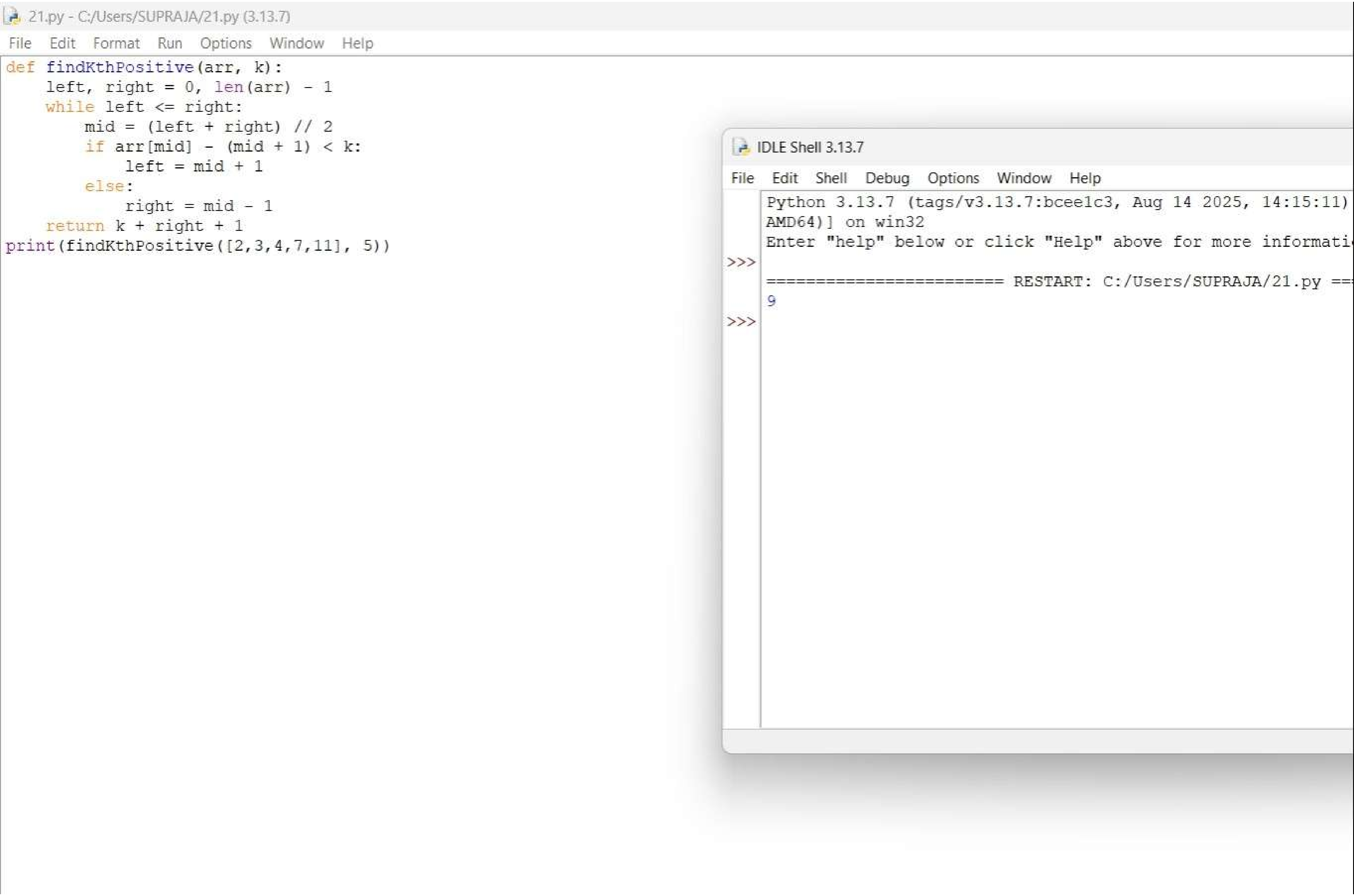
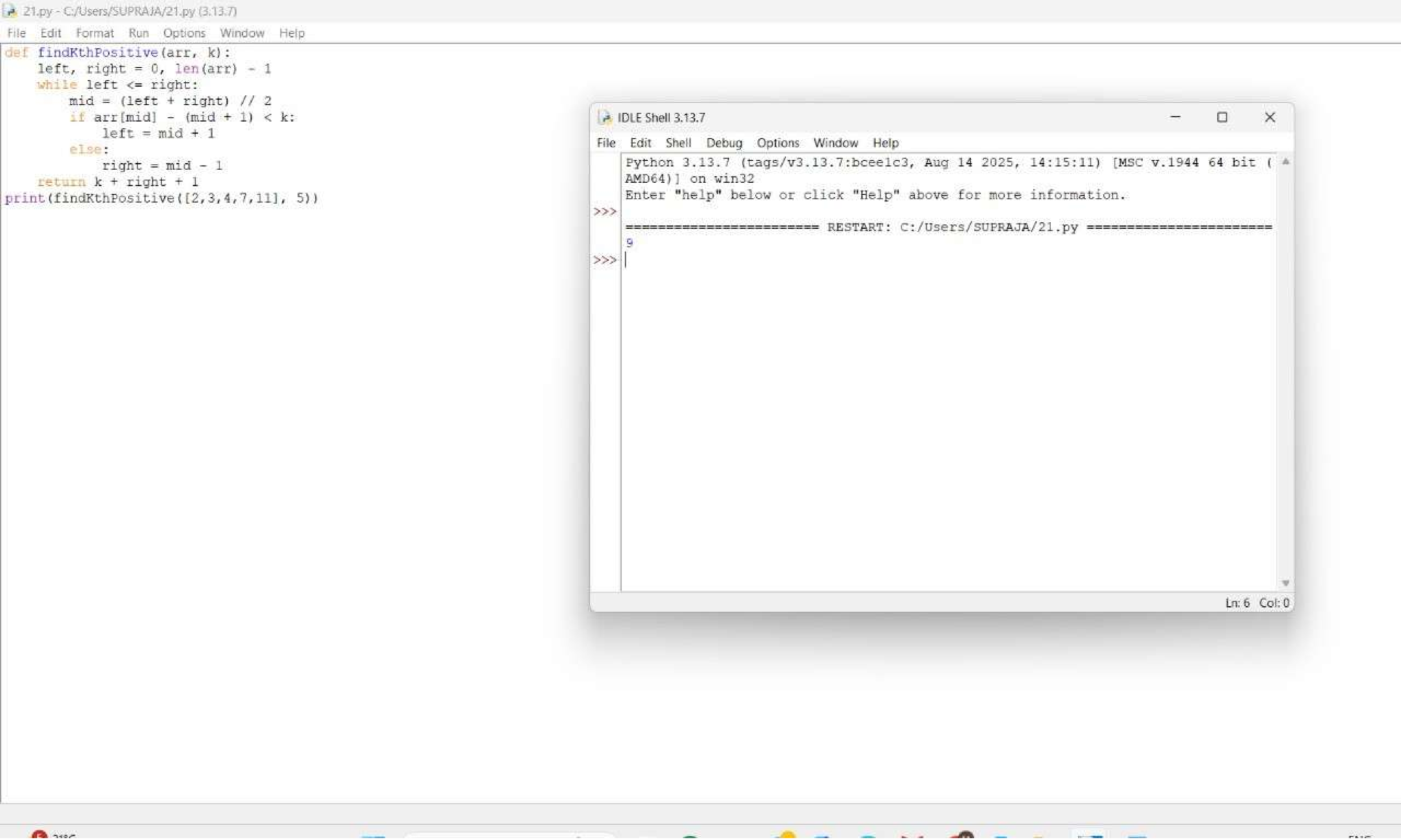

