

## AI LAB ASSIGNMENT 7

**NAME:** PRATHAPANI SATWIK

**REG NO:** 20BCD7160

Implement greedy best first search with an example

**CODE:**

```
import java.util.*; public
class Main{

    static class Node{        int
v, weight;        Node(int v,
int weight){        this.v=v;
this.weight=weight;
        }
    }

    static class pathNode{
int node;        pathNode
parent;

        pathNode(int node, pathNode parent){
this.node=node;        this.parent=parent;

        }
    }
```

```

static void addEdge(int u, int v, int weight){

    adj.get(u).add(new Node(v, weight));
}

static List<List<Node>> adj;

private static List<Integer> GBFS(int h[]
, int V, int src, int dest){

    List<pathNode> openList = new ArrayList<>();
    List<pathNode> closeList = new ArrayList<>();

    openList.add(new pathNode(src, null));

    while(!openList.isEmpty()){

        pathNode currentNode = openList.get(0);
int currentIndex = 0;

        for(int i = 0; i < openList.size(); i++){
if(h[openList.get(i).node] <
h[currentNode.node]){            currentNode =
openList.get(i);            currentIndex = i;

```

```

    }
}

openList.remove(currentIndex);
closeList.add(currentNode);

if(currentNode.node == dest){

    List<Integer> path = new ArrayList<>();
    pathNode cur = currentNode;

    while(cur != null){
path.add(cur.node);          cur
= cur.parent;
    }

    Collections.reverse(path);
return path;
}

for(Node node: adj.get(currentNode.node)){
for(pathNode x : openList){

    if(x.node == node.v) continue;
}
}

```

```

        for(pathNode x : closeList){
if(x.node == node.v) continue;
        }
        openList.add(new pathNode(node.v, currentNode));
    }
}

```

```

    return new ArrayList<>();
}
public static void main(String args[]){

```

```

    adj=new ArrayList<>();

```

```

        int V = 10;    for(int i = 0; i
< V; i++)    adj.add(new
ArrayList<>());

```

```

        addEdge(0, 1, 2);
addEdge(0, 2, 1);    addEdge(0,
3, 10);    addEdge(1, 4, 3);
addEdge(1, 5, 2);    addEdge(2,
6, 9);    addEdge(3, 7, 5);
addEdge(3, 8, 2);    addEdge(7,
9, 5);

```

```

    int h[] = {20, 22, 21, 10,

```

25, 24, 30, 5, 12, 0};

List<Integer> path = GBFS(h, V, 0, 9);

for(int i = 0; i < path.size() - 1; i++){

System.out.print(path.get(i)+" --> ");

}

System.out.println(path.get(path.size()-1));

}

}

```
Main.java
1 import java.util.*;
2 public class Main{
3
4     static class Node{
5         int v, weight;
6         Node(int v, int weight){
7             this.v=v;
8             this.weight=weight;
9         }
10    }
11
12    static class pathNode{
13        int node;
14        pathNode parent;
15        pathNode(int node, pathNode parent){
16            this.node=node;
17            this.parent=parent;
18        }
19    }
20
21    static void addEdge(int u, int v, int weight){
22
23        adj.get(u).add(new Node(v, weight));
24    }
25
26    static List<List<Node>> adj;
27
28    private static List<Integer> GBFS(int h[]
29    , int V, int src, int dest){
30
31        List<pathNode> openList = new ArrayList<>();
```

```

Main.java
31 List<pathNode> openList = new ArrayList<>();
32 List<pathNode> closeList = new ArrayList<>();
33
34 openList.add(new pathNode(src, null));
35
36 while(!openList.isEmpty()){
37
38     pathNode currentNode = openList.get(0);
39     int currentIndex = 0;
40
41
42     for(int i = 0; i < openList.size(); i++){
43         if(h[openList.get(i).node] <
44            h[currentNode.node]){
45             currentNode = openList.get(i);
46             currentIndex = i;
47         }
48     }
49
50     openList.remove(currentIndex);
51     closeList.add(currentNode);
52
53     if(currentNode.node == dest){
54
55         List<Integer> path = new ArrayList<>();
56         pathNode cur = currentNode;
57
58         while(cur != null){
59             path.add(cur.node);
60             cur = cur.parent;

```

```

Main.java
60         cur = cur.parent;
61     }
62
63
64     Collections.reverse(path);
65     return path;
66 }
67
68 for(Node node: adj.get(currentNode.node)){
69     for(pathNode x : openList){
70         if(x.node == node.v) continue;
71     }
72     for(pathNode x : closeList){
73         if(x.node == node.v) continue;
74     }
75     openList.add(new pathNode(node.v, currentNode));
76 }
77 }
78
79 return new ArrayList<>();
80 }
81 public static void main(String args[]){
82
83     adj=new ArrayList<>();
84
85     int V = 10;
86     for(int i = 0; i < V; i++)
87         adj.add(new ArrayList<>());
88
89     addEdge(0, 1, 2);
90     addEdge(0, 2, 1);

```

```

Main.java
80     }
81     public static void main(String args[]){
82
83         adj=new ArrayList<>();
84
85         int V = 10;
86         for(int i = 0; i < V; i++){
87             adj.add(new ArrayList<>());
88
89             addEdge(0, 1, 2);
90             addEdge(0, 2, 1);
91             addEdge(0, 3, 10);
92             addEdge(1, 4, 3);
93             addEdge(1, 5, 2);
94             addEdge(2, 6, 9);
95             addEdge(3, 7, 5);
96             addEdge(3, 8, 2);
97             addEdge(7, 9, 5);
98
99             int h[] = {20, 22, 21, 10,
100                 25, 24, 30, 5, 12, 0};
101             List<Integer> path = GBFS(h, V, 0, 9);
102             for(int i = 0; i < path.size() - 1; i++){
103                 System.out.print(path.get(i)+" --> ");
104             }
105             System.out.println(path.get(path.size()-1));
106
107         }
108     }
109 }
110

```

**OUTPUT:**

```

0 --> 3 --> 7 --> 9

...Program finished with exit code 0
Press ENTER to exit console.

```