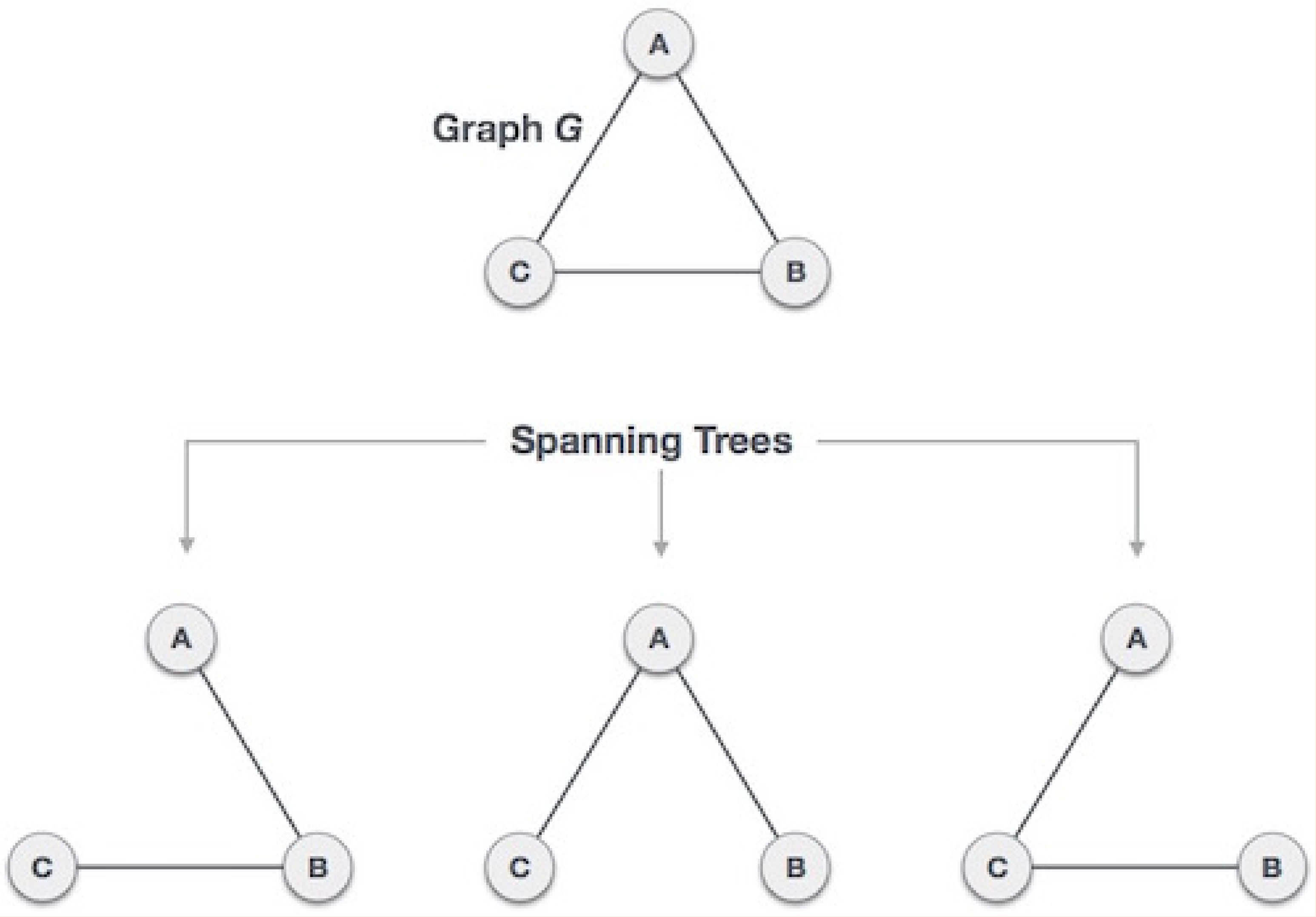
Minimum Spanning Tree

Important Terms

Spanning Tree

Given an undirected and connected graph, a spanning tree of the graph is a tree that includes every vertex of the graph

- It is a subgraph of the graph, ie every edge in the tree belongs to graph



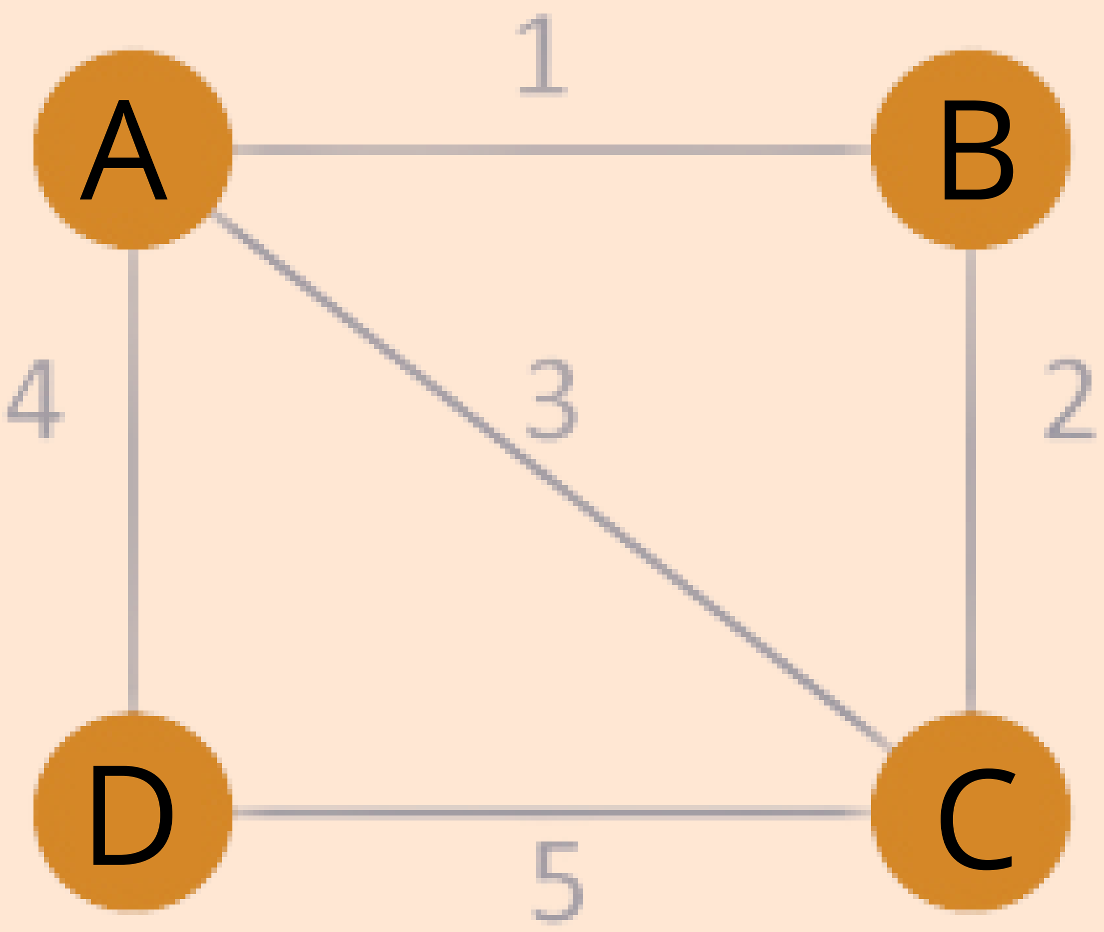
Minimum Spanning Tree

- In a connected, weighted, undirected graph, the cost of the spanning tree is the sum of the weights of all the edges in the tree.

