VIT
AP
AMARAVATI
www.vitap.ac.in

Write a program to solve A* Algorithm Using JAVA

Lab Assignment-7

CSE3002 : Artificial Intelligence

Submitted by:

Jayakumar MHK (18BCE7031)

Under the Guidance of

Prof. Manomita Chakraborty

SCOPE

VIT-AP

Task:

Write a program to solve the A* Algorithm

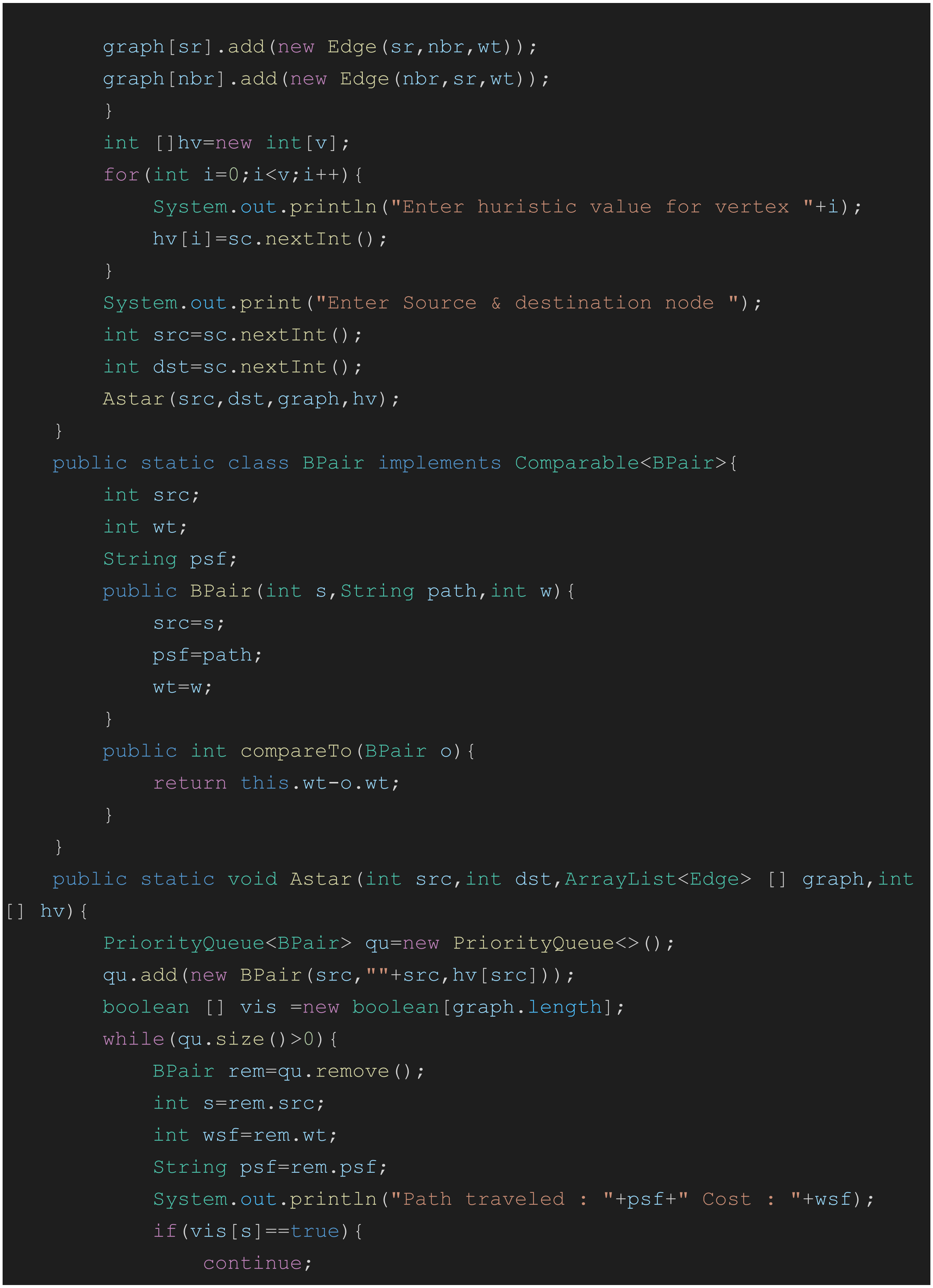
Solution:

I have used priority queue data structure which finds the minimal cost to the destination node using the A* algorithm which computes the value based on the summation of the estimated cost and the actual cost to solve this problem using Java

Below is the source code of the same.

```
import java.util.*;
public class lab7
{
    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();
```



