VIT AP
AMARAVATI
www.vitap.ac.in

# Write a program to solve the traveling salesman problem implemented in JAVA

## Lab Assignment-6

## 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 traveling salesman problem

**Solution:**

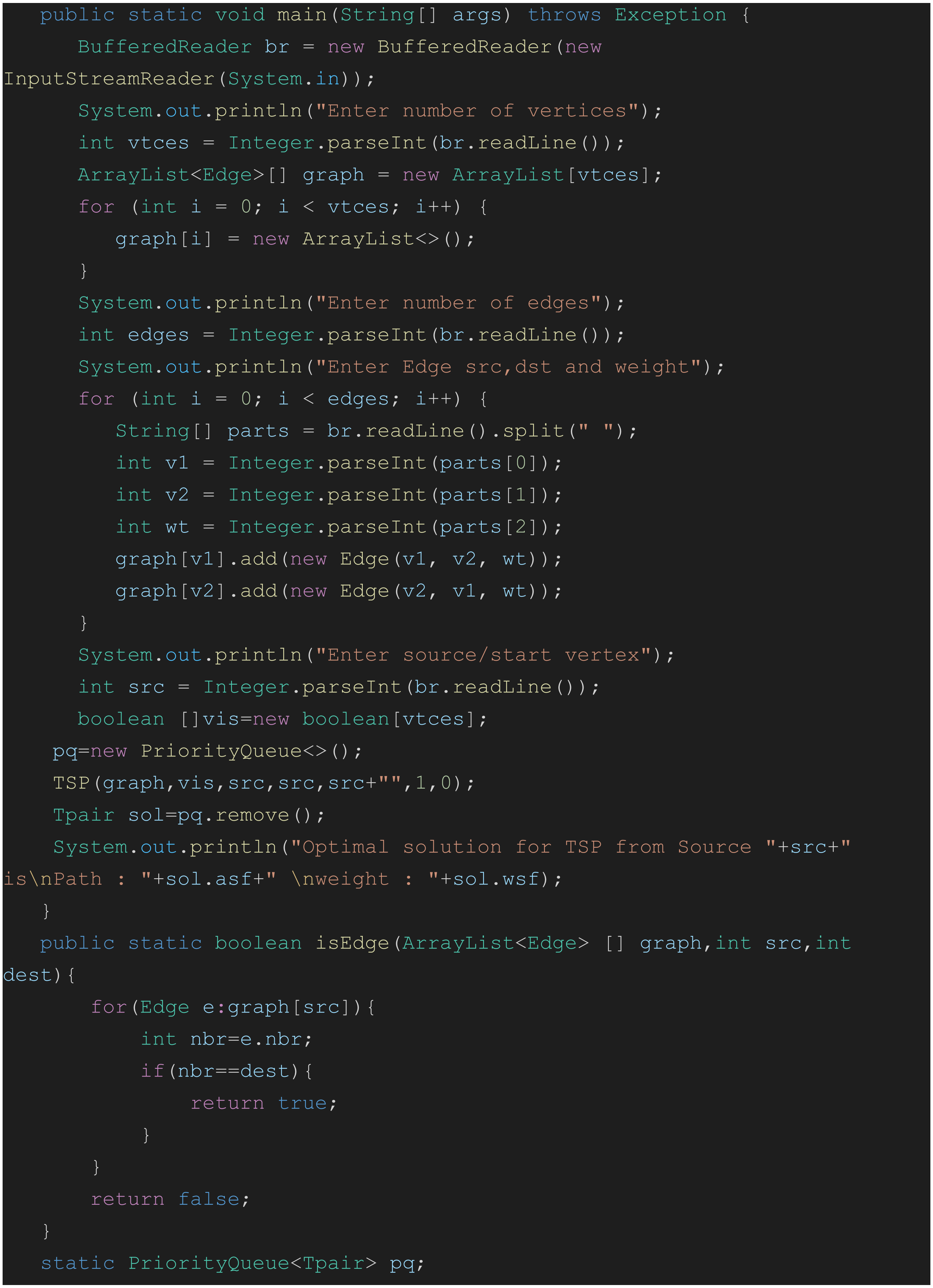
I have used the DFS algorithm to get all hamiltonian Cycle then added the results obtain to the heap to get an optimal solution which gives the complete path from a start point visiting all nodes and returning back to start with minimum weight traveled and not visiting a node more than once to solve this problem using Java

Below is the source code of the same.

```java
import java.io.*;
import java.util.*;

public class lab6 {
   static class Edge {
      int src;
      int nbr;
      int wt;

      Edge(int src, int nbr, int wt) {
         this.src = src;
         this.nbr = nbr;
         this.wt = wt;
      }
   }
    static class Tpair implements Comparable<Tpair>{
        String asf;
        int wsf;
        Tpair(String path,int weight) {
            asf=path;
            wsf=weight;
        }
        public int compareTo(Tpair o){
            return this.wsf-o.wsf;
        }
    }
```