21.py - C:/Users/SUPRAJA/21.py (3.13.7)
File Edit Format Run Options Window Help

```
def findKthPositive(arr, k):
    left, right = 0, len(arr) - 1
    while left <= right:
        mid = (left + right) // 2
        if arr[mid] - (mid + 1) < k:
            left = mid + 1
        else:
            right = mid - 1
    return k + right + 1
print(findKthPositive([2,3,4,7,11], 5))
```

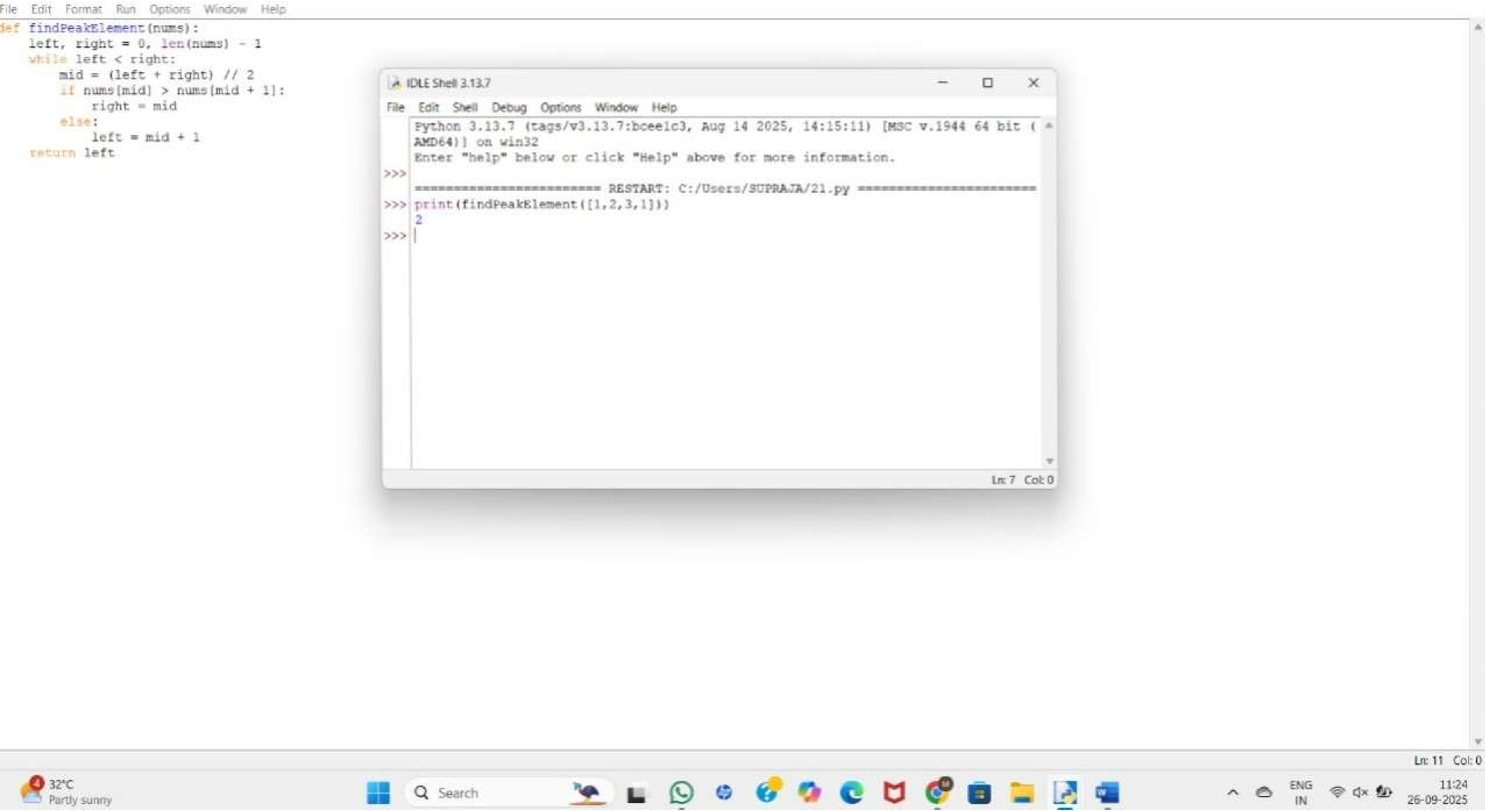
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help

```
    Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11)
    AMD64)] on win32
    Enter "help" below or click "Help" above for more informati
>>>
    ======================== RESTART: C:/Users/SUPRAJA/21.py ==
    9
>>>
```


