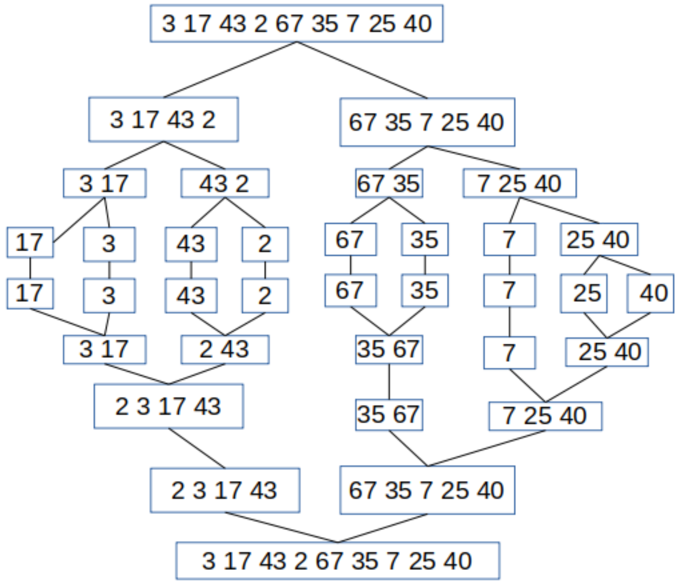
Matt Fletcher CS317-02 Homework 1

1.



2.

Three steps:

a) Move R to left, and L to R until $R < P$ and $L > P$.

b) If during (a) R and L did not cross, swap R and L, and continue (a).

c). If during (a) R and L did cross, swap R and P, freeze P in its new place, split the list in 2 around P, and recursively continue (a) on each new list.

An asterisk next to an element indicates that it is frozen.

P	L							R
43	21	90	8	44	35	6	2	13
P		L						R
43	21	90	8	44	35	6	2	13
P		R						L
43	21	13	8	44	35	6	2	90
P		L						R
43	21	13	8	44	35	6	2	90
P				L			R	
43	21	13	8	44	35	6	2	90
P				L			R	
43	21	13	8	2	35	6	44	90
P						R	L	
43	21	13	8	2	35	6	44	90
R						P	L	
6	21	13	8	2	35	43	44	90
P	L				R		PL	R
6	21	13	8	2	35	43∗	44	90
P	L			R			PR	L
6	21	13	8	2	35	43∗	44	90

```
P      R      L
6      2      13     8      21     35     43*    44*    90

              P      L             R
2      6*     13     8      21     35     43*    44*    90

              P      R      L
2      6*     13     8      21     35     43*    44*    90

                            PL     R
2      6*     8      13*    21     35     43*    44*    90

                            PR     L
2      6*     8      13*    21     35     43*    44*    90


2      6*     8      13*    21*    35     43*    44*    90
```

All elements are either frozen or single element lists, therefore, the list is sorted.

3.

Five applications of graph theory: social media connections, navigation, cellular connections to towers, circuit design, register allocation

---

4. A directed graph has edges which indicate the direction they travel. The direction of travel cannot oppose the node. On the other hand, an undirected graph allows travel both ways along any edge.

5.

a) Edges: ac, cf, ab, be, bd

Nodes:a,b,c,d,e,f

b) Edges: fc, ca, ba, be, bd

Nodes: a,c,f,b,e,d

6.

a) Adjacency List

```
[ a ]-->[ b ]-->[ c ]-->[ d ]-->[ e ]-->[ f ]-->[ g ]
  |       |       |       |       |       |       |
  v       v       v       v       v       v       v
[->b]   [->d]   [->f]   [->b]   [->b]   [->c]   [->e]
  |       |       |       |       |       |       |
  v       v       v       v       v       v       v
[->c]   [->e]   [->a]   [->0]   [->g]   [->g]   [->f]
  |       |       |               |       |       |
  v       v       v               v       v       v
[->0]   [->a]   [->0]           [->0]   [->0]   [->0]
          |
          v
        [->0]
```

a) Adjacency Matrix

	A	B	C	D	E	F	G
A	0	1	1	0	0	0	0
B	1	0	0	1	1	0	0
C	1	0	0	0	0	1	0
D	0	1	0	0	0	0	0
E	0	1	0	0	0	0	1
F	0	0	1	0	0	0	1
G	0	0	0	0	1	1	0

b) Adjacency List

```
[ a ]-->[ b ]-->[ c ]-->[ d ]-->[ e ]-->[ f ]-->[ g ]
  |       |       |       |       |       |       |
  v       v       v       v       v       v       v
[->b]   [->d]   [->a]   [->0]   [->0]   [->c]   [->0]
  |       |       |                       |
  v       v       v                       v
[->g]   [->e]   [->0]                   [->g]
  |       |                               |
  v       v                               v
[->0]   [->0]                           [->0]
```

b) Adjacency Matrix

	A	B	C	D	E	F	G
A	0	1	0	0	0	0	1
B	0	0	0	1	1	0	0
C	1	0	0	0	0	0	0
D	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0
F	0	0	1	0	0	0	1
G	0	0	0	0	0	0	0