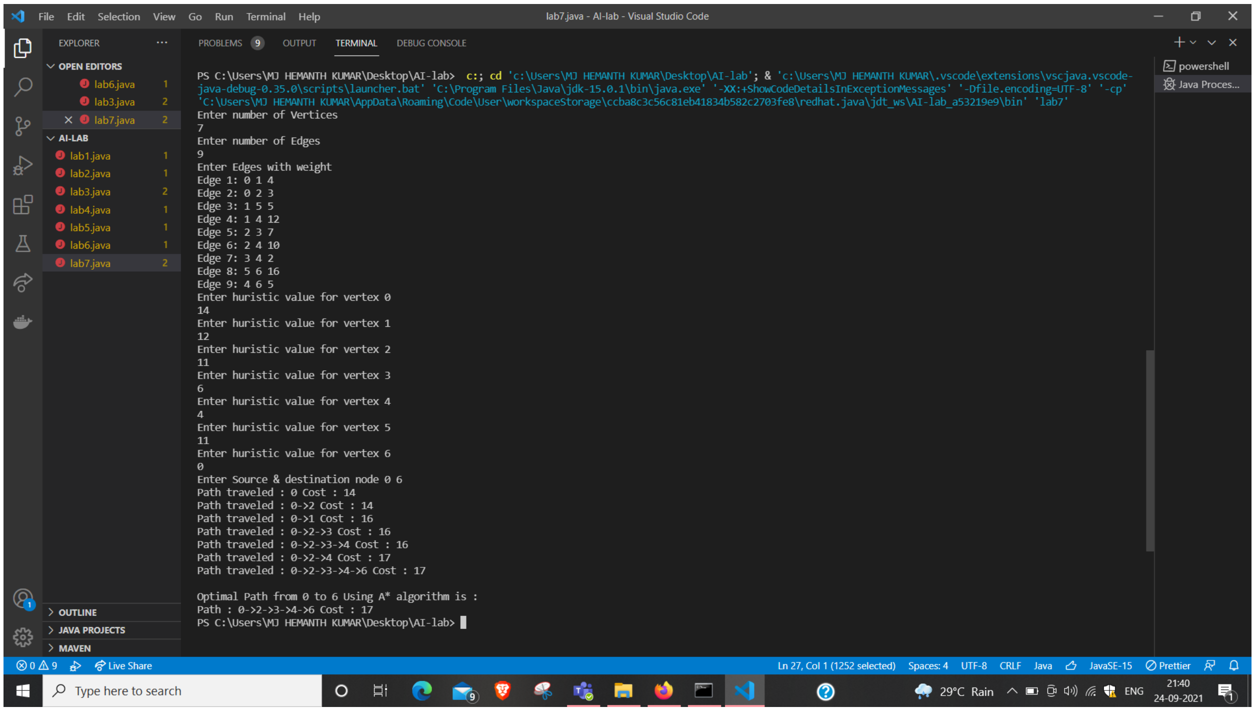
```
        graph[sr].add(new Edge(sr,nbr,wt));
        graph[nbr].add(new Edge(nbr,sr,wt));
        }
        int []hv=new int[v];
        for(int i=0;i<v;i++){
            System.out.println("Enter huristic value for vertex "+i);
            hv[i]=sc.nextInt();
        }
        System.out.print("Enter Source & destination node ");
        int src=sc.nextInt();
        int dst=sc.nextInt();
        Astar(src,dst,graph,hv);
    }
    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 Astar(int src,int dst,ArrayList<Edge> [] graph,int
[] hv){
        PriorityQueue<BPair> qu=new PriorityQueue<>();
        qu.add(new BPair(src,""+src,hv[src]));
        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;
            System.out.println("Path traveled : "+psf+" Cost : "+wsf);
            if(vis[s]==true){
                continue;
                }
                vis[s]=true;

                if(s==dst){
                     System.out.println("\nOptimal Path from "+src+" to "+dst+"
Using A* algorithm is :\nPath : "+psf+" Cost : "+wsf);
                     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+-hv[s]+hv[nb]));
                     }
                }


           }
      }
}
```

Output below :



```
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' 'lab7'
Enter number of Vertices
7
Enter number of Edges
9
Enter Edges with weight
Edge 1: 0 1 4
Edge 2: 0 2 3
Edge 3: 1 5 5
Edge 4: 1 4 12
Edge 5: 2 3 7
Edge 6: 2 4 10
Edge 7: 3 4 2
Edge 8: 5 6 16
Edge 9: 4 6 5
Enter huristic value for vertex 0
14
Enter huristic value for vertex 1
12
Enter huristic value for vertex 2
11
Enter huristic value for vertex 3
6
Enter huristic value for vertex 4
4
Enter huristic value for vertex 5
11
Enter huristic value for vertex 6
0
Enter Source & destination node 0 6
Path traveled : 0 Cost : 14
Path traveled : 0->2 Cost : 14
Path traveled : 0->1 Cost : 16
Path traveled : 0->2->3 Cost : 16
Path traveled : 0->2->3->4 Cost : 16
Path traveled : 0->2->4 Cost : 17
Path traveled : 0->2->3->4->6 Cost : 17

Optimal Path from 0 to 6 Using A* algorithm is :
Path : 0->2->3->4->6 Cost : 17
PS C:\Users\MJ HEMANTH KUMAR\Desktop\AI-lab> 
```