prims**(** vertex v **)**

**{** mark **v** as visited and

include in spanning tree**;**

**while(** there are unvisited vertices **)**

**{** find the least**-**cost edge**(**v**,** u**)**

from some visited vertex **v**

to some unvisited vertex **u;**

mark**(** u **);**

add u and edge**(**v**,** u**)** to spanning tree**;**

**}** // end while

**}**

From starting point (vertex) we look around for the cheapest edge. We move to the new vertex and we add edge together with vertex to the spanning tree. Then we continue from current vertex, we search for cheapest edge, but we also consider if there is cheaper edges from already visited vertices. We continue until there is no unvisited vertex.

There is only one solution. No matter from where we start, we will get only one solution.

In both Dijkstra and Minimum Spanning Tree, the primary subject is to notice the shortest edge to the connected vertices. –Aimo Suikkanen; November 2014; Denmark; Zealand; Ballerup.

**Dijkstra shortest path algorithm** is preferable when we want to know the cheapest route from starting node to specific one.

**Minimum spanning tree algorithm** is convenient when we want to know the cheapest route through all nodes.

Big O notation for minimum spanning tree is (N2)