---

7. Traversal order:
A 1/16
B 2/9
C 10/15
D 3/4
E 5/8
F 11/14
G 12/13
Stack order:
Top →→→ →Bottom
$\boxed{GFCHEDBA}$

---

8. Traversal order:
A 1
B 2
C 2
D 3
E 3
F 4
G 5
Queue:
Front →→→ →Back
$\boxed{ABCDEFG}$

9.
Efficiency of Matrix: $\Theta(\|V^2\|)$
Efficiency of List: $\Theta(\|V\| + \|E\|)$

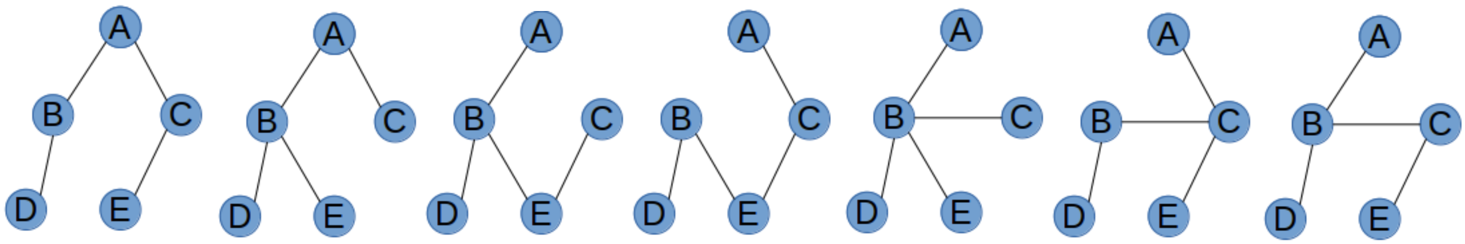
---

10. The greedy technique builds a solution piece by piece by picking the most obvious benefit first, without ever reconsidering a decision. It generates a globally optimal solution from a locally optimized choice.

TODO Greedy algorithm and the greedy choice property.

---

11. A spanning tree is a set of edges that connects every node in the graph without forming loops or cycles.



TODO FIND ALL SPANNING TREES.

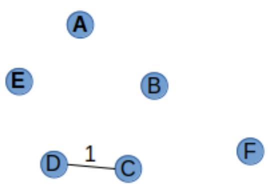
---

12. A minimum spanning tree (MST) is the smallest tree that connects every node in the graph. If the graph is weighted, the smallest tree has the smallest sum of weights. If the graph is unweighted, the MST has the smallest number of edges.
An example of a minimum spanning tree is determining the minimum amount of wire needed to connect multiple nodes.

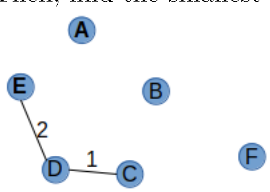
13.
Kruskal's Algorithm:
→1. Create a table with each edge sorted in ascending order.

Edge	CD	DE	AD	AE	BC	AB	BE	BD	AC	BC
Weight	1	2	3	4	5	6	6	8	9	11

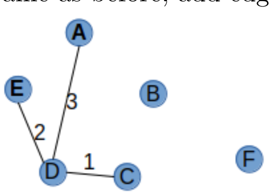
→2. Starting from either node on the smallest line, attach the next smallest line that does not create a loop.
Start by attaching line CD, weight 1.



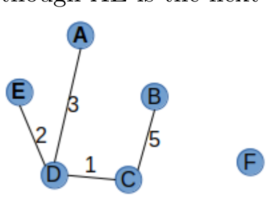
Then, find the smallest weighted edge from any end node that does not create a cycle. This is edge DE.



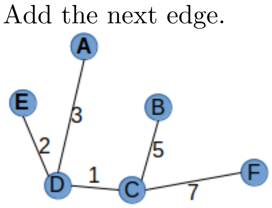
Same as before, add edge AD.



Although AE is the next smallest edge, adding it would create a cycle. So we add BC instead.



Add the next edge.



At this point, all the nodes are connected and adding any more lines would result in a cycle being created. Hence, we are done.

14.

15.

16.