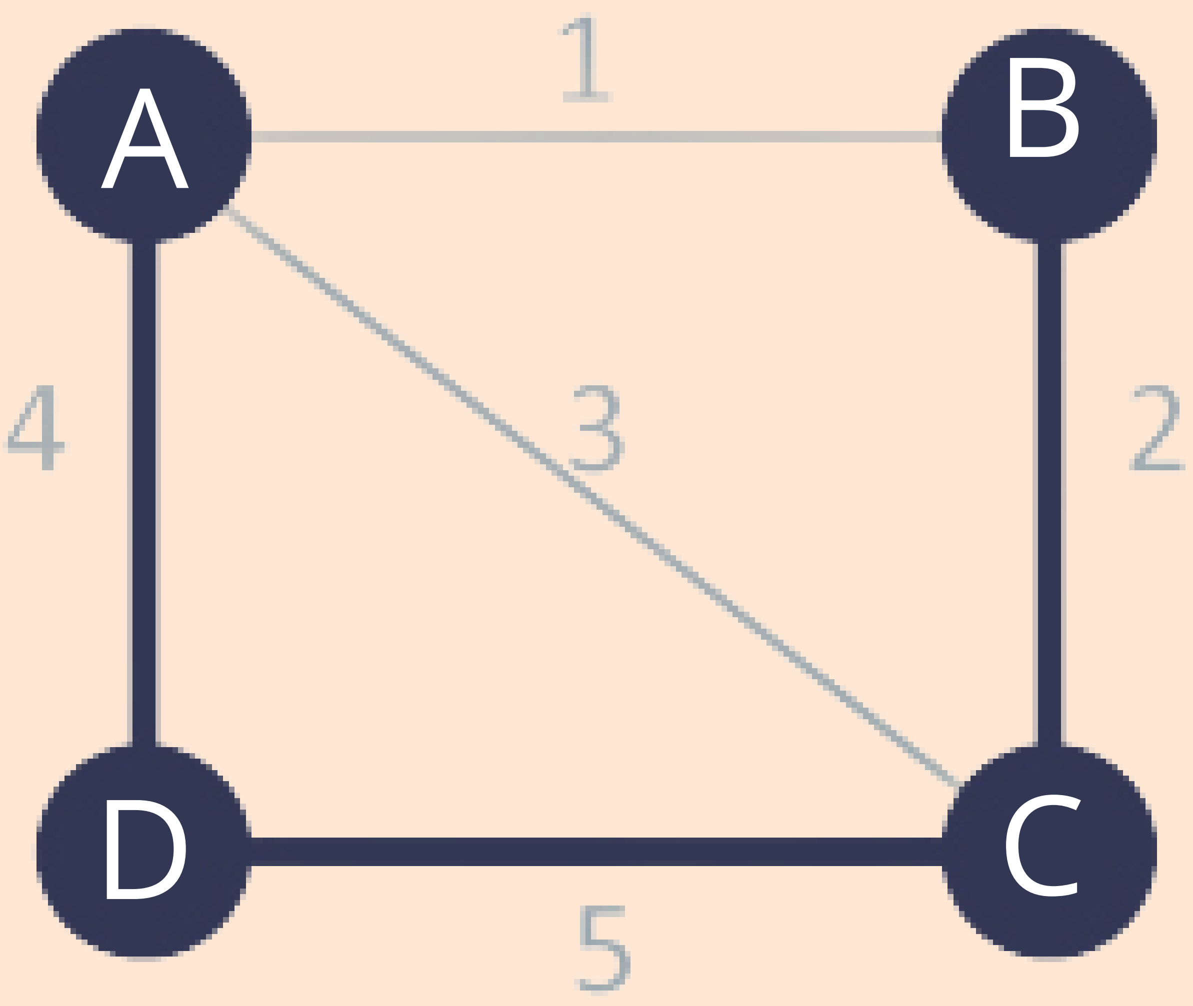
- There can be many spanning trees.

- **Minimum spanning tree** is the spanning tree where the cost is minimum among all the spanning trees.

