Write a program to Search a list of items using best first search implemented in JAVA

Lab Assignment-3

CSE3002 : Artificial Intelligence

Submitted by:

Jayakumar MHK (18BCE7031)

Under the Guidance of

Prof. Manomita Chakraborty

SCOPE

VIT-AP

Task:
Write a program to Search a list of items using best first search
Solution:
I have used PriorityQueue to solve this problem using Java to get the cost-effective(cheapest) path from source node to the destination node

Below is the source code of the same.

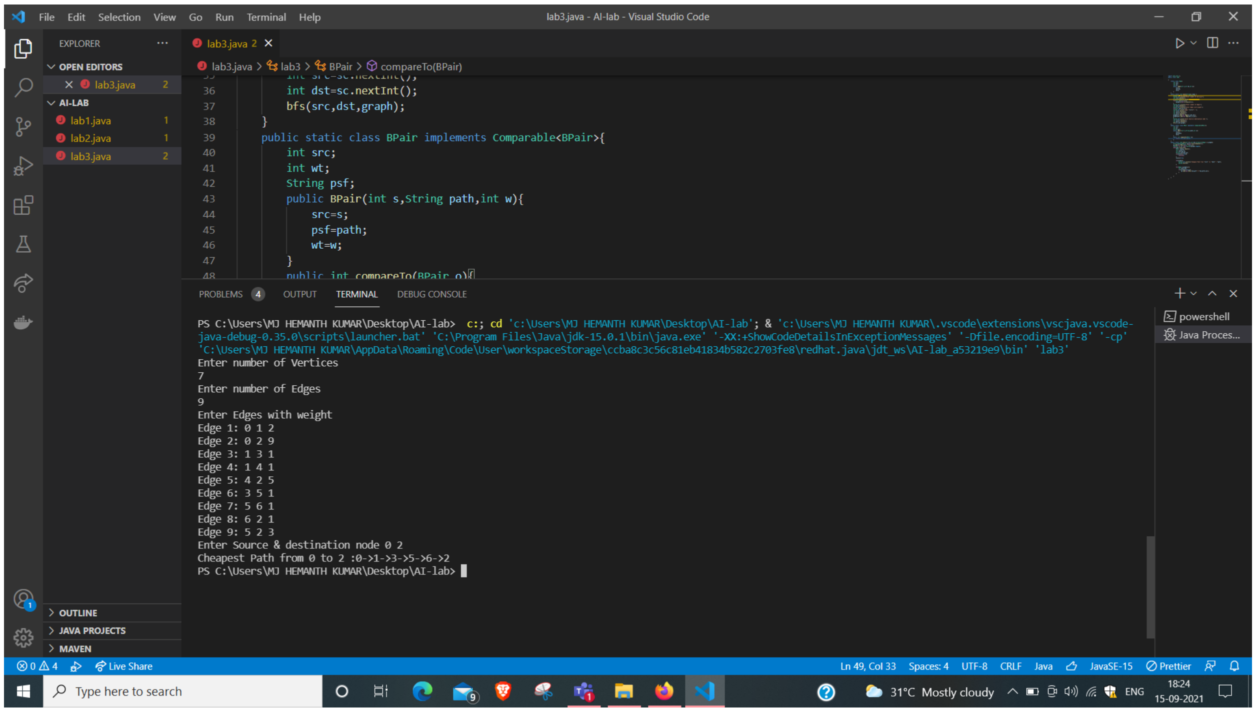
```
import java.util.*;
public class lab3

{
    static class Edge{
        int src;
        int nbr;
        int wt;
        public Edge(int s,int nb,int w){
            src=s;
            nbr=nb;
            wt=w;
        }
    }
    public static void main(String[] args) {
        Scanner sc=new Scanner (System.in);
        System.out.println("Enter number of Vertices");
        int v=sc.nextInt();
        ArrayList<Edge>[] graph=new ArrayList[v];
        for(int i=0;i<v;i++){
            graph[i]=new ArrayList<>();
        }
        System.out.println("Enter number of Edges");
        int e=sc.nextInt();
        System.out.println("Enter Edges with weight");
        for(int i=0;i<e;i++){
        System.out.print("Edge "+(i+1)+": ");
        int sr=sc.nextInt();
        int nbr=sc.nextInt();
        int wt=sc.nextInt();
        graph[sr].add(new Edge(sr,nbr,wt));
        graph[nbr].add(new Edge(nbr,sr,wt));
        }
        System.out.print("Enter Source & destination node ");
        int src=sc.nextInt();
        int dst=sc.nextInt();
        bfs(src,dst,graph);
    }
    public static class BPair implements Comparable<BPair>{
        int src;
        int wt;
        String psf;
        public BPair(int s,String path,int w){
            src=s;
            psf=path;
            wt=w;
        }
        public int compareTo(BPair o){
            return this.wt-o.wt;
        }
    }
    public static void bfs(int src,int dst,ArrayList<Edge> [] graph){
        PriorityQueue<BPair> qu=new PriorityQueue<>();
        qu.add(new BPair(src,""+src,0));
        boolean [] vis =new boolean[graph.length];
        while(qu.size()>0){
            BPair rem=qu.remove();
            int s=rem.src;
            int wsf=rem.wt;
            String psf=rem.psf;
            if(vis[s]==true){
                continue;
            }
            vis[s]=true;

            if(s==dst){
                System.out.println("Cheapest Path from "+src+" to "+dst+"
:"+psf);
                System.exit(0);
            }
            for(Edge e:graph[s]){
                int nb=e.nbr;
                if(vis[nb]==false){
                    qu.add(new BPair(nb,psf+"->"+nb,wsf+e.wt));
                }
            }


        }
    }
}
```

Output below :



```
        int src=sc.nextInt();
        int dst=sc.nextInt();
        bfs(src,dst,graph);
    }
    public static class BPair implements Comparable<BPair>{
        int src;
        int wt;
        String psf;
        public BPair(int s,String path,int w){
            src=s;
            psf=path;
            wt=w;
        }
        public int compareTo(BPair o){
```

```
PS C:\Users\MJ HEMANTH KUMAR\Desktop\AI-lab>  c:; cd 'c:\Users\MJ HEMANTH KUMAR\Desktop\AI-lab'; & 'c:\Users\MJ HEMANTH KUMAR\.vscode\extensions\vscjava.vscode-java-debug-0.35.0\scripts\launcher.bat' 'C:\Program Files\Java\jdk-15.0.1\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\MJ HEMANTH KUMAR\AppData\Roaming\Code\User\workspaceStorage\ccba8c3c56c81eb41834b582c2703fe8\redhat.java\jdt_ws\AI-lab_a53219e9\bin' 'lab3'
Enter number of Vertices
7
Enter number of Edges
9
Enter Edges with weight
Edge 1: 0 1 2
Edge 2: 0 2 9
Edge 3: 1 3 1
Edge 4: 1 4 1
Edge 5: 4 2 5
Edge 6: 3 5 1
Edge 7: 5 6 1
Edge 8: 6 2 1
Edge 9: 5 2 3
Enter Source & destination node 0 2
Cheapest Path from 0 to 2 :0->1->3->5->6->2
PS C:\Users\MJ HEMANTH KUMAR\Desktop\AI-lab>
```