Experiment No: 10 b

Write Java programs to Implement **Travelling Sales Person problem** using Dynamic programming.

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

static int graph[][]= new int[10][10];
static int visited[]= new int[10];
static int cost=0,n;
	public static void main(String[] args) {
		int i,j;
		Scanner sc= new Scanner(System.in);
		System.out.println("Enter the number of cities");
		n= sc.nextInt();
		System.out.println("Enter the weighted graph");
		for(i=1;i<=n;i++)
		{
			visited[i]=0;
		  for(j=1;j<=n;j++)
		   graph[i][j]=sc.nextInt();
		}
		System.out.println("The weighted graph is ");
		for(i=1;i<=n;i++)
		{
			for(j=1;j<=n;j++)
				System.out.print(graph[i][j] +"\t");
			System.out.println();

		}
		System.out.println("The path is ");
		mincost(1);
		System.out.println("\n The total cost of the tour is " + cost);
			}
	public static void mincost(int city)
	{
		int i,city_no;
		visited[city]=1;
		System.out.print(city+ " ->");
		city_no= least(city);
		if(city_no==999)
		{
			city_no=1;
			System.out.print("" + city_no);
			cost += graph[city][city_no];
			return;
		}
		mincost(city_no);
	}
	public static int least(int c)
	{
		int i,min_node=999,new_min=0,min=999;
		for(i=1;i<=n;i++)
		{
			if(graph[c][i]!=0 && visited[i]==0)
			{
				if(graph[c][i]< min)
```

```
                        {
                                min= graph[i][1]+ graph[c][i];
                                new_min= graph[c][i];
                                min_node=i;
                        }
                }
        }
        if(min!=999)
        cost+=new_min;
        return min_node;
        }

}
```

/* output

```
Enter the number of cities

4
Enter the weighted graph
0 16 11 6
8 0 13 16
4 7 0 9
5 12 2 0
The weighted graph is
0       16      11      6
8       0       13      16
4       7       0       9
5       12      2       0
The path is
1 ->4 ->3 ->2 ->1
 The total cost of the tour is 23
*/
```