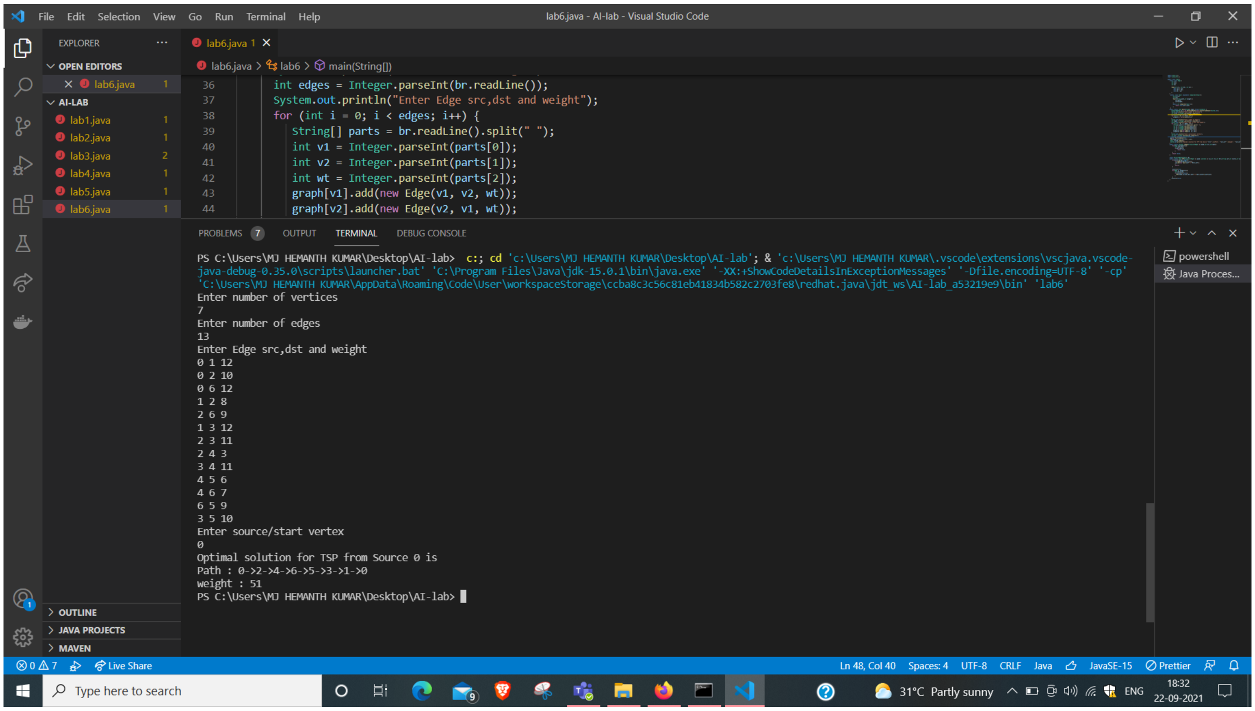


```
   public static void main(String[] args) throws Exception {
      BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
      System.out.println("Enter number of vertices");
      int vtces = Integer.parseInt(br.readLine());
      ArrayList<Edge>[] graph = new ArrayList[vtces];
      for (int i = 0; i < vtces; i++) {
         graph[i] = new ArrayList<>();
      }
      System.out.println("Enter number of edges");
      int edges = Integer.parseInt(br.readLine());
      System.out.println("Enter Edge src,dst and weight");
      for (int i = 0; i < edges; i++) {
         String[] parts = br.readLine().split(" ");
         int v1 = Integer.parseInt(parts[0]);
         int v2 = Integer.parseInt(parts[1]);
         int wt = Integer.parseInt(parts[2]);
         graph[v1].add(new Edge(v1, v2, wt));
         graph[v2].add(new Edge(v2, v1, wt));
      }
      System.out.println("Enter source/start vertex");
      int src = Integer.parseInt(br.readLine());
      boolean []vis=new boolean[vtces];
    pq=new PriorityQueue<>();
    TSP(graph,vis,src,src,src+"",1,0);
    Tpair sol=pq.remove();
    System.out.println("Optimal solution for TSP from Source "+src+"
is\nPath : "+sol.asf+" \nweight : "+sol.wsf);
   }
   public static boolean isEdge(ArrayList<Edge> [] graph,int src,int
dest){
       for(Edge e:graph[src]){
           int nbr=e.nbr;
           if(nbr==dest){
               return true;
           }
       }
       return false;
   }
   static PriorityQueue<Tpair> pq;
```

```
    public static void TSP(ArrayList<Edge> [] graph ,boolean [] vis,int
src,int Osrc,String asf,int visvtx,int wsf){
        if(visvtx==graph.length){
            if(isEdge(graph,Osrc,src)){
             pq.add(new Tpair(asf+"->"+Osrc,wsf));
            }
            return;
        }

        vis[src]=true;
        for(Edge e: graph[src]){
            int nbr=e.nbr;
            if(vis[nbr]==false){
              TSP(graph,vis,nbr,Osrc,asf+"->"+nbr,visvtx+1,wsf+e.wt);
            }
        }
       vis[src]=false;
    }
}
```

# Output below :



```
        int edges = Integer.parseInt(br.readLine());
        System.out.println("Enter Edge src,dst and weight");
        for (int i = 0; i < edges; i++) {
            String[] parts = br.readLine().split(" ");
            int v1 = Integer.parseInt(parts[0]);
            int v2 = Integer.parseInt(parts[1]);
            int wt = Integer.parseInt(parts[2]);
            graph[v1].add(new Edge(v1, v2, wt));
            graph[v2].add(new Edge(v2, v1, wt));
```

```
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' 'lab6'
Enter number of vertices
7
Enter number of edges
13
Enter Edge src,dst and weight
0 1 12
0 2 10
0 6 12
1 2 8
2 6 9
1 3 12
2 3 11
2 4 3
3 4 11
4 5 6
4 6 7
6 5 9
3 5 10
Enter source/start vertex
0
Optimal solution for TSP from Source 0 is
Path : 0->2->4->6->5->3->1->0
weight : 51
PS C:\Users\MJ HEMANTH KUMAR\Desktop\AI-lab> 
```