21.py - C:/Users/SUPRAJA/21.py (3.13.7)
File Edit Format Run Options Window Help
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0

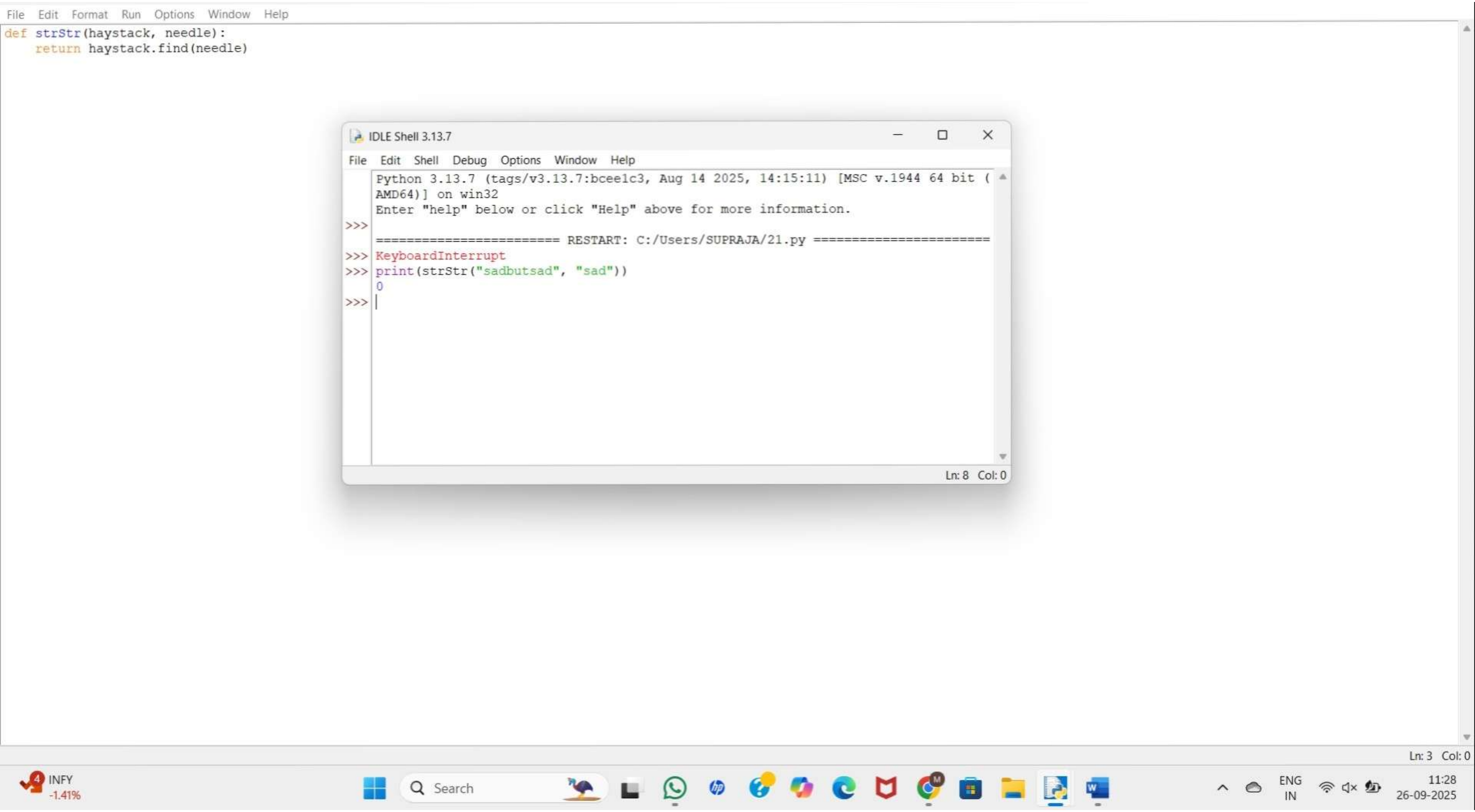
```python
def findKthPositive(arr, k):
    left, right = 0, len(arr) - 1
    while left <= right:
        mid = (left + right) // 2
        if arr[mid] - (mid + 1) < k:
            left = mid + 1
        else:
            right = mid - 1
    return k + right + 1
print(findKthPositive([2,3,4,7,11], 5))
```

```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
======================= RESTART: C:/Users/SUPRAJA/21.py ======================
9
>>> 
```