Suppose we have an undirected graph with 4 vertices

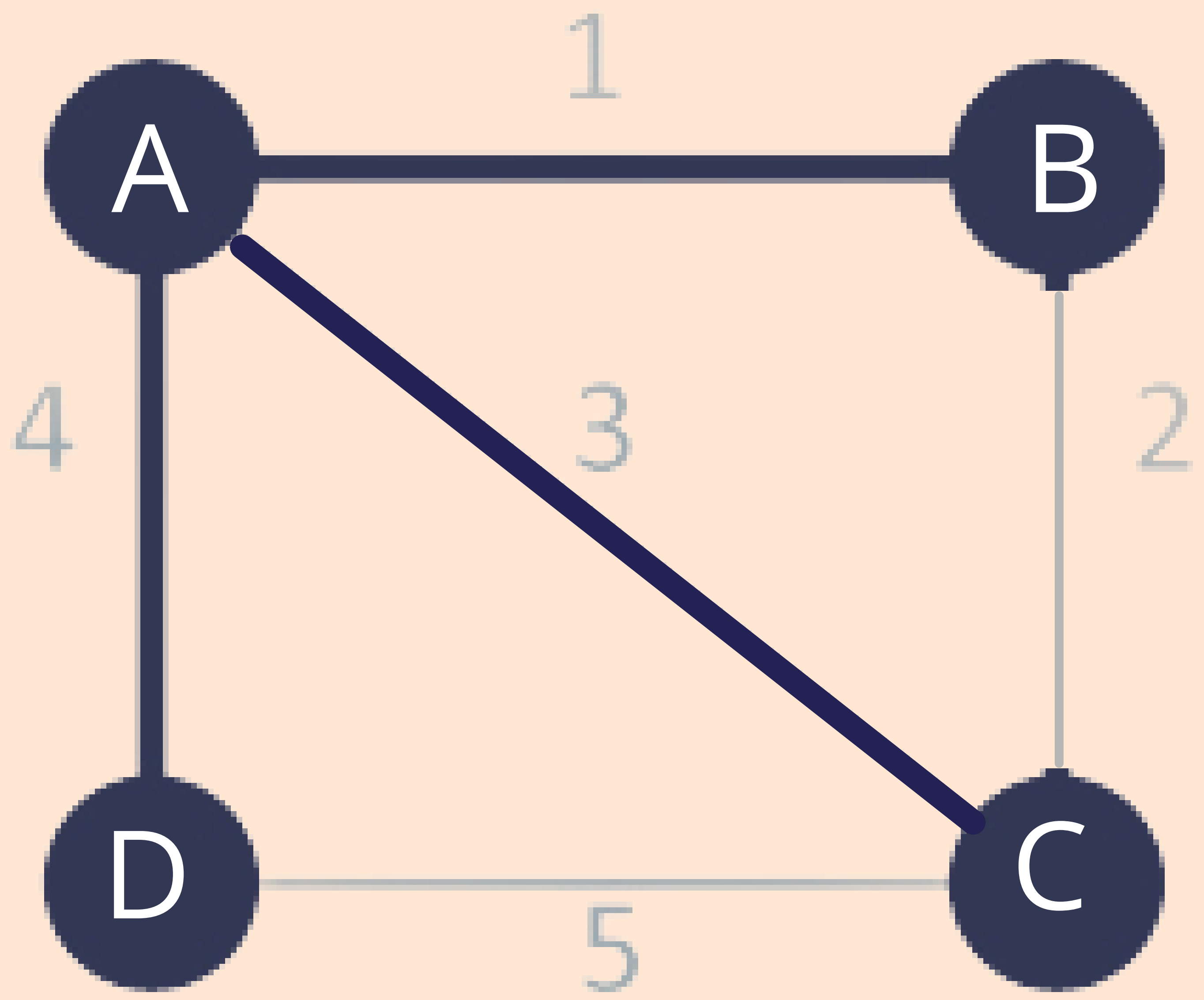


Undirected Graph

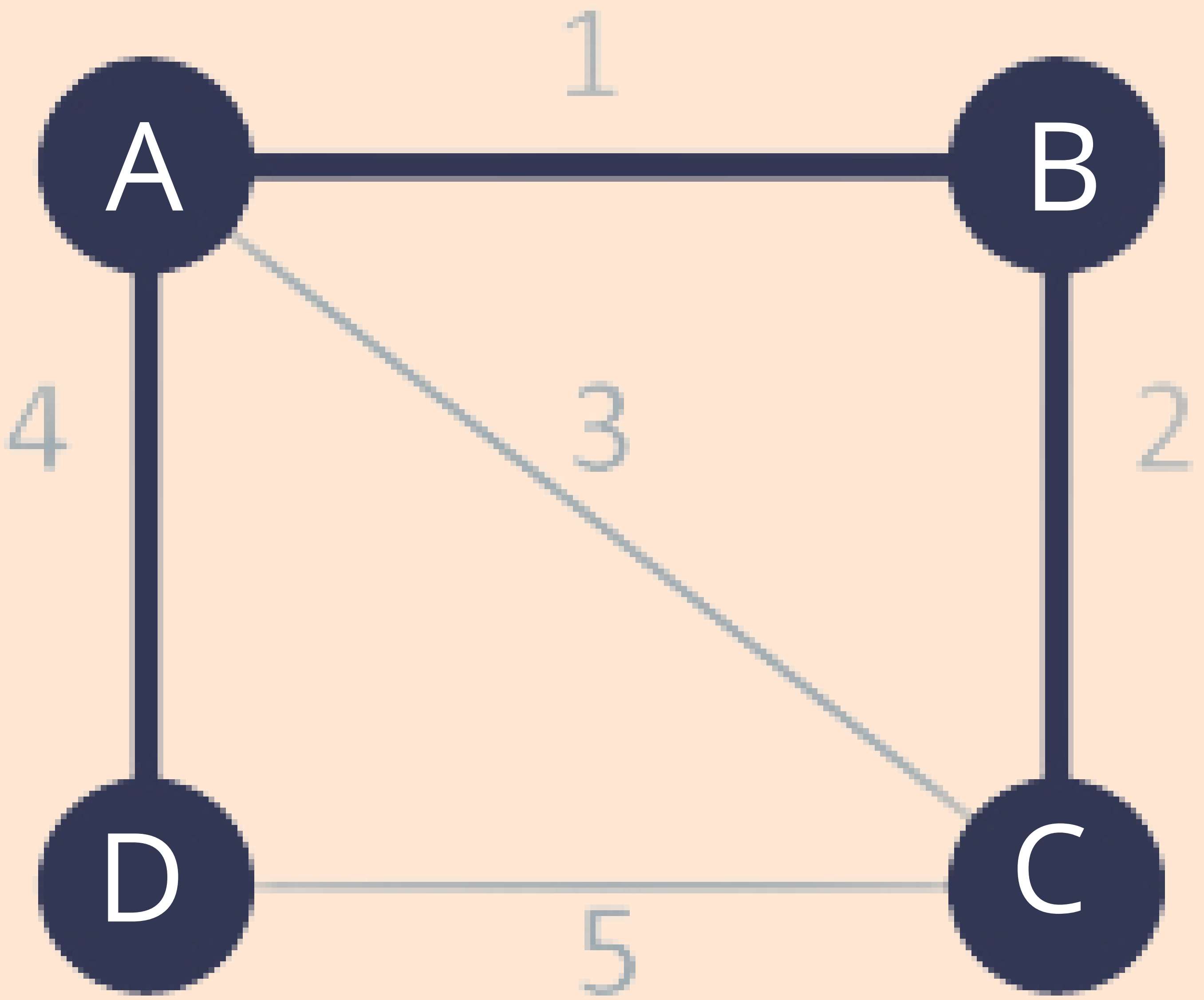


Spanning Tree

Cost= 11 (4+5+2)



Spanning Tree
Cost= 8 (4+3+1)



Minimum Spanning Tree
Cost= 7 (4+1+2)

Application of MST problem

✓ Network design

N stations to be linked using a communication network & laying of communication links between any two stations involves a cost.
The ideal solution would be to extract a subgraph termed as minimum cost spanning tree.

