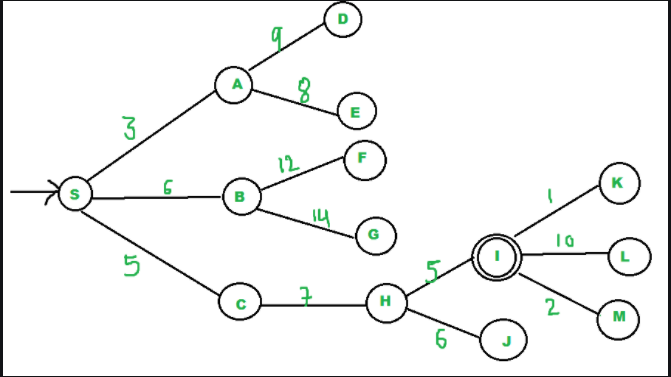
**EX5A: implementation of best first search algorithm**

**Name: V. Siva Supradeep**

**Reg no: RA1911030010104**

**AIM:** To implement the best first search algorithm using python.

**Algorithm:**

****

We start from source "S" and search for

goal "I" using given costs and Best

First search.

pq initially contains S

We remove s from and process unvisited

neighbors of S to pq.

pq now contains {A, C, B} (C is put

before B because C has lesser cost)

We remove A from pq and process unvisited

neighbors of A to pq.

pq now contains {C, B, E, D}

We remove C from pq and process unvisited

neighbors of C to pq.

pq now contains {B, H, E, D}

We remove B from pq and process unvisited

neighbors of B to pq.

pq now contains {H, E, D, F, G}

We remove H from pq. Since our goal

"I" is a neighbor of H, we return.

**Code:**

from queue import PriorityQueue

v = 14

graph = [[] for i in range(v)]

def best\_first\_search(source, target, n):

visited = [0] \* n

visited[0] = True

pq = PriorityQueue()

pq.put((0, source))

while pq.empty() == False:

u = pq.get()[1]

print(u, end=" ")

if u == target:

break

for v, c in graph[u]:

if visited[v] == False:

visited[v] = True

pq.put((c, v))

print()

def addedge(x, y, cost):

graph[x].append((y, cost))

graph[y].append((x, cost))

addedge(0, 1, 3)

addedge(0, 2, 6)

addedge(0, 3, 5)

addedge(1, 4, 9)

addedge(1, 5, 8)

addedge(2, 6, 12)

addedge(2, 7, 14)

addedge(3, 8, 7)

addedge(8, 9, 5)

addedge(8, 10, 6)

addedge(9, 11, 1)

addedge(9, 12, 10)