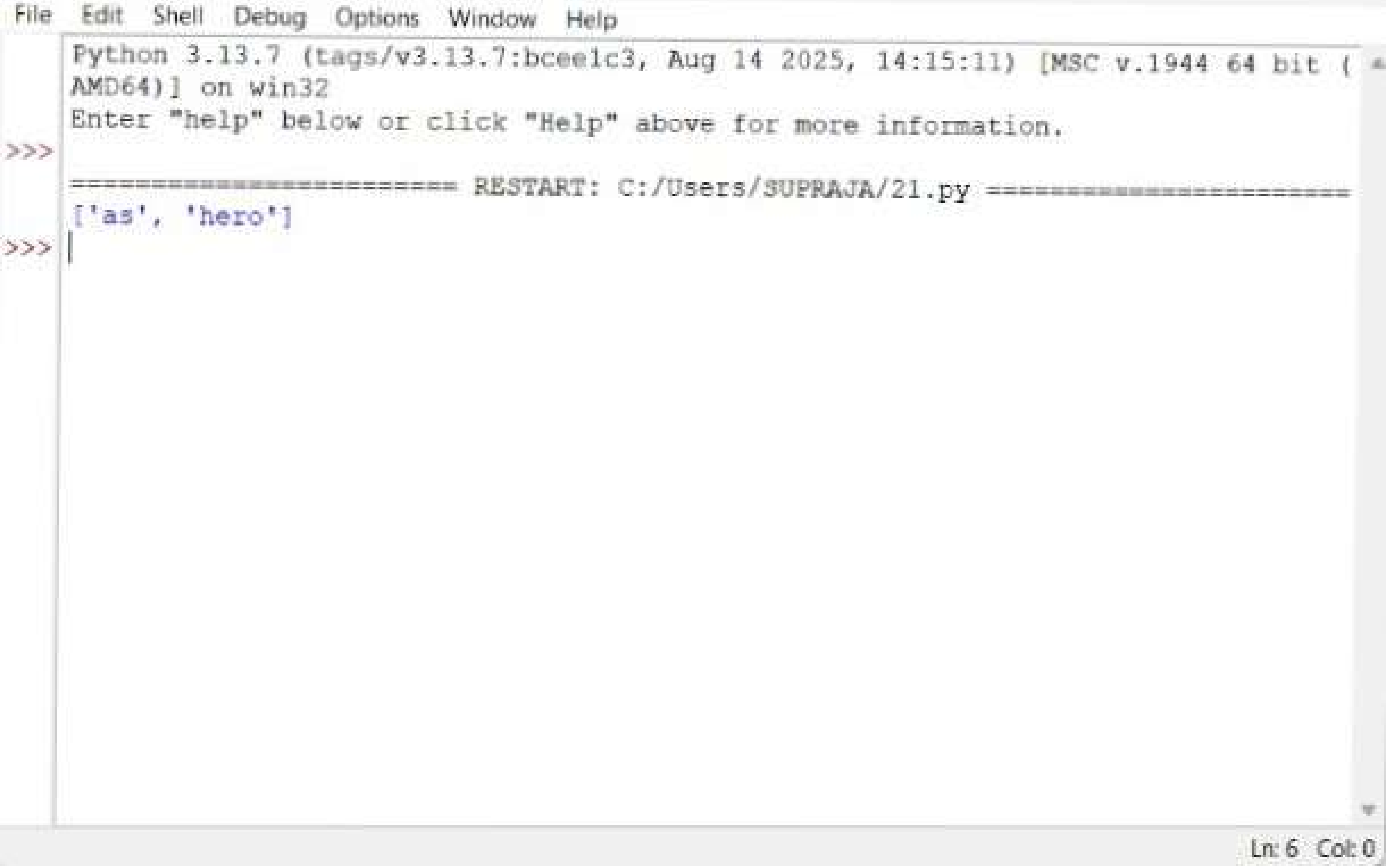
```
def findPeakElement(nums):
    left, right = 0, len(nums) - 1
    while left < right:
        mid = (left + right) // 2
        if nums[mid] > nums[mid + 1]:
            right = mid
        else:
            left = mid + 1
    return left
```

```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
======================== RESTART: C:/Users/SUPRAJA/21.py =======================
>>> print(findPeakElement([1,2,3,1]))
2
>>>
```

