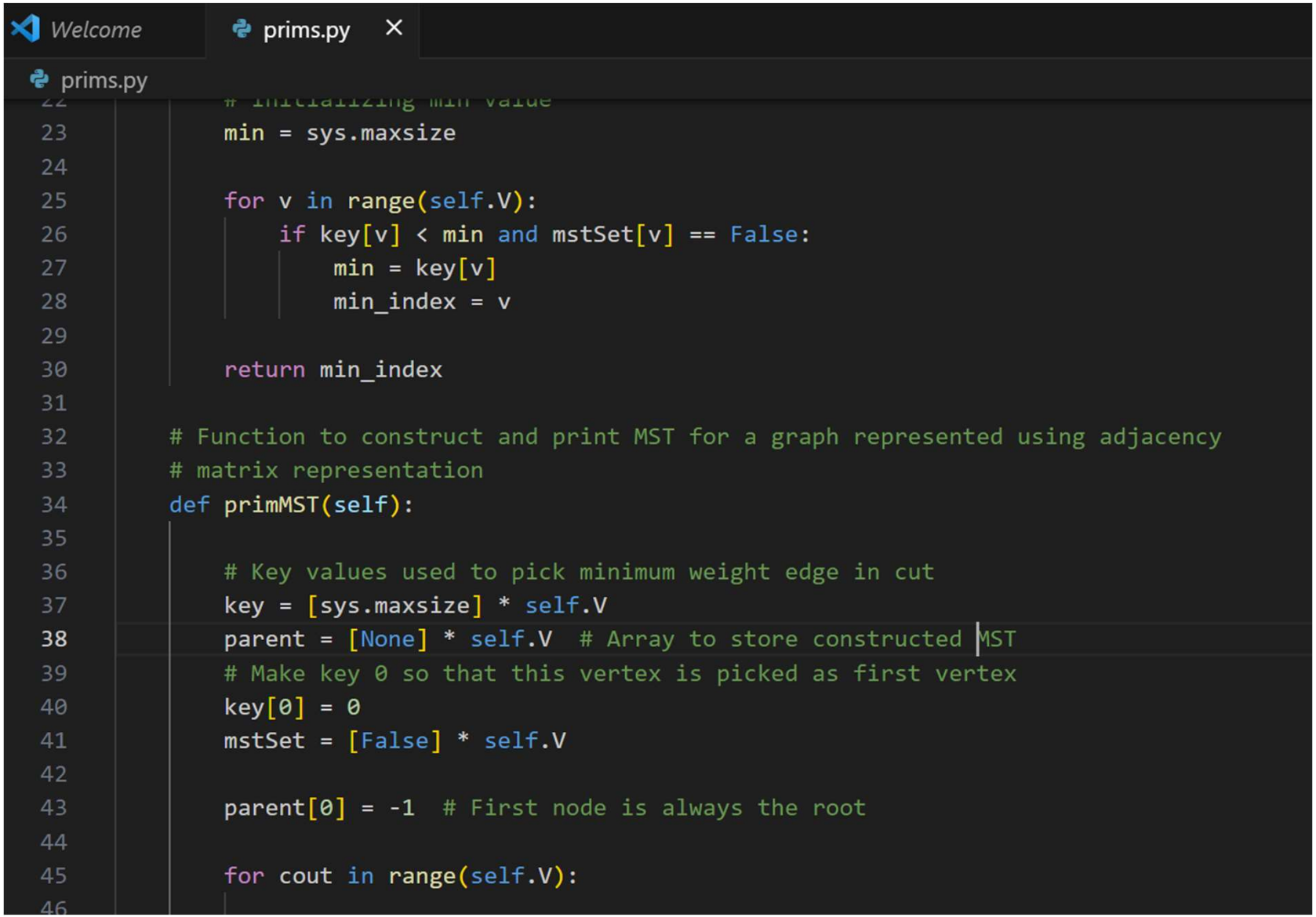
