**Prim’s Algorithm Code:**

import java.io.\*;

import java.lang.\*;

import java.util.\*;

class MST {

int cost = 0;

int minKey(int key[], Boolean mstSet[], int vertices){

int min = Integer.MAX\_VALUE, mini = -1;

for (int v = 0; v < vertices; v++)

if (mstSet[v] == false && key[v] < min) {

min = key[v];

mini = v;

}

return mini;

}

void printMST(int parent[], int graph[][], int vertices){

System.out.println("Edge \tWeight");

for (int i = 1; i < vertices; i++){

System.out.println(parent[i] + " - " + i + "\t" + graph[i][parent[i]]);

cost += graph[i][parent[i]];

}

System.out.println("Total cost is " + cost);

}

void primMST(int graph[][], int vertices)

{

int parent[] = new int[vertices];

int key[] = new int[vertices];

Boolean mstSet[] = new Boolean[vertices];

for (int i = 0; i < vertices; i++) {

key[i] = Integer.MAX\_VALUE;

mstSet[i] = false;

}

key[0] = 0;

parent[0] = -1;

for (int count = 0; count < vertices-1; count++) {

int u = minKey(key, mstSet, vertices);

mstSet[u] = true;

for (int v = 0; v < vertices; v++)

if (graph[u][v] != 0 && mstSet[v] == false

&& graph[u][v] < key[v]) {

parent[v] = u;

key[v] = graph[u][v];

}

}

printMST(parent, graph, vertices);

}

public static void main(String[] args)

{

MST t = new MST();

Scanner sc = new Scanner(System.in);

System.out.println("Enter number of vertices");

int vertices = sc.nextInt();

int graph[][] = new int[vertices][vertices];

System.out.println("Enter the adjacency matrix");

for(int i=0; i<vertices; i++){

for(int j=0; j<vertices; j++){

graph[i][j] = sc.nextInt();

}

}

t.primMST(graph, vertices);

}

}

**Output:**

Enter number of vertices

5

Enter the adjacency matrix

0 2 0 6 0

2 0 3 8 5

0 3 0 0 7

6 8 0 0 9

0 5 7 9 0

Edge Weight

0 - 1 2

1 - 2 3

0 - 3 6

1 - 4 5

Total cost is 16

\*\* Process exited - Return Code: 0 \*\*