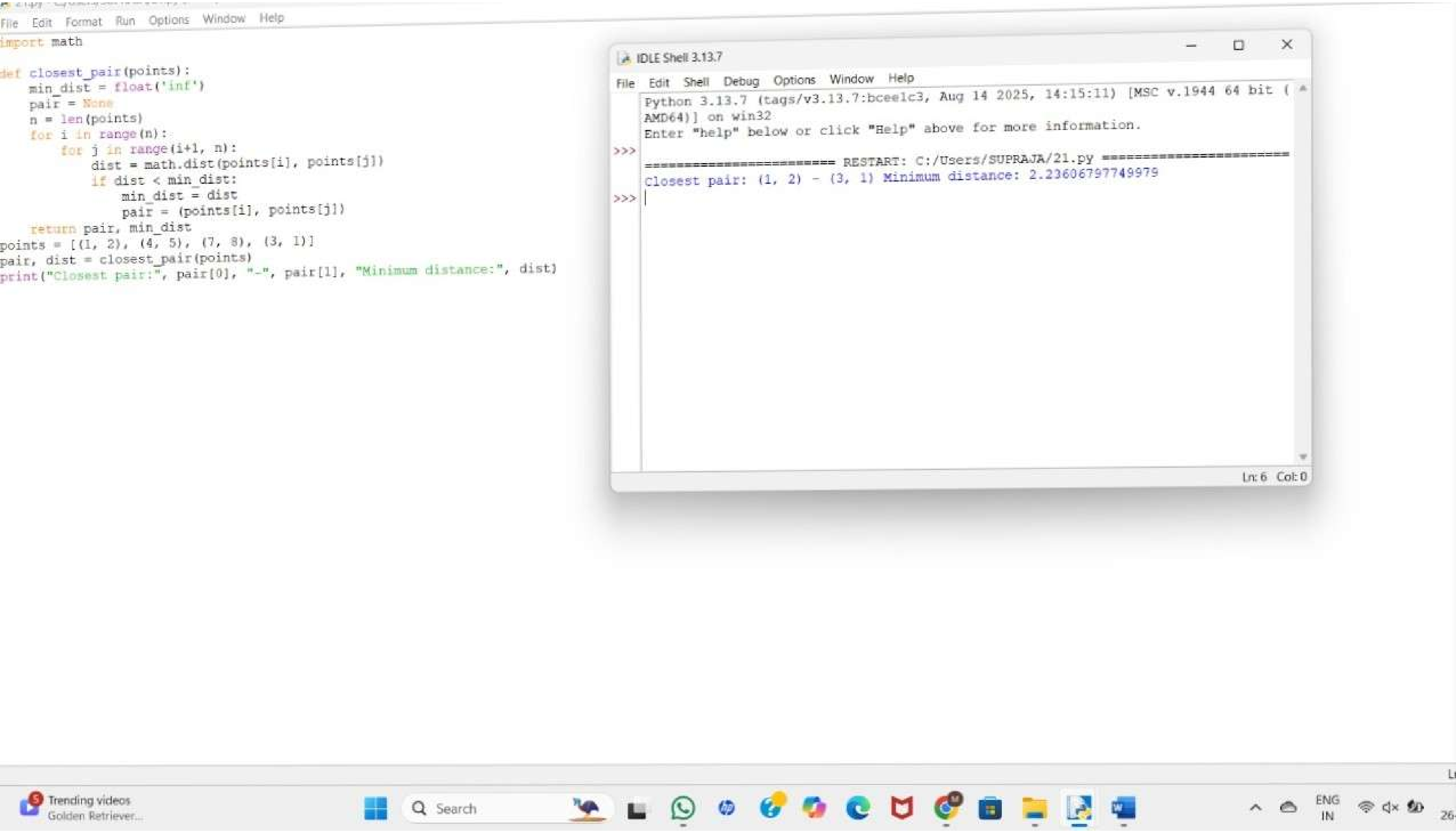

File Edit Format Run Options Window Help
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Ln: 8 Col: 0
Ln: 3 Col: 0
INFY
-1.41%
Search
ENG
IN
11:28
26-09-2025

```python
def strStr(haystack, needle):
    return haystack.find(needle)
```

```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
======================== RESTART: C:/Users/SUPRAJA/21.py =======================
>>> KeyboardInterrupt
>>> print(strStr("sadbutsad", "sad"))
0
>>> 
```



```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
========================= RESTART: C:/Users/SUPRAJA/21.py ========================
['as', 'hero']
>>>
```

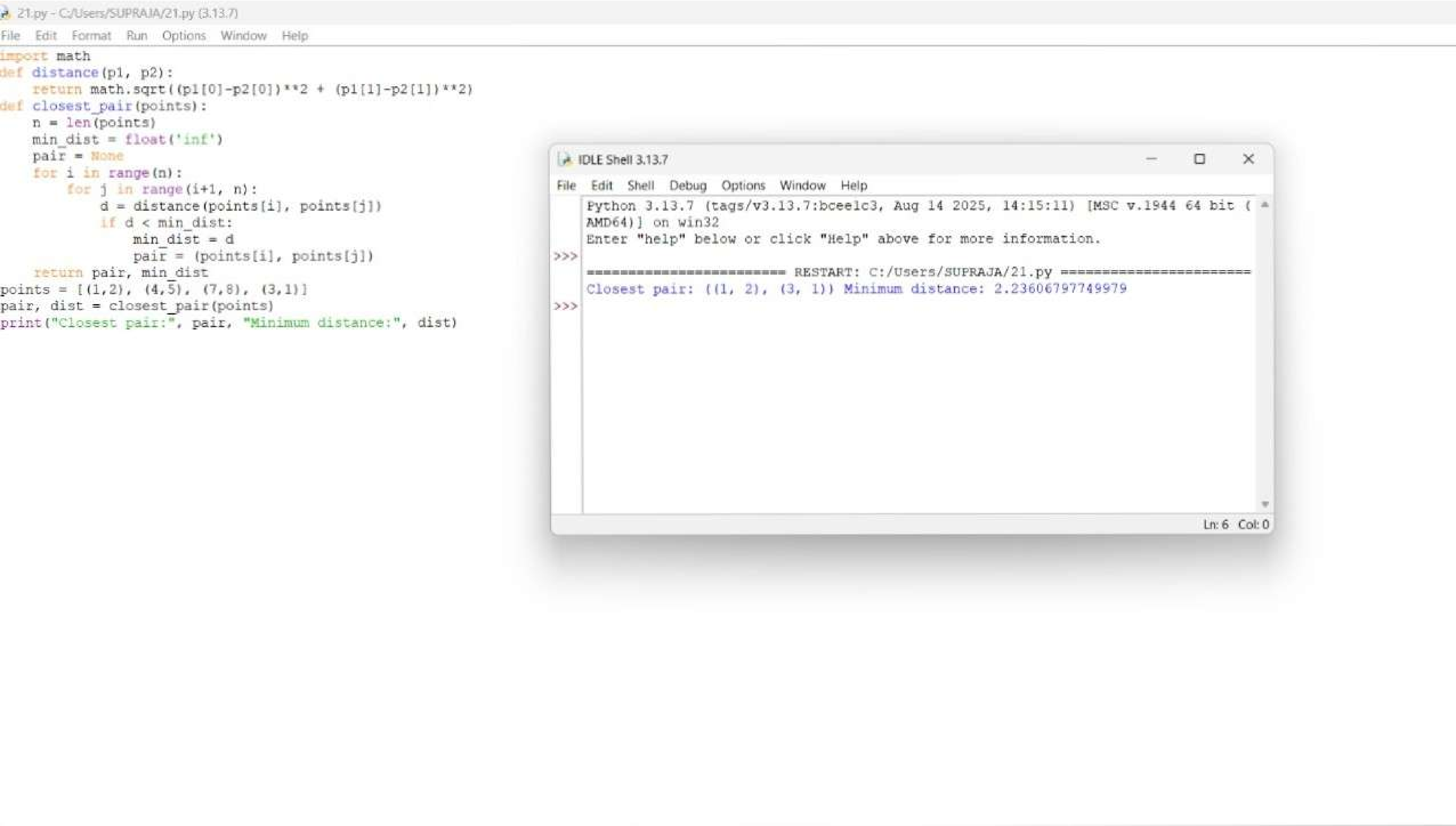


File Edit Format Run Options Window Help

```
import math

def closest_pair(points):
    min_dist = float('inf')
    pair = None
    n = len(points)
    for i in range(n):
        for j in range(i+1, n):
            dist = math.dist(points[i], points[j])
            if dist < min_dist:
                min_dist = dist
                pair = (points[i], points[j])
    return pair, min_dist
points = [(1, 2), (4, 5), (7, 8), (3, 1)]
pair, dist = closest_pair(points)
print("Closest pair:", pair[0], "-", pair[1], "Minimum distance:", dist)
```

IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help

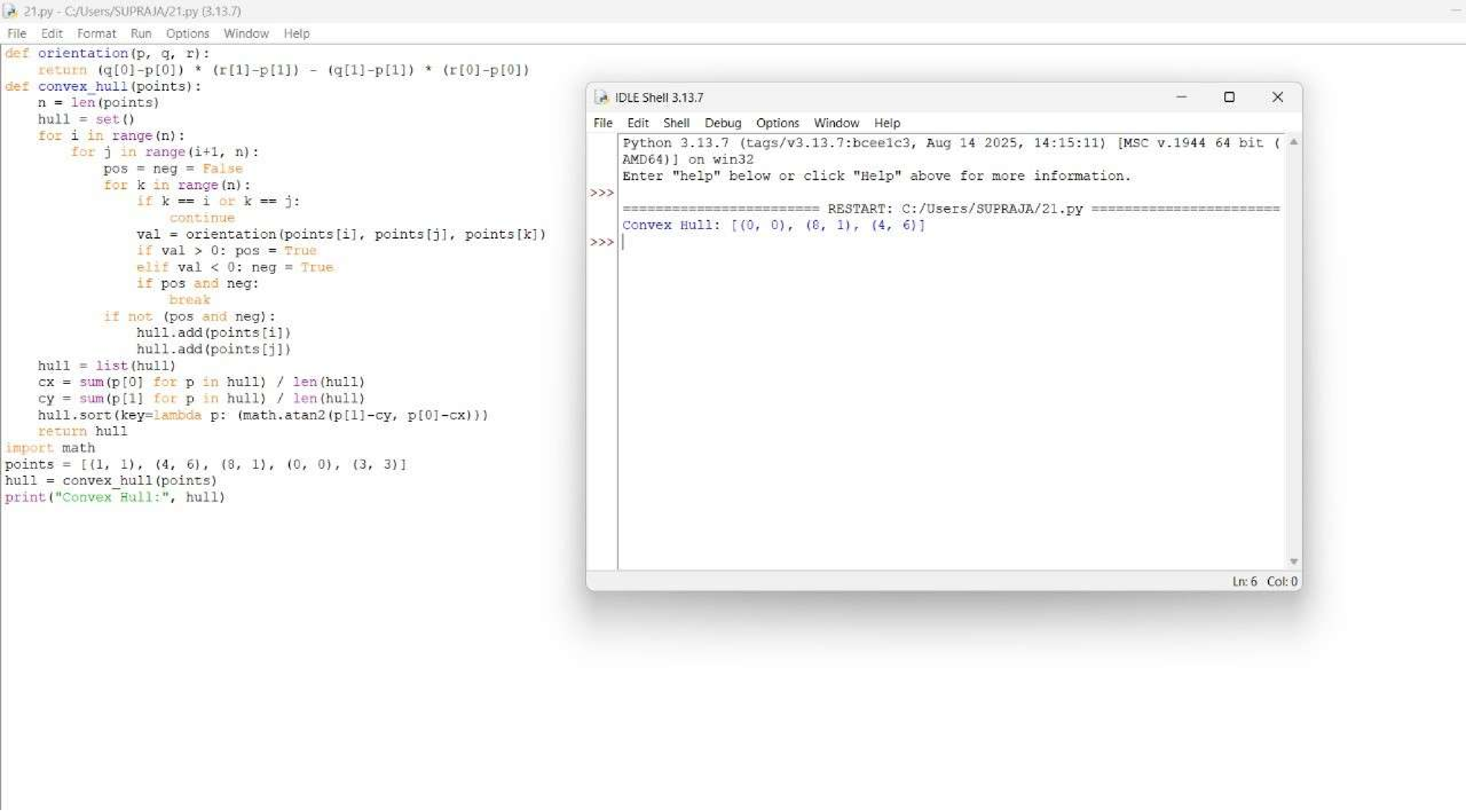
```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
========================= RESTART: C:/Users/SUPRAJA/21.py ========================
Closest pair: (1, 2) - (3, 1) Minimum distance: 2.23606797749979
>>> |
```


21.py - C:/Users/SUPRAJA/21.py (3.13.7)
File Edit Format Run Options Window Help

```python
import math
def distance(p1, p2):
    return math.sqrt((p1[0]-p2[0])**2 + (p1[1]-p2[1])**2)
def closest_pair(points):
    n = len(points)
    min_dist = float('inf')
    pair = None
    for i in range(n):
        for j in range(i+1, n):
            d = distance(points[i], points[j])
            if d < min_dist:
                min_dist = d
                pair = (points[i], points[j])
    return pair, min_dist
points = [(1,2), (4,5), (7,8), (3,1)]
pair, dist = closest_pair(points)
print("Closest pair:", pair, "Minimum distance:", dist)
```

