # first row & column = 1

    for i in range(1, m):
        for j in range(1, n):
            dp[i][j] = dp[i-1][j] + dp[i][j-1]

    return dp[m-1][n-1]


# Example 1
print(uniquePaths(3, 7))  # Output: 28

# Example 2
print(uniquePaths(3, 2))  # Output: 3
```



33.py - C:/Users/HP/33.py (3.4.3)
File Edit Format Run Options Window Help

```
def greedy_load(weights, max_capacity):
    # Sort items in descending order (heaviest first)
    weights.sort(reverse=True)

    total_weight = 0
    for w in weights:
        if total_weight + w <= max_capacity:
            total_weight += w
    return total_weight

# Test Case 1
weights1 = [10, 20, 30, 40, 50]
max_capacity1 = 60
print("Test Case 1 Output:", greedy_load(weights1, max_capacity1))  # Expected: 50

# Test Case 2
weights2 = [5, 10, 15, 20, 25, 30]
max_capacity2 = 50
print("Test Case 2 Output:", greedy_load(weights2, max_capacity2))  # Expected: 50
```

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Test Case 1 Output: 60
Test Case 2 Output: 50
>>> 
```