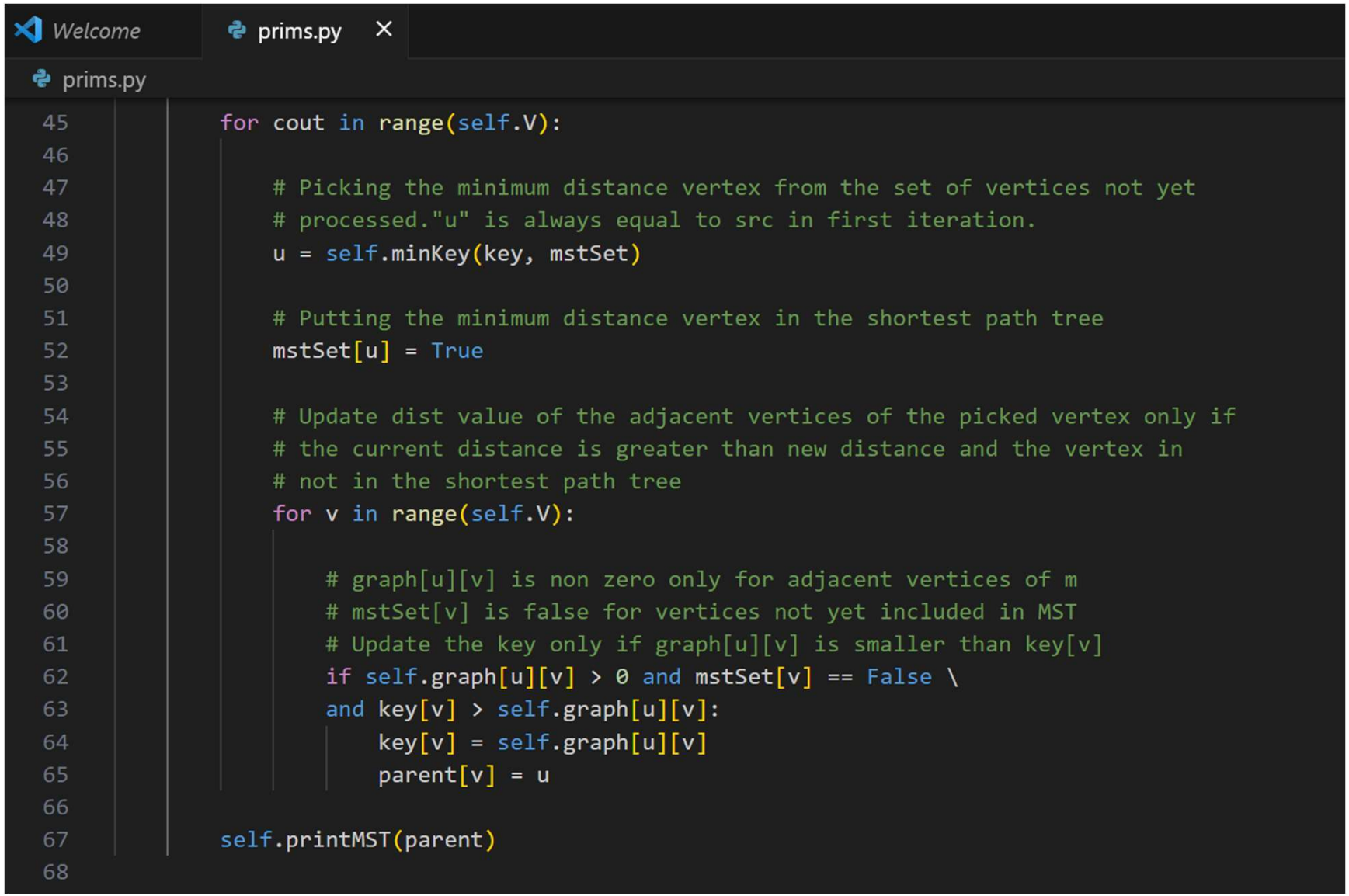
Code (Prim's Algorithm)

