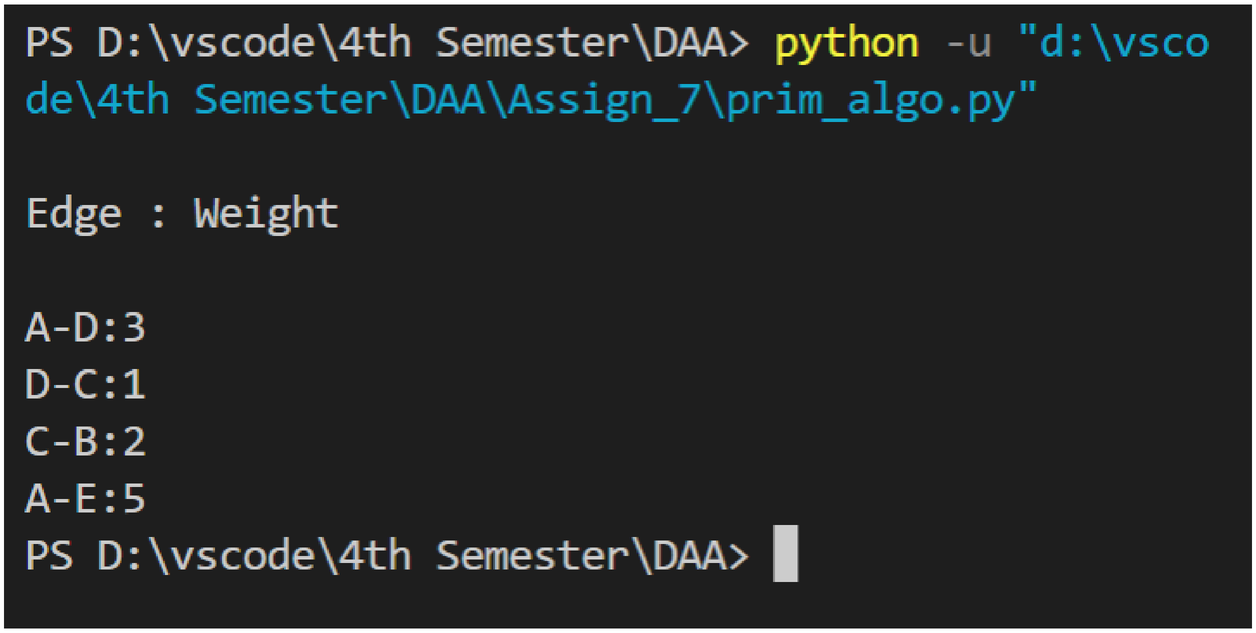
Name : Sabarivasan V
Class : CSE - B
Reg No : 205001085

Question 1 :

```
# 1) Implement Prim's Algorithm for
# computing MST using Greedy Approach


def prim( G , V ):

    INF = 9999999

    # create a array to track selected vertex
    # selected will become true otherwise false
    selected = [0, 0, 0, 0, 0]
    # set number of edge to 0

    no_edge = 0
    # the number of egde in minimum spanning tree will be
    # always less than(V - 1), where V is number of vertices in
    # graph
    # choose 0th vertex and make it true
    selected[0] = True
    # print for edge and weight
    print("\nEdge : Weight\n")
    while (no_edge < V - 1):
        # For every vertex in the set S, find the all adjacent
vertices
        #, calculate the distance from the vertex selected at step 1.
        # if the vertex is already in the set S, discard it otherwise
        # choose another vertex nearest to selected vertex  at step
1.
        minimum = INF
        x = 0
        y = 0
```

```python
        for i in range(V):
            if selected[i]:
                for j in range(V):
                    if ((not selected[j]) and G[i][j]):
                        # not in selected and there is an edge
                        if minimum > G[i][j]:
                            minimum = G[i][j]
                            x = i
                            y = j
        print(chr(65+x) + "-" + chr(65+y) + ":" + str(G[x][y]))
        selected[y] = True
        no_edge += 1

# number of vertices in graph
V = 5

# create a 2d array of size 5x5
# for adjacency matrix to represent graph
G = [[0, 4, 0, 3, 5],
     [4, 0, 2, 0, 0],
     [0, 2, 0, 1, 0],
     [3, 0, 1, 0, 0],
     [5, 0, 0, 0, 0]]

prim(G,V)
```

Output :

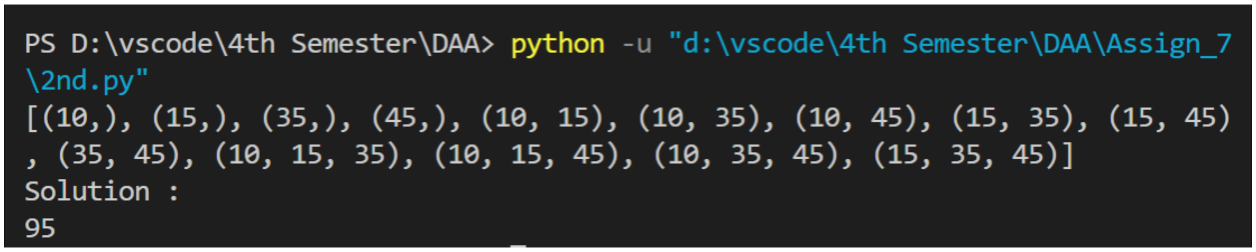


```
PS D:\vscode\4th Semester\DAA> python -u "d:\vsco
de\4th Semester\DAA\Assign_7\prim_algo.py"

Edge : Weight

A-D:3
D-C:1
C-B:2
A-E:5
PS D:\vscode\4th Semester\DAA>
```

Question 2 :

```python
# 2) Implement Greedy solution for the
# Knapsack problem

from itertools import combinations as c

capacity  = 100
weigths = [10, 15, 35, 45]
possibilities = []
best = 0
flag = 0

for i in range(len(weigths)) :
    comb = c(weigths,i+1)
    for i in comb :
        if sum(i) <= capacity:
            best = max(best,sum(i))
            if best == capacity:
                print("Solution : ")
                print(i)
                flag = 1
                break
            possibilities.append(i)
    if flag ==  1 :
        break

if flag == 0 :
    print(possibilities)
    print("Solution : ")
    print(best)
```

Output :



```
PS D:\vscode\4th Semester\DAA> python -u "d:\vscode\4th Semester\DAA\Assign_7
\2nd.py"
[(10,), (15,), (35,), (45,), (10, 15), (10, 35), (10, 45), (15, 35), (15, 45)
, (35, 45), (10, 15, 35), (10, 15, 45), (10, 35, 45), (15, 35, 45)]
Solution :
95
```