AI LAB ASSIGNMENT 7

NAME: PRATHAPANI SATWIKA

REG NO: 20BCD7160

Implement greedy best first search with an example

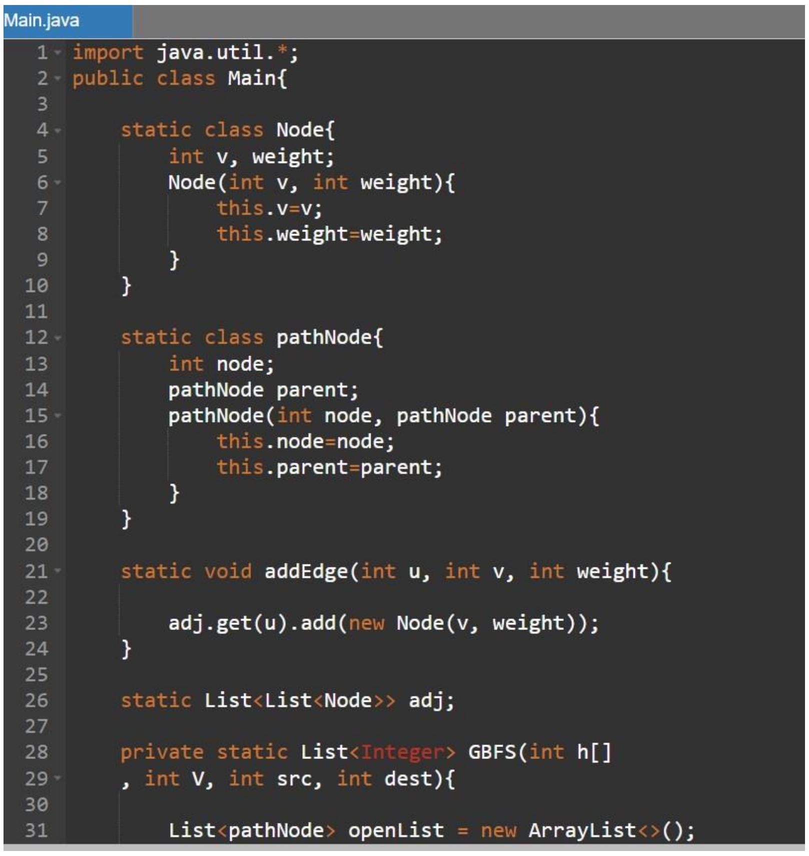
CODE:

```
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;

      }
   }
```

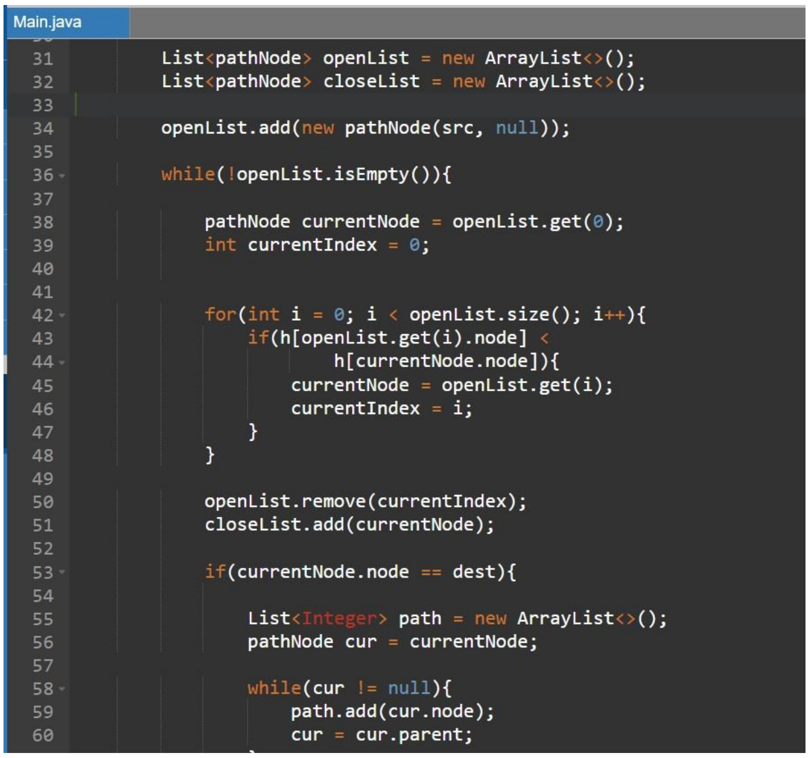
```
    static void addEdge(int u, int v, int weight){

        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;
            }
```

