Minimum Spanning Trees:

$G = (N, A)$ connected, undirected graph.

A subgraph $G' = (N, T)$ of G such that T has exactly $(n-1)$ edges with all the nodes remain connected is called a spanning tree.

Remark: Spanning tree is not unique!

Suppose each edge has given a nonnegative length or cost or weight. Then a spanning tree such that sum of the edges in T is as small as possible is called a minimum spanning tree.

Method 1: Start with an empty set T, select at every stage the shortest edge that has not been chosen or rejected regardless where this edge is situated in G.

Method 2: Choose a node and build a tree from there, selecting at every stage the shortest available edge that can extend the tree to an additional node. ~~tree.~~

- A set of edges is a soln. if it constitutes a sp. tree for the ~~so~~ n nodes in N.
- feasible if it does not include a cycle.
- a feasible set of edges is promising if it can be extended to produce not merely a soln., but an optimal soln.
- an edge leaves a given set if exactly one end of this edge is in the set. Fails if no end or both end.

Disjoint set structures:

N objects numbered 1 to N. To group these into disjoint sets, so that each object is exactly in one set.

In each set, we choose one member to serve as label for that set. choose minimum element {2,5,7} – set 2.

label: smallest member of each set

array: set (1 .. N).

place the label of each individual element in its position

1	2	3	4	5	.	.	.	.	.	N

place the label of the set corresponding to each object in the appropriate array element.

eg. {1,3,4}, {5,6}, {2,7} {8}

1	2	1	1	5	5	2	8

function find1(x): {finds the label of the set containing x}

return set(x). O(1)

merge1(a, b): {merges the sets labelled a & b, a ≠ b}

```
i ← min(a, b)
j ← max(a, b)
for k ← 1 to N
    if set(k) = j then set(k) ← i
```

O(n)

eg. {1,3,4}, {5,6}, {2,7}

1	2	1	1	5	5	2

merge(1,5)

i ← 1

j ← 5

1	2	1	1	1	1	2

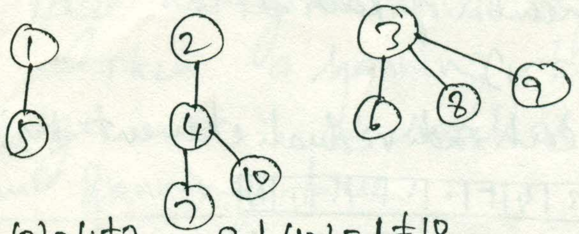
find(x) {1,3,4,5,6}, {2,7}.

Aliter: Represent each set as a rooted tree, where each node contains a single pointer to its parent.

If $set(i) = i$, then i is both label of its set and the root of the corresponding tree.

If $set(i) = j \neq i$, then j is the parent of i in some tree.

1	2	3	2	1	3	4	3	3	/	4



$set(1) = 1$
$set(2) = 2$
$set(3) = 3$
$set(4) = 2 \neq 4$, $set(7) = 4 \neq 7$, $set(10) = 4 \neq 10$
$set(5) = 1 \neq 5$, $set(8) = 3 \neq 8$
$set(6) = 3 \neq 6$, $set(9) = 3 \neq 9$

find2(x):

```
  r ← x
  while set(r) ≠ r
        r ← set(r)

  return r
```

merge2(a, b):

```
  If a < b then set(b) ← a
           else set(a) ← b
```

Kruskal(G = (N, A) : set of edges):
 {initialization}
 sort A by ↑ length
 n ← no. of nodes in N.
 T ← ∅ (edges of minimum sp. tree).
 Initialize n sets, each containing a different element of N.
 {greedy}
 repeat
  e ← (u, v) shortest edge not yet been considered
  ucomp ← find(u)
  vcomp ← find(v)
  if ucomp ≠ vcomp then
   merge(ucomp, vcomp)
   T ← T ∪ {e}
 until T contains (n-1) edges
 return T

<u>Demo:</u> A = {(1,2), (2,3), (4,5), (6,7), (1,4), (2,5), (4,7), (3,5), (2,4), (3,6), (5,7), (5,6)}

n ← 7, T ← ∅

Set initialization

1	2	3	4	5	6	7

e ← (1,2), (2,3), (4,5), (6,7), (1,4), (2,5)*, (4,7)
ucomp ← 1, 1, 4, 6, 1, 1, 1
vcomp ← 2, 3, 5, 7, 4, 1, 6
if ucomp ≠ vcomp then
 merge(ucomp, vcomp)

1 ≠ 2

1	1	3	4	5	6	7

1 ≠ 3

1	1	1	4	5	6	7

4 ≠ 5

1	1	1	4	4	6	7

6 ≠ 7

1	1	1	4	4	6	6

1 ≠ 4

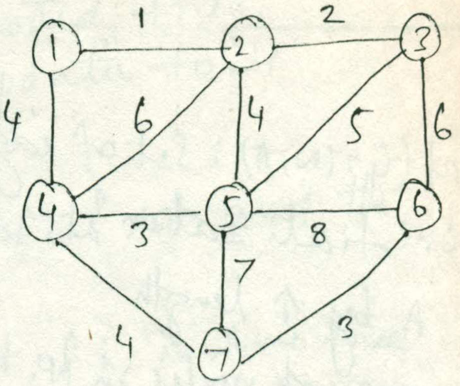
1	1	1	1	4	6	6

1 ≠ 6

1	1	1	1	4	1	6

T ← ∅ ∪ {(1,2), (2,3), (4,5), (6,7), (1,4), (4,7)}
 contains n-1 = 6
 return T.

Kruskal's Algorithm:



1. List all the edges in ↑ order.
2. Select a smallest edge.
3. For each stage select from the remaining edges of G, another smallest edge that make no cycle with the previously selected edges.
4. continue until (n-1) edges have been selected.

~~Step~~1. (1,2), (2,3), (4,5), (6,7), (1,4), (2,5), (4,7), (3,5), (2,4), (3,6), (5,7), (5,6).

Step.	edge considered	connected components
initial	–	{1}, {2}, {3}, {4}, {5}, {6}
1	(1,2)	{1,2}, {3}, {4}, {5}, {6}, {7}
2	(2,3)	{1,2,3}, {4}, {5}, {6}, {7}
3	(4,5)	{1,2,3}, {4,5}, {6}, {7}
4	(6,7)	{1,2,3}, {4,5}, {6,7}
5	(1,4)	{1,2,3,4,5}, {6,7}
6	(2,5)	rejected
7	(4,7)	{1,2,3,4,5,6,7} shown by ...