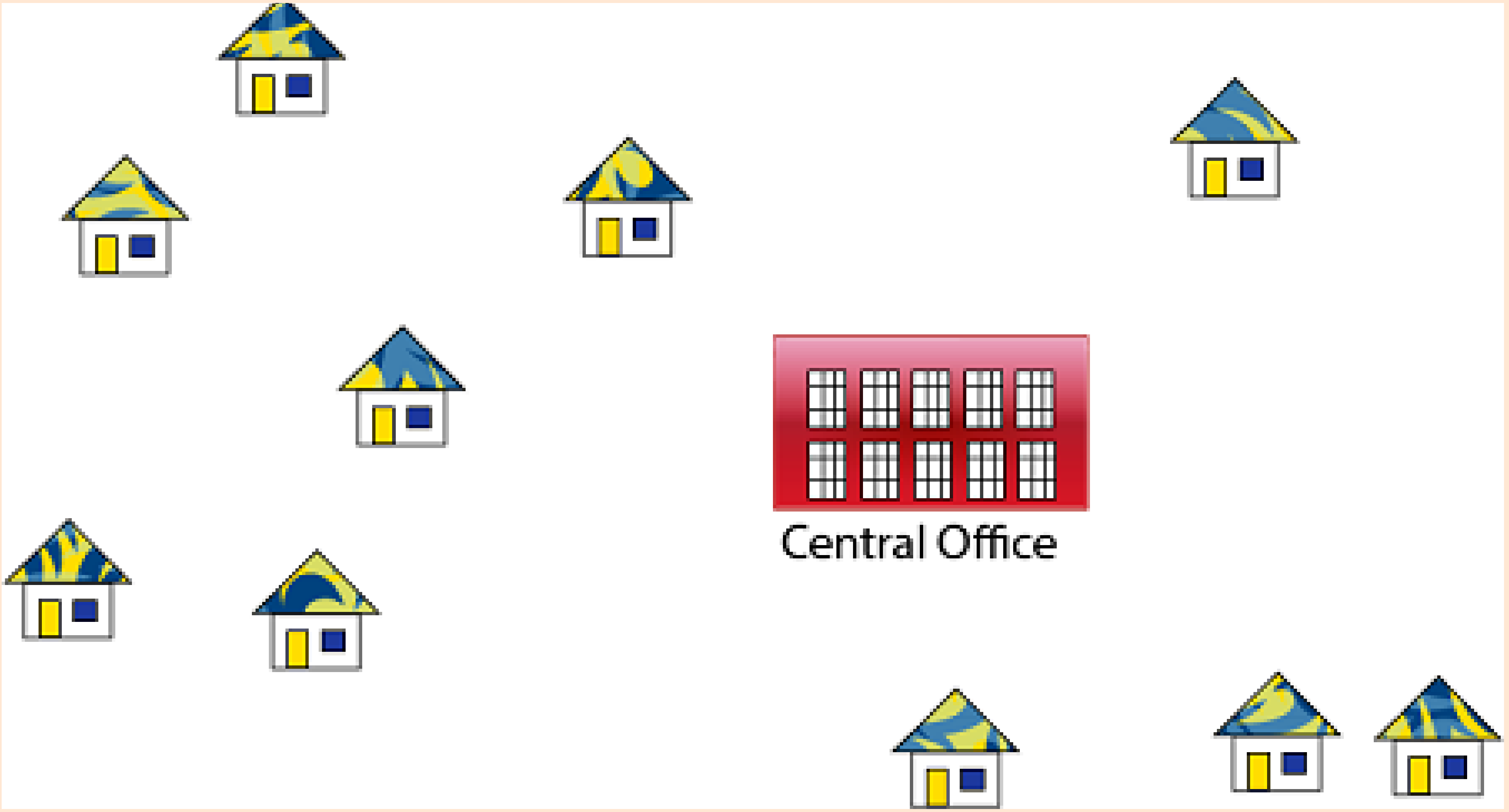
✓ Connecting Cities

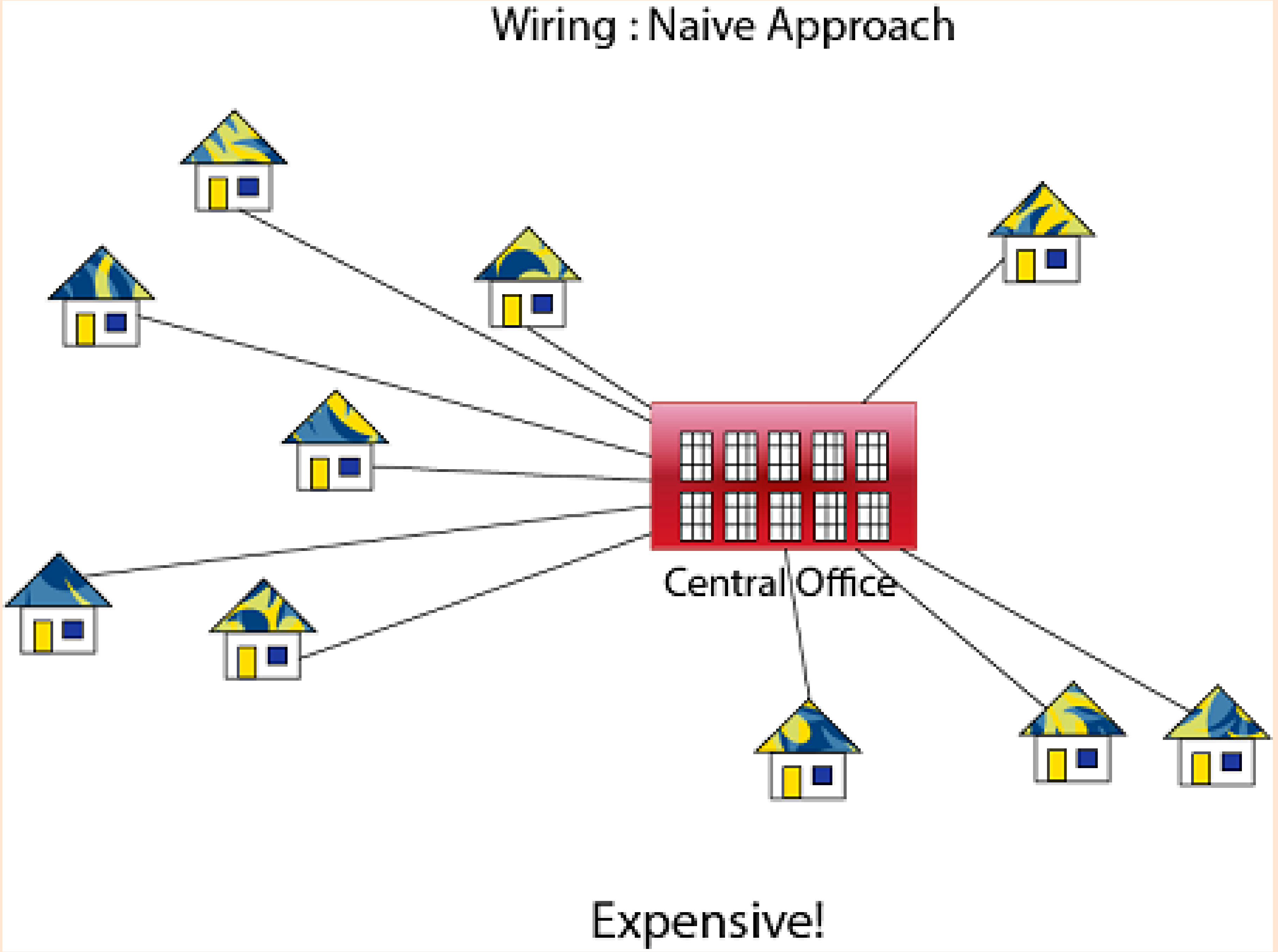
To construct highways or railroads spanning several cities then we can use the concept of minimum spanning trees.

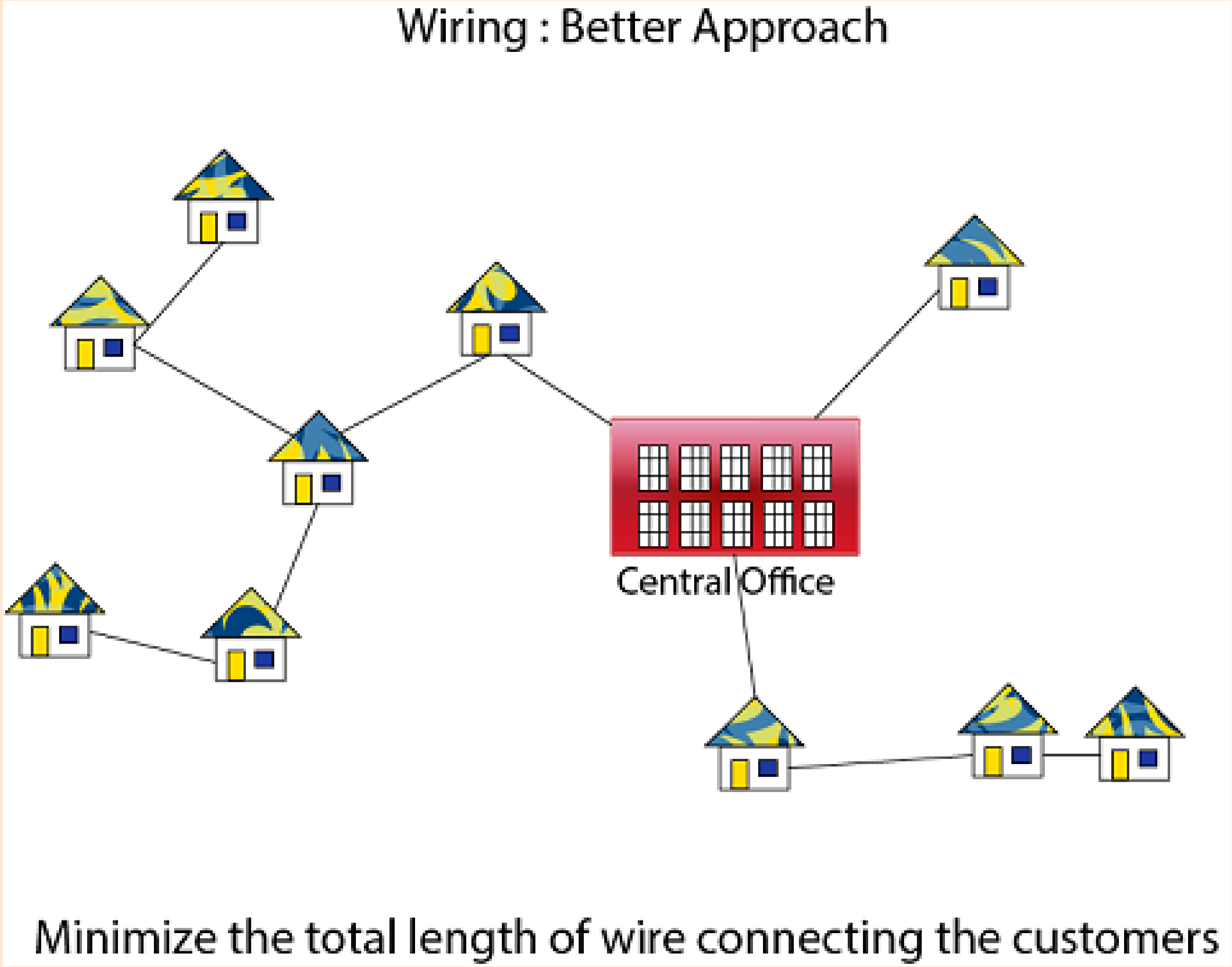
Supply of power lines

To supply a set of houses with

- Electric Power
- Water
- Telephone lines
- Sewage lines




Wiring : Naive Approach
Central Office
Expensive!


Wiring : Better Approach
Central Office
Minimize the total length of wire connecting the customers

Kruskal's algorithm

- Kruskal’s Algorithm builds the spanning tree by adding edges one by one into a growing spanning tree.
- Kruskal's algorithm follows **greedy approach** as in each iteration it finds an edge which has least weight and add it to the growing spanning tree.

