**26.Write a Prolog Program to implement Best First Search algorithm**

**Program:**

from queue import PriorityQueue

v = 14 graph = [[] for i in range(v)]

# Function For Implementing Best First Search

# Gives output path having lowest cost def best\_first\_search(actual\_Src, target, n):

visited = [False] \* n pq = PriorityQueue() pq.put((0, actual\_Src)) visited[actual\_Src] = True while pq.empty() == False:

u = pq.get()[1]

# Displaying the path having lowest cost 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()

# Function for adding edges to graph def addedge(x, y, cost): graph[x].append((y, cost)) graph[y].append((x, cost))

# The nodes shown in above example(by alphabets) are # implemented using integers addedge(x,y,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:**

