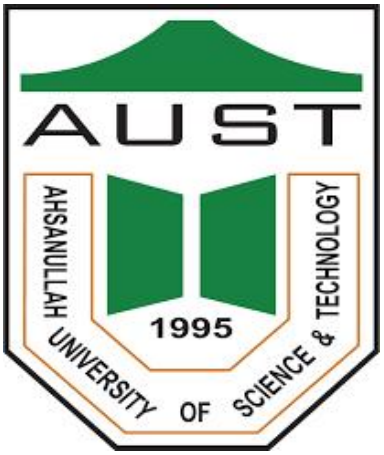

Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Course No : **CSE4108**
Course Title : **Artificial Intelligence Lab**
Assignment No : **3**

Date of Performance : **6.06.23**
Date of Submission : **20.06.23**

Submitted To : **Dr. S.M.A. Al-Mamun**
&
Mr. Raihan Tanvir

Submitted By
Group : **B2**
Name : **MD Fardin Jaman Aranyak**
Id : **190204093**
Section : **B**

Answer:

```
graph = {}
ht ={}

node_count = int(input("Enter the number of nodes in the graph: "))
edge_count = int(input("Enter the number of edges in the graph: "))

for _ in range(edge_count):
    src, dest, weight = input("Enter source node, destination node, and weight (space-
separated): ").split()
    weight = int(weight)



    if src in graph:
        graph[src][dest] = weight
    else:
        graph[src] = {dest: weight}

    if dest in graph:
        graph[dest][src] = weight
    else:
        graph[dest] = {src: weight}

print("Graph:")
print(graph)

for _ in range(node_count):
    node, huristic = input("Enter node, huristic value (space-separated): ").split()
    ht[node] = huristic
print(ht)

src =input("SRC: ")
des =input("DES: ")

def gbfs(graph,ht,src,des):

    visited =[]
    PriorityQueue =[]
    weight={src:0}
```

```
    visited.append(src)
    PriorityQueue.append((ht[src],src))
    while PriorityQueue:
        y=PriorityQueue[0][1];
        PriorityQueue.pop(0);
        visited.append(y)

        if y == des:
            print("Path Cost: ",weight[des])
            print(weight)
            return;

        print("->",y,end=" ")

        edges=graph[y].keys()
        for edge in edges:
            if edge not in visited:
                visited.append(edge)
                PriorityQueue.append((ht[edge],edge))
                PriorityQueue.sort()
                weight[edge] = weight[y] + graph[y][edge]

    print("Goal node not reachable.")

gbfs(graph,ht,src,des)
```