An MST has N-1 edges

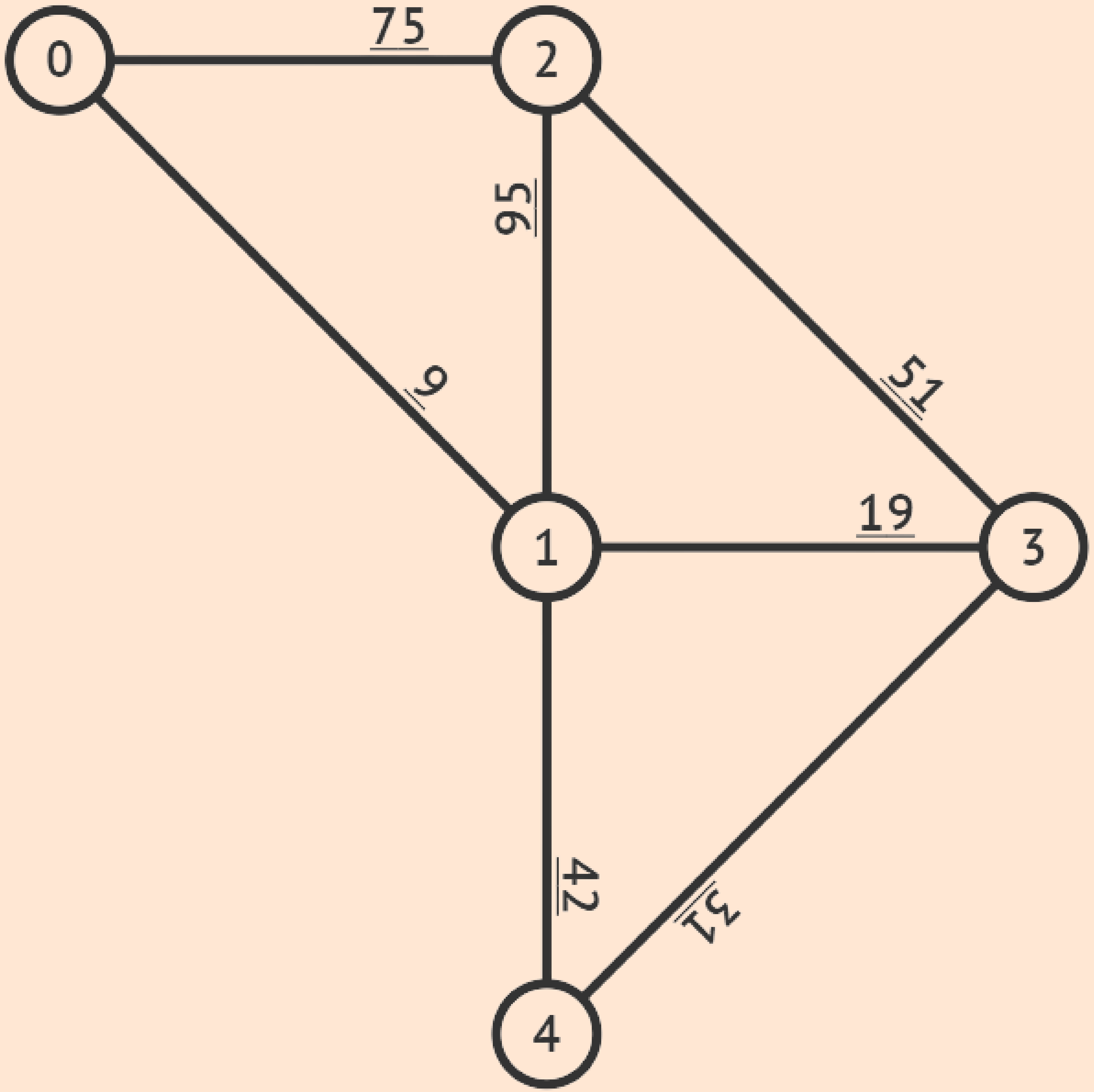
Tree by definition is acyclic.
An MST can not have cycles

Steps

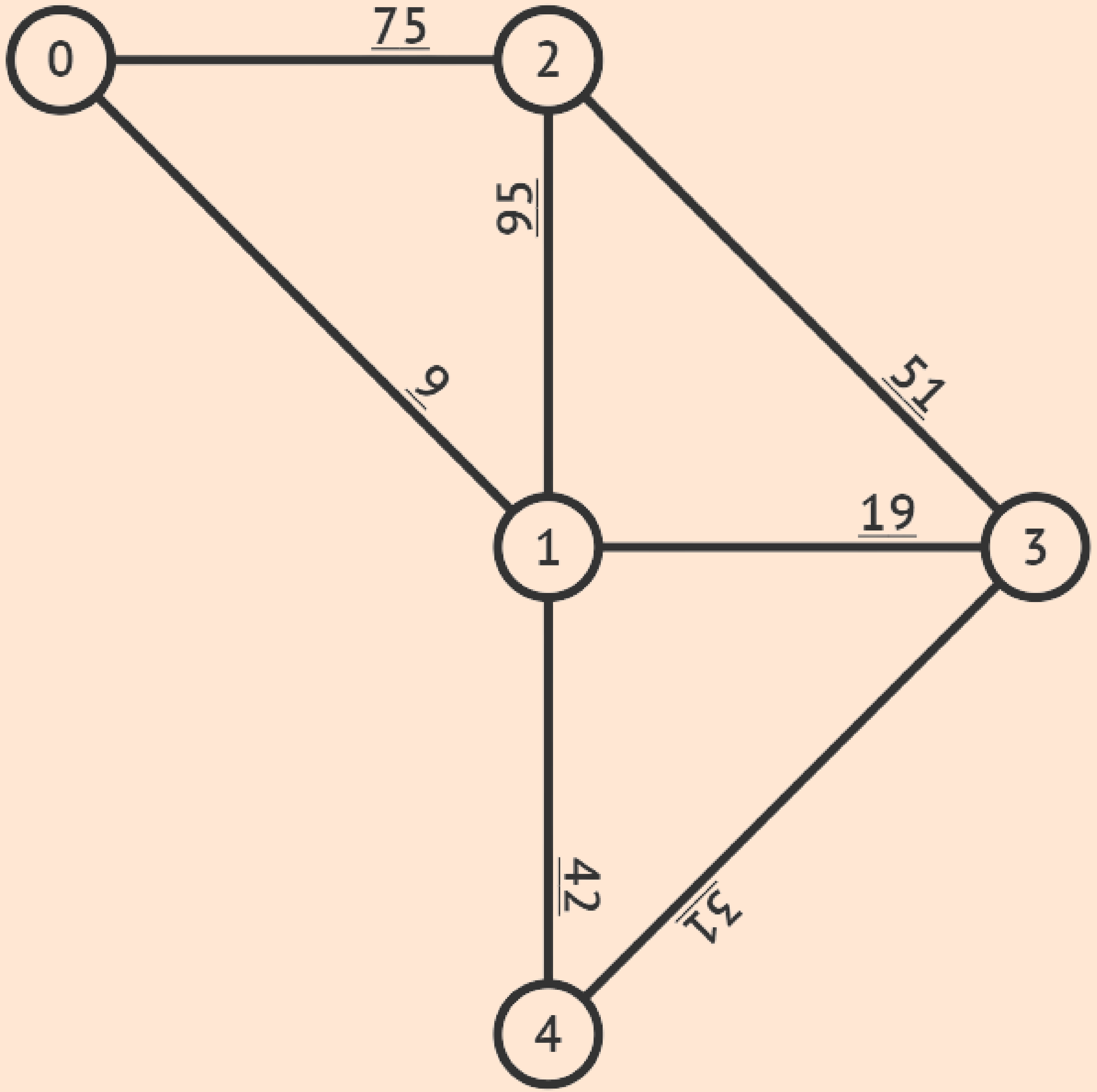
1. Sort all the edges in non-decreasing order of their weight.

2. Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.