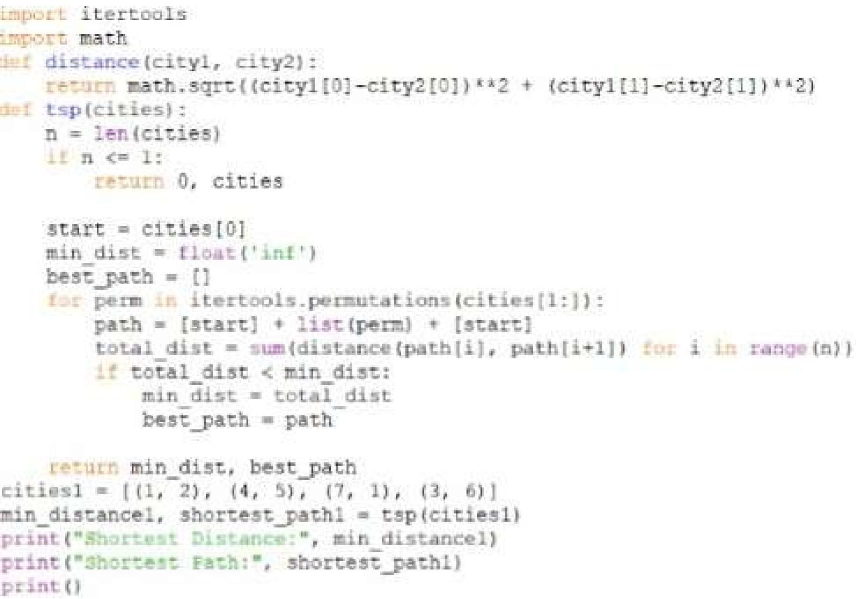
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0

```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
======================== RESTART: C:/Users/SUPRAJA/21.py =======================
Closest pair: ((1, 2), (3, 1)) Minimum distance: 2.23606797749979
>>>
```

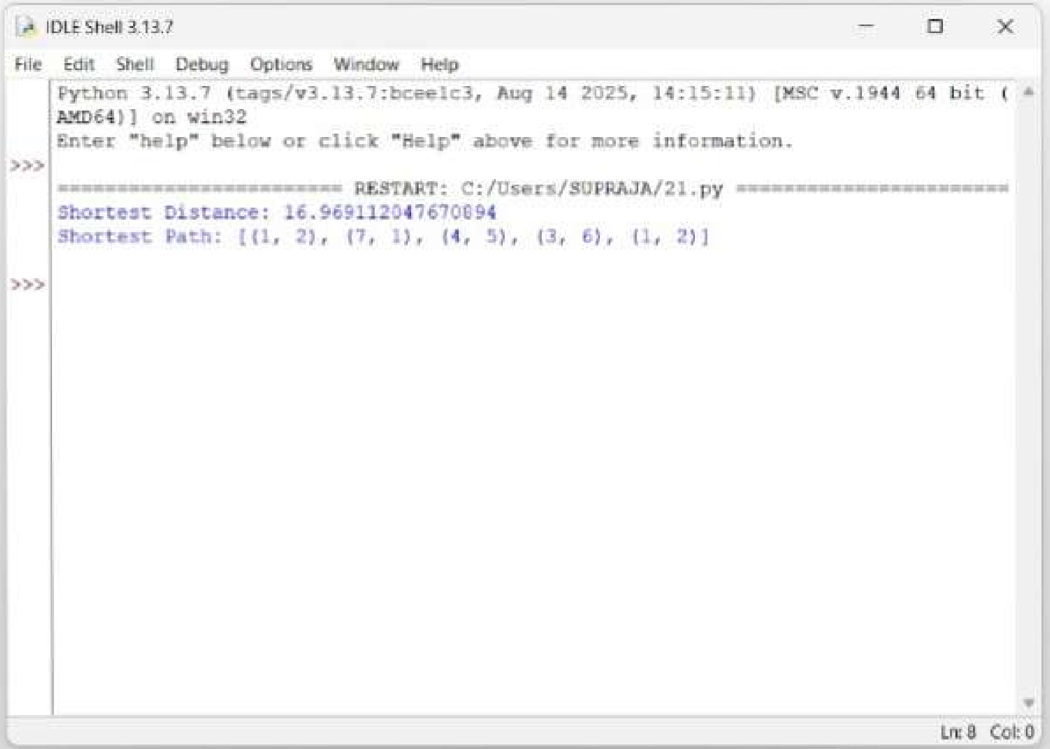

21.py - C:/Users/SUPRAJA/21.py (3.13.7)
File Edit Format Run Options Window Help
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0

```python
def orientation(p, q, r):
    return (q[0]-p[0]) * (r[1]-p[1]) - (q[1]-p[1]) * (r[0]-p[0])
def convex_hull(points):
    n = len(points)
    hull = set()
    for i in range(n):
        for j in range(i+1, n):
            pos = neg = False
            for k in range(n):
                if k == i or k == j:
                    continue
                val = orientation(points[i], points[j], points[k])
                if val > 0: pos = True
                elif val < 0: neg = True
                if pos and neg:
                    break
            if not (pos and neg):
                hull.add(points[i])
                hull.add(points[j])
    hull = list(hull)
    cx = sum(p[0] for p in hull) / len(hull)
    cy = sum(p[1] for p in hull) / len(hull)
    hull.sort(key=lambda p: (math.atan2(p[1]-cy, p[0]-cx)))
    return hull
import math
points = [(1, 1), (4, 6), (8, 1), (0, 0), (3, 3)]
hull = convex_hull(points)
print("Convex Hull:", hull)
```