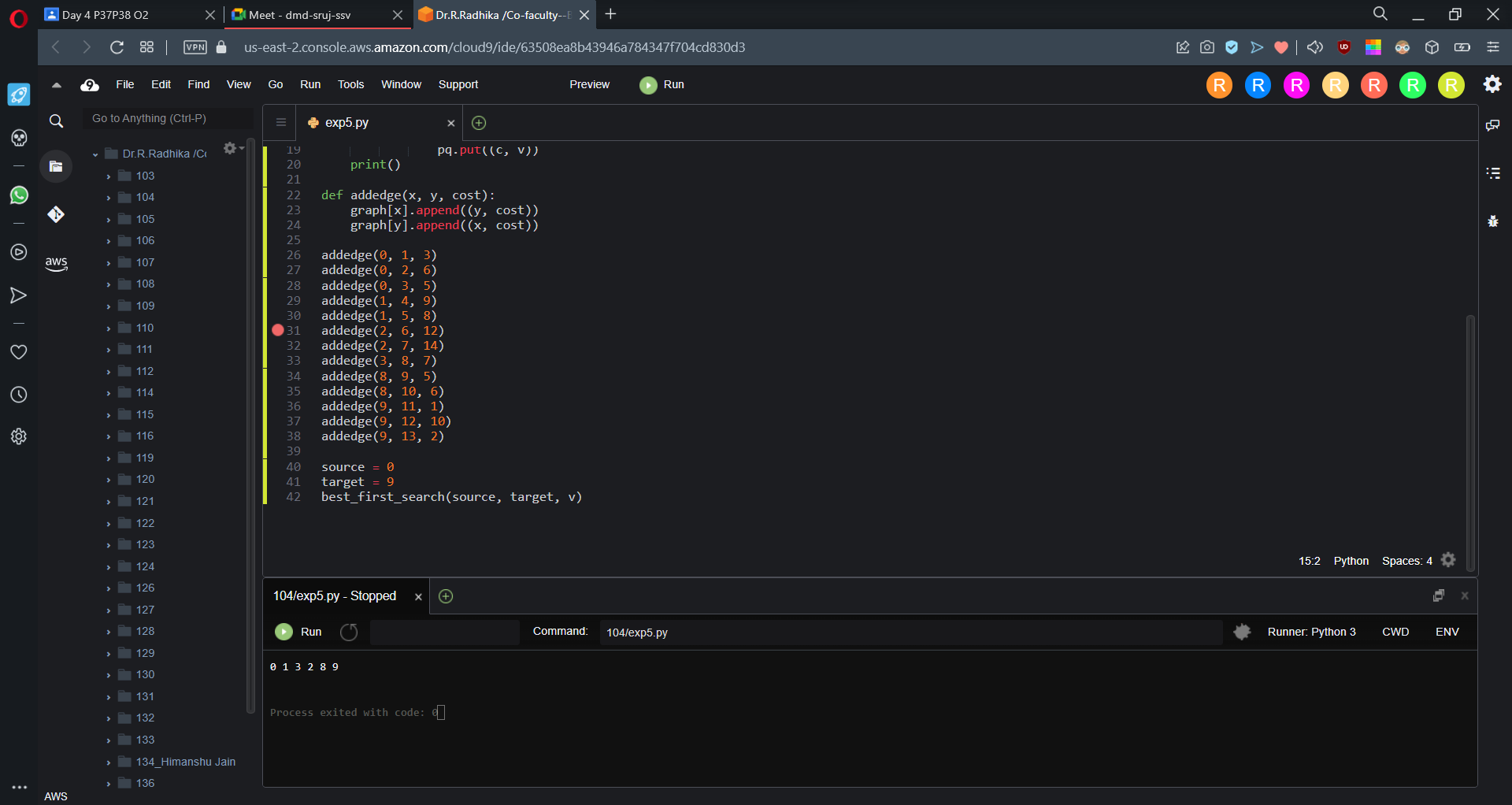
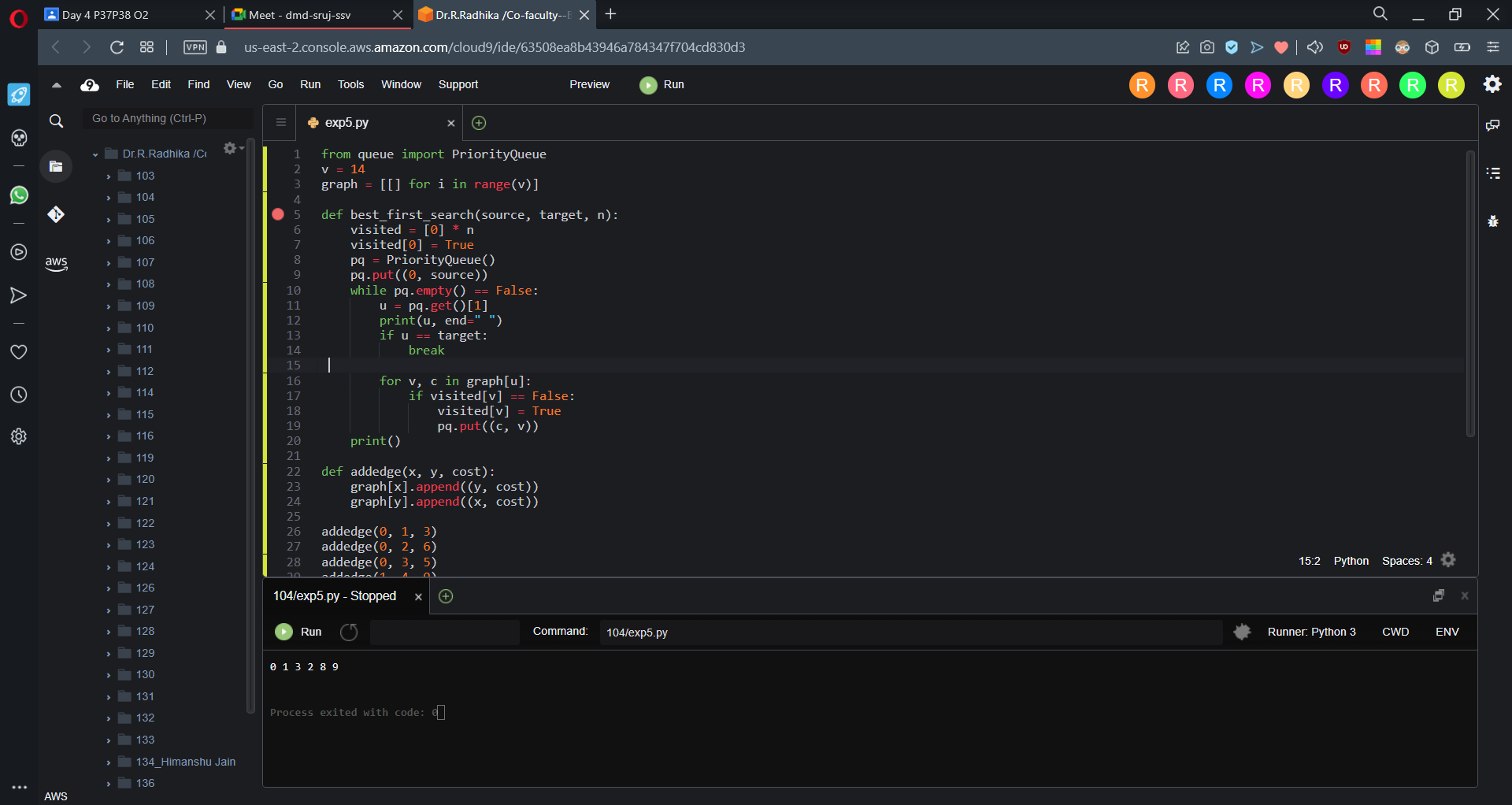
addedge(9, 13, 2)

source = 0

target = 9

best\_first\_search(source, target, v)

**OUTPUT:**



**RESULT:** Hence we have successfully implemented the best first search algorithm using python.