3. Repeat step#2 until there are (V-1) edges in the spanning tree.


0
1
2
3
4
75
95
9
51
19
42
31

9 | 19 | 31 | 42 | 51 | 75 | 95



Parent array

0	1	2	3	4
0	1	2	3	4

Count = 0

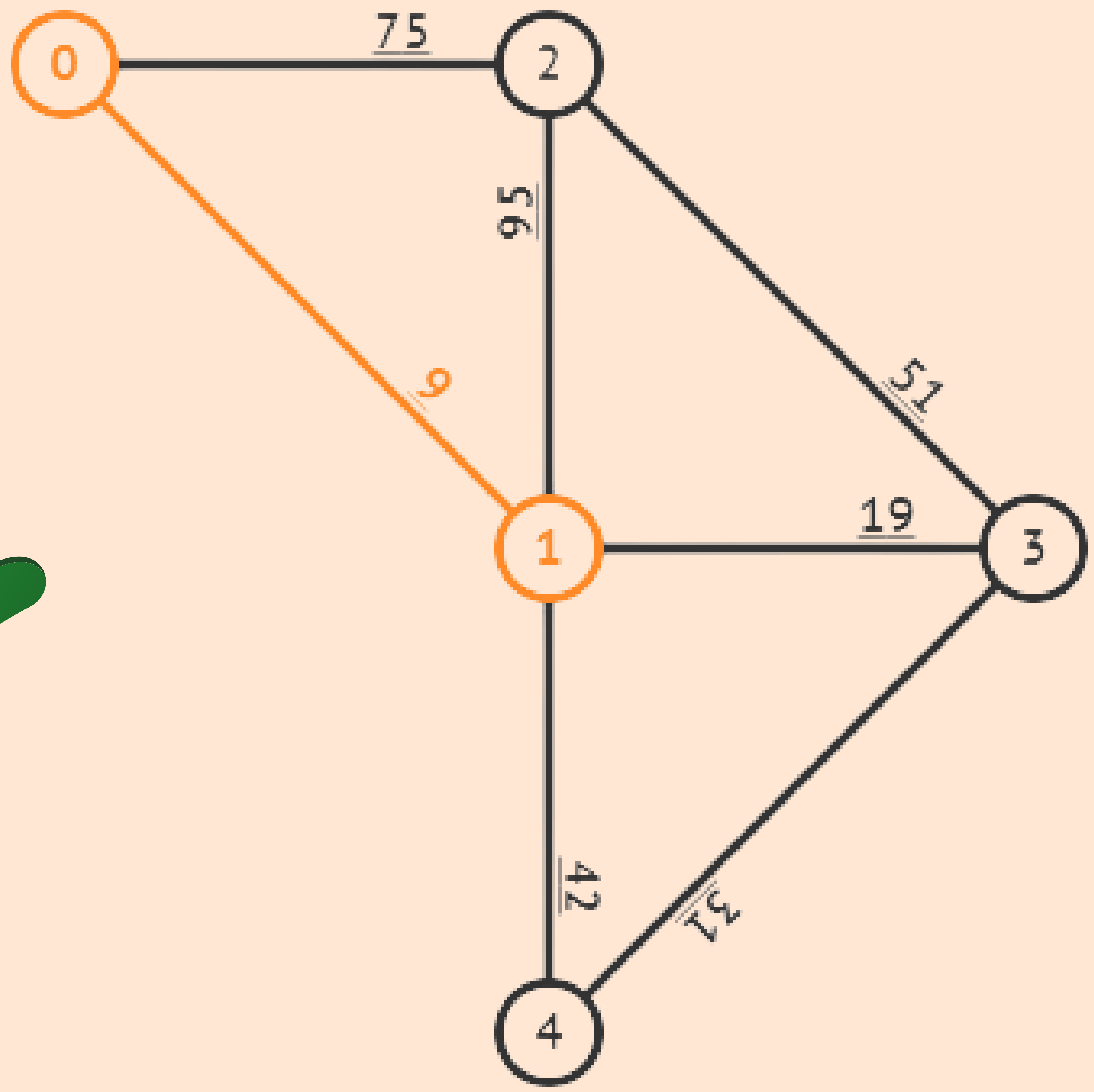
9 | 19 | 31 | 42 | 51 | 75 | 95

9

Parent array

0	1	2	3	4
0	1	2	3	4

0	0	2	3	4
0	1	2	3	4



Count = 1

9 | 19 | 31 | 42 | 51 | 75 | 95

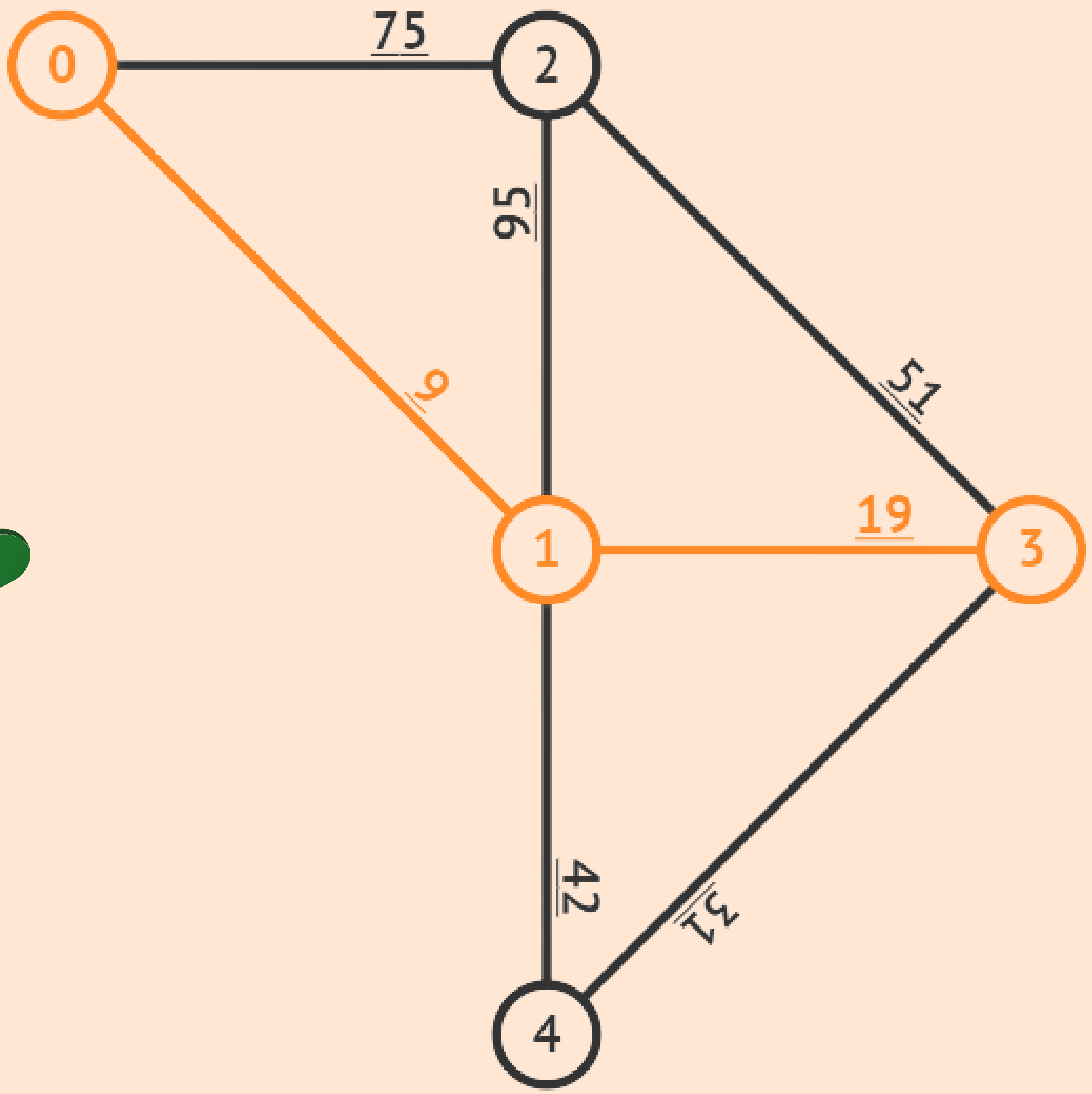
19

Parent array

0	0	2	3	4
0	1	2	3	4

0	0	2	0	4
0	1	2	3	4

Count = 2



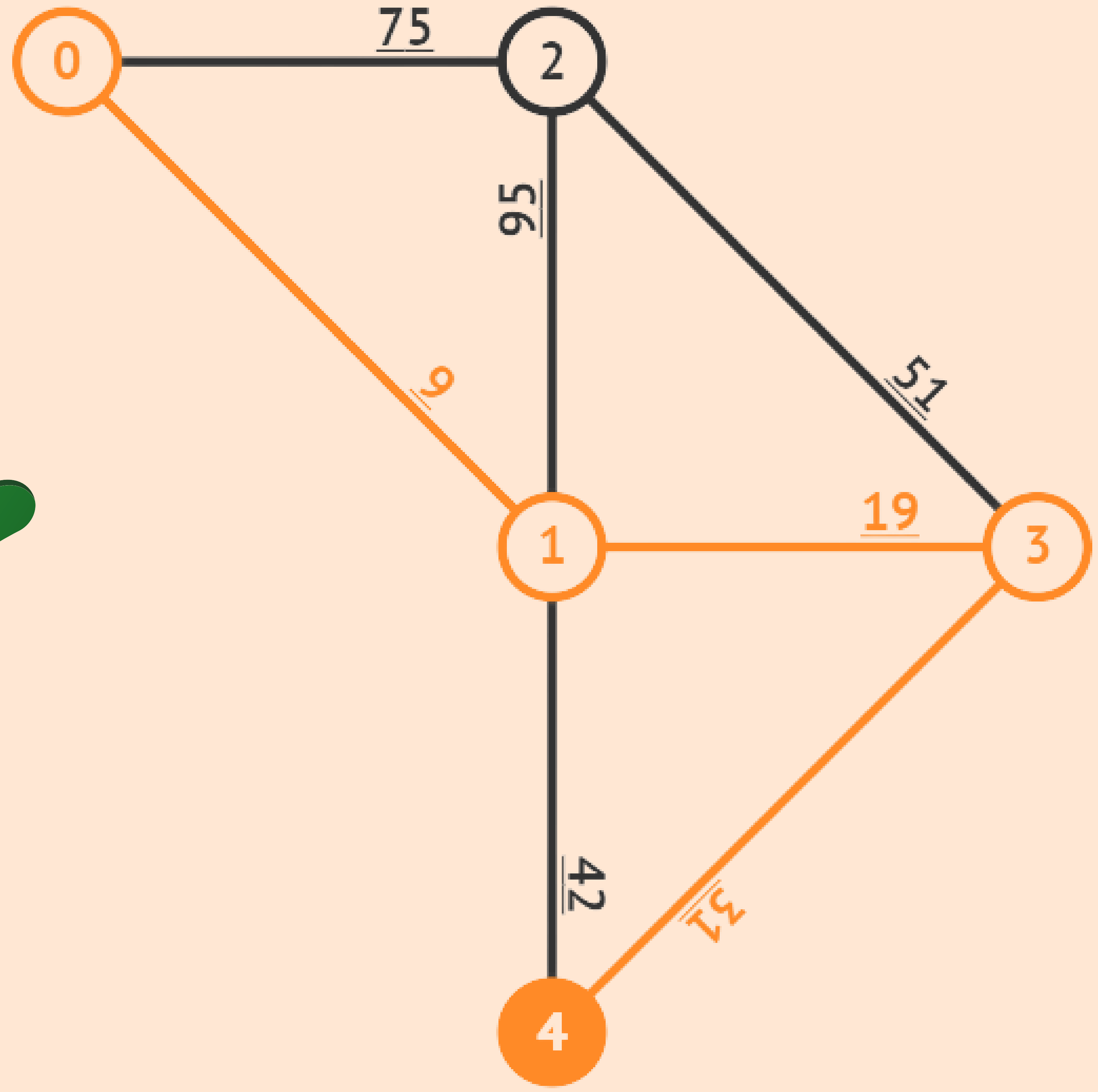
9 | 19 | 31 | 42 | 51 | 75 | 95

31

Parent array

0	0	2	0	4
0	1	2	3	4

0	0	2	0	0
0	1	2	3	4



Count = 3

9 | 19 | 31 | 42 | 51 | 75 | 95

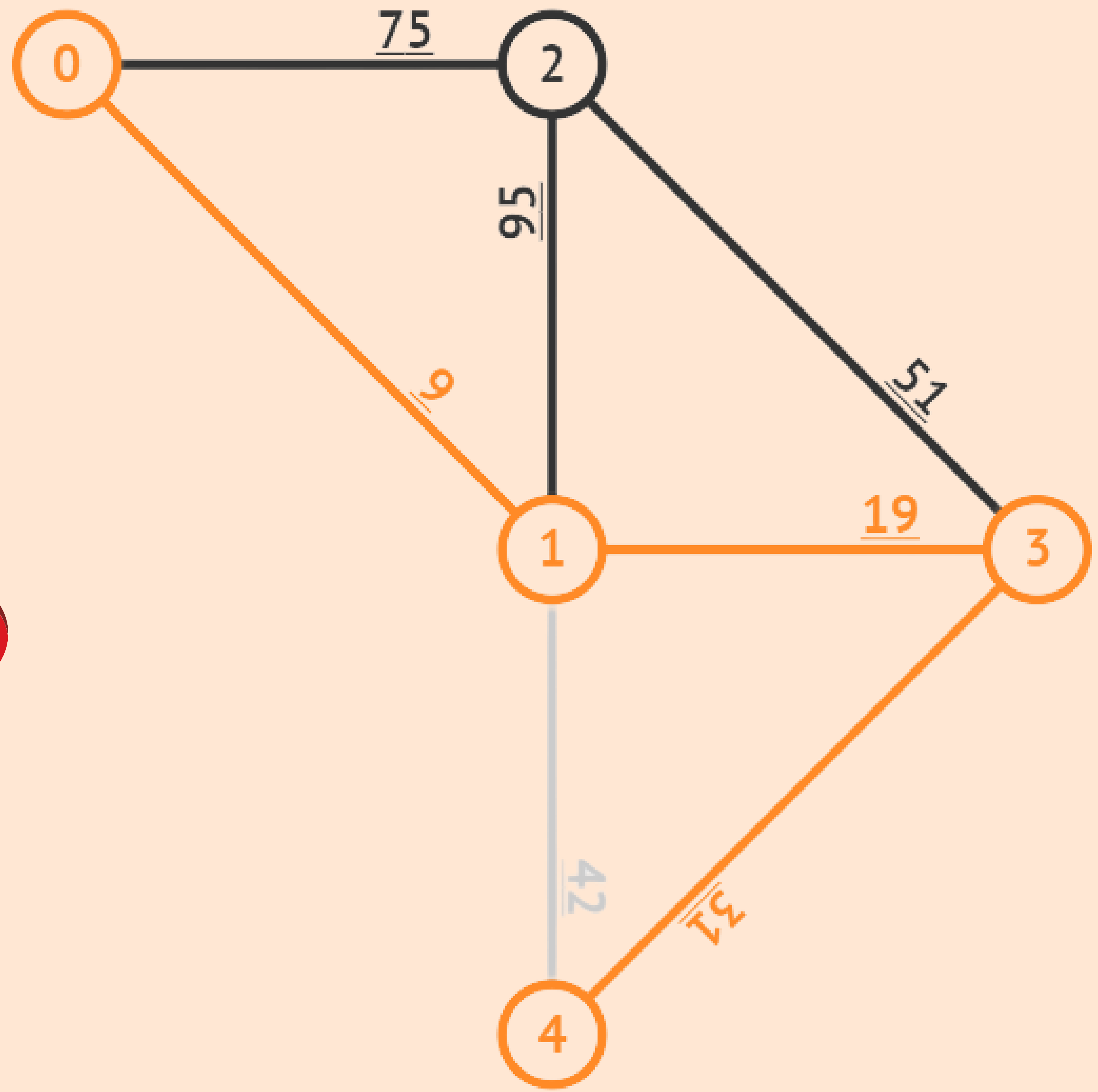
42

Parent array