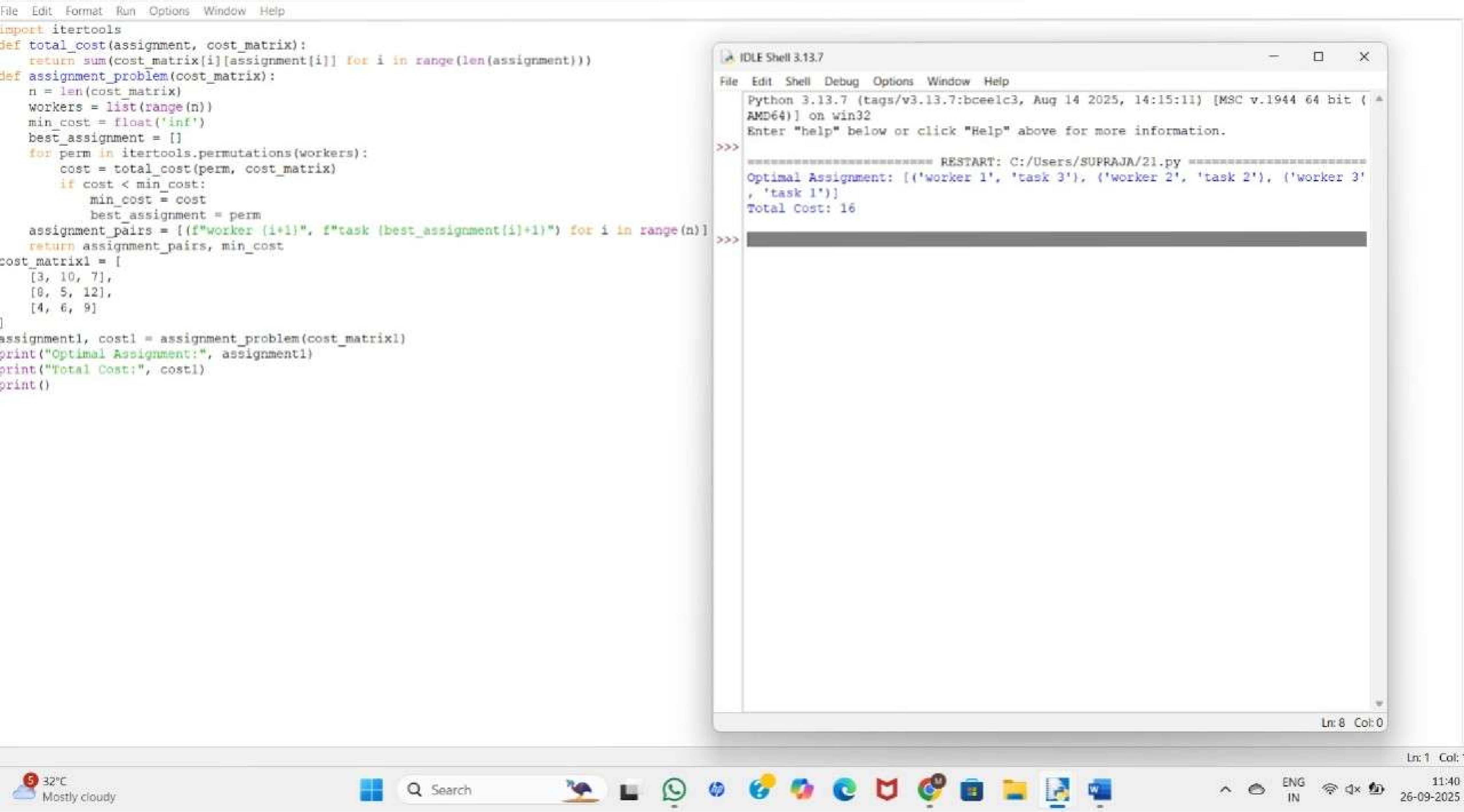
```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
======================== RESTART: C:/Users/SUPRAJA/21.py ========================
Convex Hull: [(0, 0), (8, 1), (4, 6)]
>>>
```



```
import itertools
import math
def distance(city1, city2):
    return math.sqrt((city1[0]-city2[0])**2 + (city1[1]-city2[1])**2)
def tsp(cities):
    n = len(cities)
    if n <= 1:
        return 0, cities

    start = cities[0]
    min_dist = float('inf')
    best_path = []
    for perm in itertools.permutations(cities[1:]):
        path = [start] + list(perm) + [start]
        total_dist = sum(distance(path[i], path[i+1]) for i in range(n))
        if total_dist < min_dist:
            min_dist = total_dist
            best_path = path

    return min_dist, best_path
cities1 = [(1, 2), (4, 5), (7, 1), (3, 6)]
min_distance1, shortest_path1 = tsp(cities1)
print("Shortest Distance:", min_distance1)
print("Shortest Path:", shortest_path1)
print()
```



```
    Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
    AMD64)] on win32
    Enter "help" below or click "Help" above for more information.
>>>
    ======================== RESTART: C:/Users/SUPRAJA/21.py =======================
    Shortest Distance: 16.969112047670894
    Shortest Path: [(1, 2), (7, 1), (4, 5), (3, 6), (1, 2)]

>>>
```



```
import itertools
def total_cost(assignment, cost_matrix):
    return sum(cost_matrix[i][assignment[i]] for i in range(len(assignment)))
def assignment_problem(cost_matrix):
    n = len(cost_matrix)
    workers = list(range(n))
    min_cost = float('inf')
    best_assignment = []
    for perm in itertools.permutations(workers):
        cost = total_cost(perm, cost_matrix)
        if cost < min_cost:
            min_cost = cost
            best_assignment = perm
    assignment_pairs = [(f"worker {i+1}", f"task {best_assignment[i]+1}") for i in range(n)]
    return assignment_pairs, min_cost
cost_matrix1 = [
    [3, 10, 7],
    [8, 5, 12],
    [4, 6, 9]
]
assignment1, cost1 = assignment_problem(cost_matrix1)
print("Optimal Assignment:", assignment1)
print("Total Cost:", cost1)
print()
```

```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
========================= RESTART: C:/Users/SUPRAJA/21.py ========================
Optimal Assignment: [('worker 1', 'task 3'), ('worker 2', 'task 2'), ('worker 3'
, 'task 1')]
Total Cost: 16

>>>
```