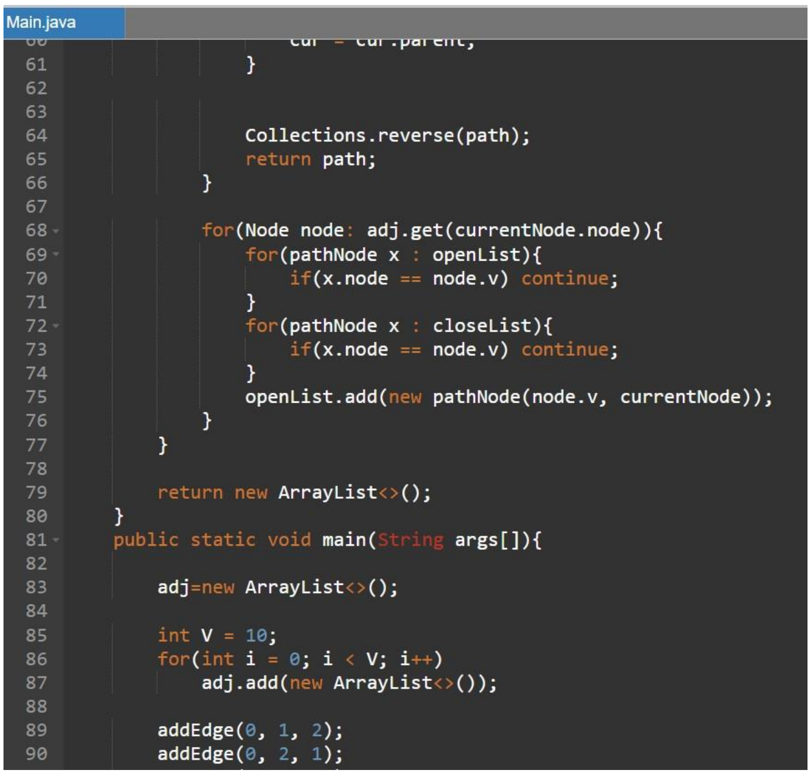
```
                for(pathNode x : closeList){
if(x.node == node.v) continue;
                }
                openList.add(new pathNode(node.v, currentNode));
            }
        }

        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));
    }
}
```



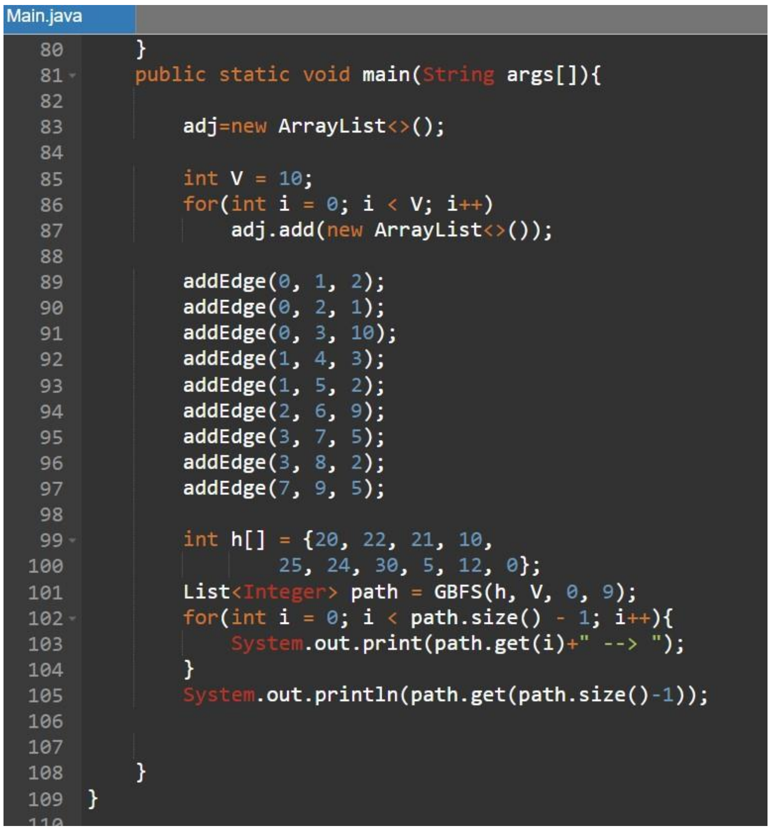
```
Main.java
import java.util.*;
public class Main{

    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<>();
```



```java
        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;
```

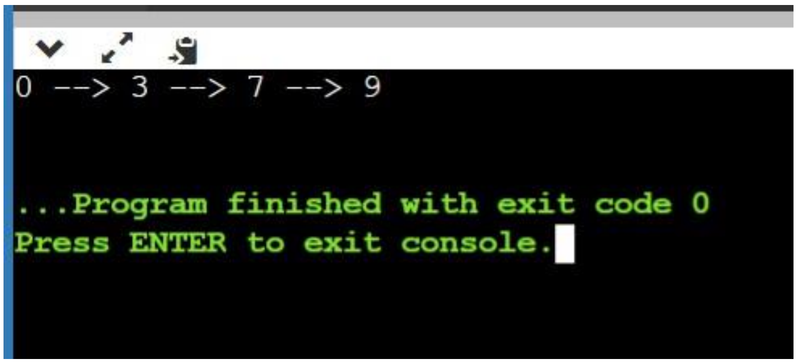


```java
                }


                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);
```


Main.java

```java
    }
    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));


    }
}
```

OUTPUT:



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


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