Welcome prims.py

prims.py

```python
# Prim's Minimum Spanning Tree (MST) algorithm.
# The program is for adjacency matrix representation of the graph.

import sys

class Graph():
    def __init__(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)]
                      for row in range(vertices)]

    # A utility function to print the constructed MST stored in parent[]
    def printMST(self, parent):
        print("Edge \tWeight")
        for i in range(1, self.V):
            print(parent[i], "-", i, "\t", self.graph[i][parent[i]])

    # A utility function to find the vertex with minimum distance value, from the set of
    # vertices not yet included in shortest path tree
    def minKey(self, key, mstSet):

        # Initializing min value
        min = sys.maxsize
```

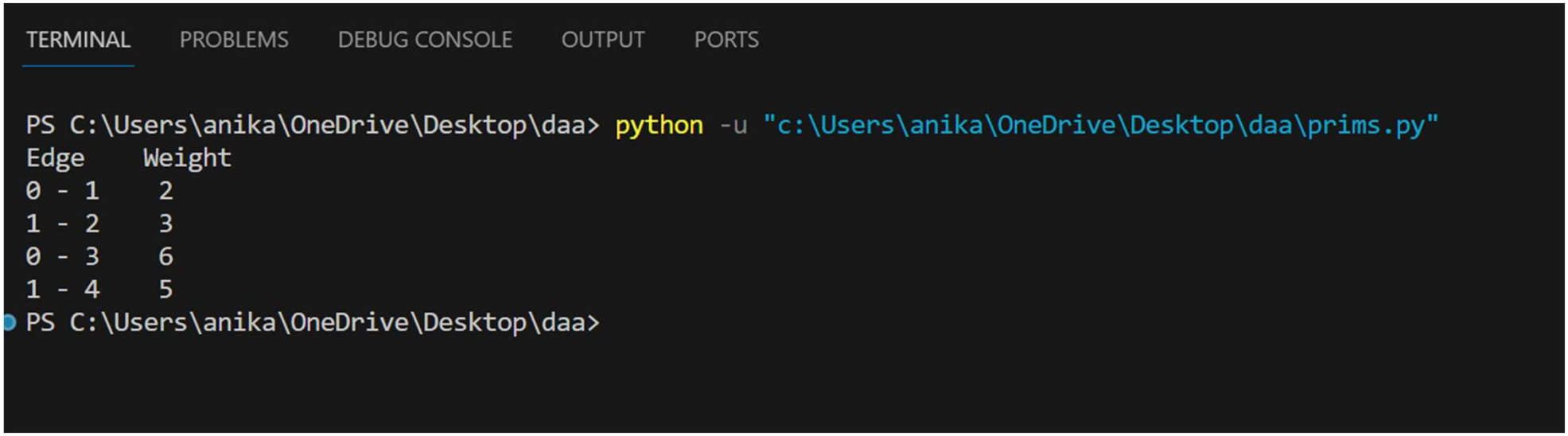

Welcome
prims.py
prims.py
# Initializing min value
min = sys.maxsize
for v in range(self.V):
if key[v] < min and mstSet[v] == False:
min = key[v]
min_index = v
return min_index
# Function to construct and print MST for a graph represented using adjacency
# matrix representation
def primMST(self):
# Key values used to pick minimum weight edge in cut
key = [sys.maxsize] * self.V
parent = [None] * self.V # Array to store constructed MST
# Make key 0 so that this vertex is picked as first vertex
key[0] = 0
mstSet = [False] * self.V
parent[0] = -1 # First node is always the root
for cout in range(self.V):


Welcome
prims.py
prims.py
45 for cout in range(self.V):
46
47 # Picking the minimum distance vertex from the set of vertices not yet
48 # processed."u" is always equal to src in first iteration.
49 u = self.minKey(key, mstSet)
50
51 # Putting the minimum distance vertex in the shortest path tree
52 mstSet[u] = True
53
54 # Update dist value of the adjacent vertices of the picked vertex only if
55 # the current distance is greater than new distance and the vertex in
56 # not in the shortest path tree
57 for v in range(self.V):
58
59 # graph[u][v] is non zero only for adjacent vertices of m
60 # mstSet[v] is false for vertices not yet included in MST
61 # Update the key only if graph[u][v] is smaller than key[v]
62 if self.graph[u][v] > 0 and mstSet[v] == False \
63 and key[v] > self.graph[u][v]:
64 key[v] = self.graph[u][v]
65 parent[v] = u
66
67 self.printMST(parent)
68

prims.py

```
66
67              self.printMST(parent)
68
69
70      # Driver's code
71      if __name__ == '__main__':
72          g = Graph(5)
73          g.graph = [[0, 2, 0, 6, 0],
74                     [2, 0, 3, 8, 5],
75                     [0, 3, 0, 0, 7],
76                     [6, 8, 0, 0, 9],
77                     [0, 5, 7, 9, 0]]
78
79          g.primMST()
80
81
```

Output


TERMINAL
PROBLEMS
DEBUG CONSOLE
OUTPUT
PORTS
PS C:\Users\anika\OneDrive\Desktop\daa> python -u "c:\Users\anika\OneDrive\Desktop\daa\prims.py"
Edge Weight
0 - 1 2
1 - 2 3
0 - 3 6
1 - 4 5
PS C:\Users\anika\OneDrive\Desktop\daa>