0	1	2	3	4
0	0	2	0	0

0	1	2	3	4
0	0	2	0	0

Count = 3



9 | 19 | 31 | 42 | 51 | 75 | 95

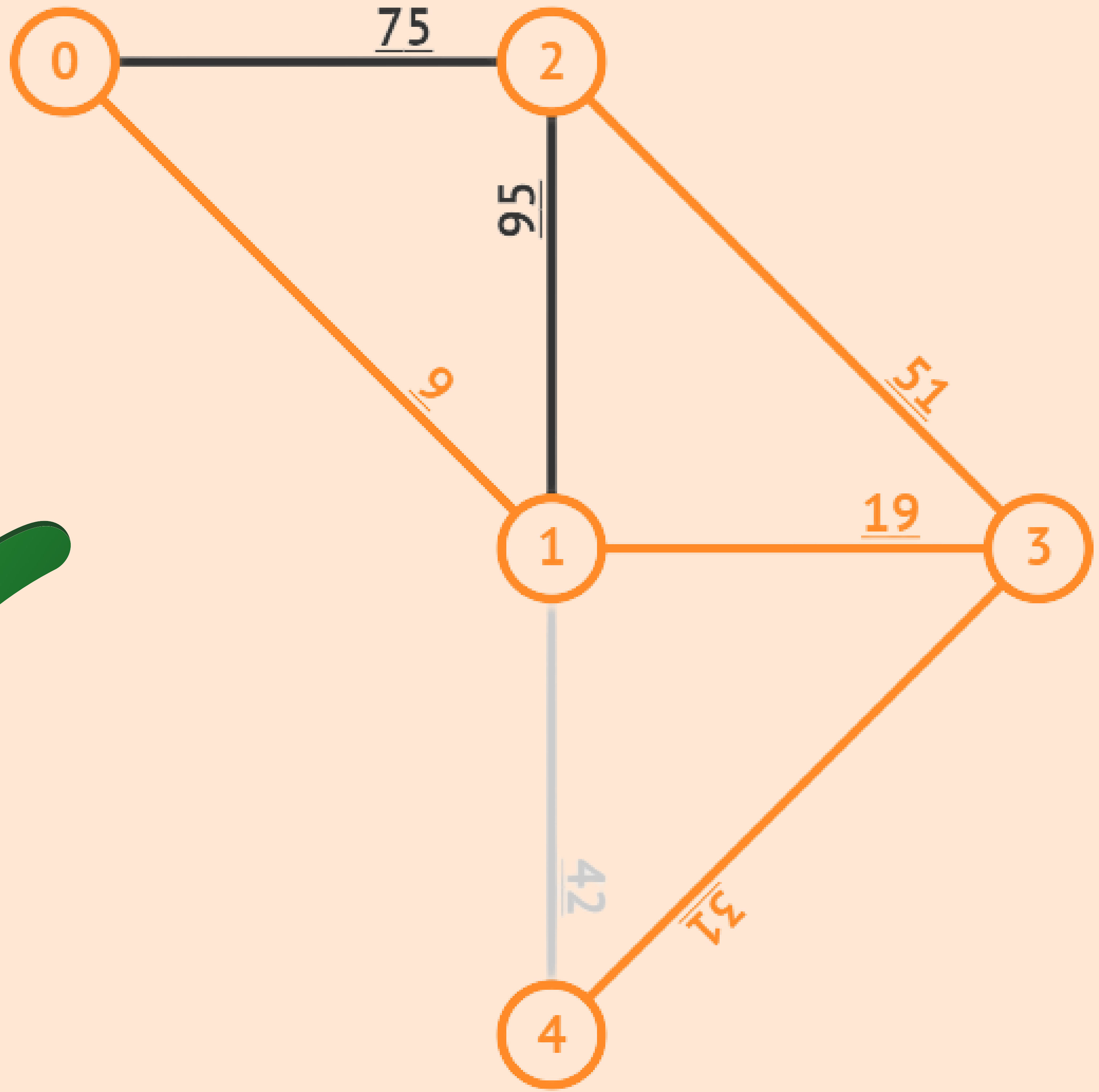
51

Parent array

0	0	2	0	0
0	1	2	3	4

0	0	0	0	0
0	1	2	3	4

Count = 4



9 | 19 | 31 | 42 | 51 | 75 | 95

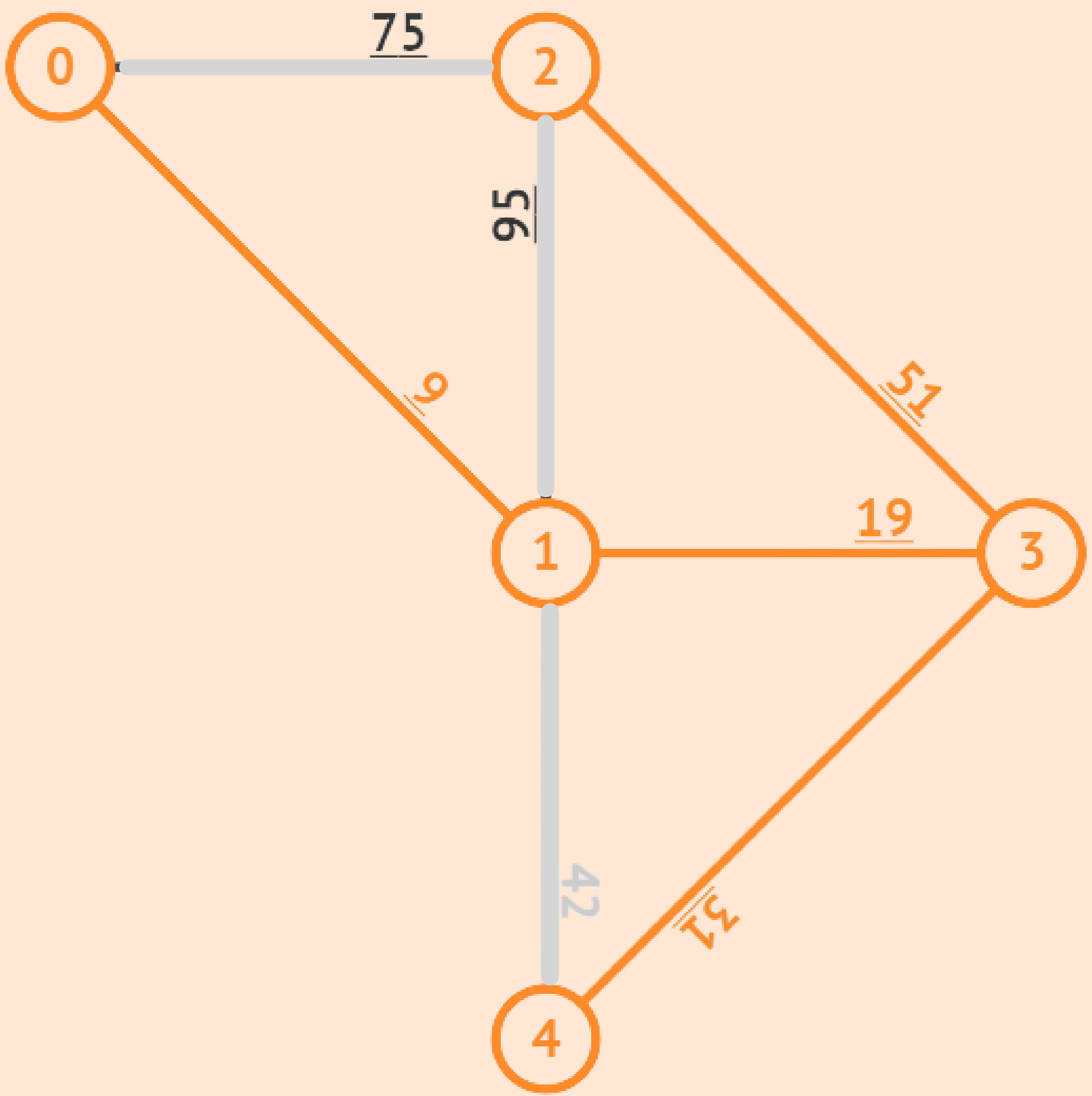
9+19+31+51= 110

Parent array

0	0	2	0	0
0	1	2	3	4

0	0	0	0	0
0	1	2	3	4

Count = 4 = V-1



MST

Pseudo Code

```
for each vertex: make set            // intialise parent array
sort each edge in non decreasing order by weight
for each edge (u,v) do:
    if findSet(u)!=findSet(v)
        Union(u,v)
        cost=cost+edge(u,v)
```

Time & Space Complexity

for each vertex: make set — O(V)

sort each edge in non decreasing order by weight — O(ElogE)

```
for each edge(u,v) do:              O(E)  ┐
    if findSet(u)!=findSet(v)       O(1)  │
        Union(u,v)                  O(V)  ├ ........ O(E*V)
        cost=cost+edge(u,v)         O(1)  ┘
```

Time Complexity: O(V+ElogE+E.V)

O(ElogE+E.V)

Space Complexity : O(V+E)

: O(E)

Optimise Time Complexity

- The union find method can take upto O(V) time to find and change the parent array.
- **Union by Rank and path compression** is an optimised algorithm which will at worst take O(logV) time for union find operations.
- Hence the time complexity will then be: O(ElogE+ElogV)

  = O(ElogE)

Q: Min cost to connect all points

https://leetcode.com/problems/min-cost-to-connect-all-points