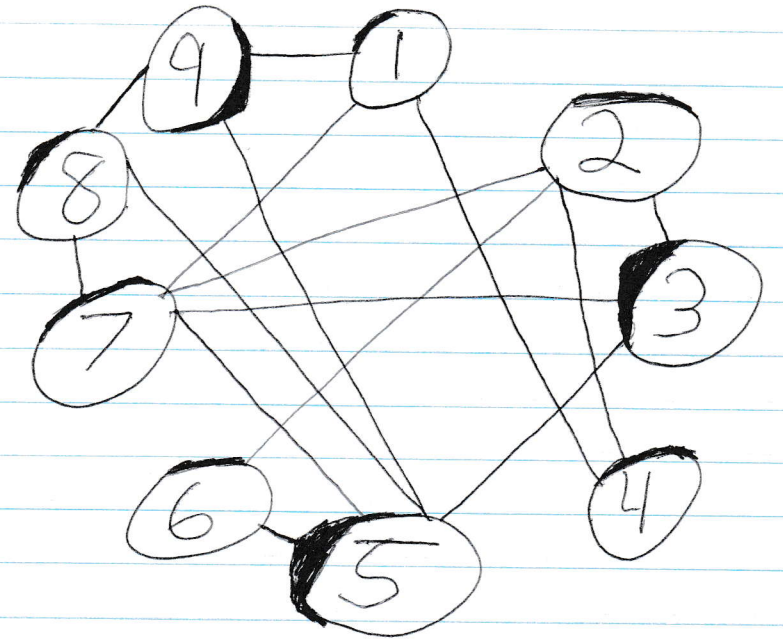
Barak Barclay

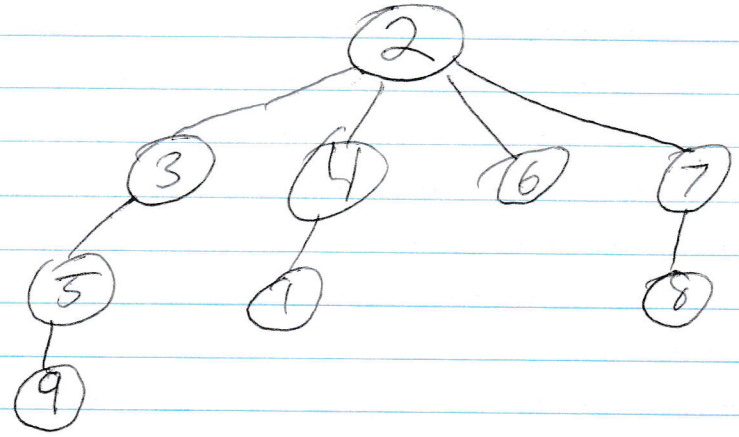
5.) 1. 20, 10, 16, 40, 30, 27, 80, 50
2. 10, 16, 20, 27, 30, 40, 50, 80
3. 16, 10, 27, 30, 50, 80, 40, 20

Barak Barclay

7.)



Parent	2	2	2	2	3	4	7	5
Child	3	4	6	7	5	1	8	9



Barak Barclay

9.) 1. Input: A vertex in a graph
2. Ensure the vertex exists. If the does not exist, got to step 7, else go to step 3.
3. Check if the vertex has an edge. If the vertex has an edge, go to step 4, else go to step 5.
4. Add the vertex from the edge to a neighbors vector, and return to step 3 (excluding already checked edges!)
5. Return the neighbors vectors to the user. Go to step 6.
6. Stop the algorithm.
7. Inform the user that the vertex does not exist, and Exit the algorithm with an error.

Barak Barclay

10.) 1. Input: A vertex.
2. Ensure the vertex exists in the graph. If the vertex does not exist, go to step , else go to step 3.
3. ~~Traversed the vertexs and edges in the graph.~~ If Search the edges of the graph. If there is an edge from the current vertex to the inputted vertex, or vice versa, go to step 4. Once the search is complete go to step 5.
4. Remove the edge. Go back to step 3.
5. Remove the vertex. Go to step 6.
6. Stop the algorithm.
7. Inform the user that the vertex does not exist, and exit the algorithm with an error.

Barak Barclay

11.) 1. Traverse the array, starting with the begining index. If there is a vertex, go to step 2, else go to step 3.

2. If there is an edge from the vertex at that index to another vertex, set the weight of the edge to the respective index in the adjacency matrix. Go back to step 1.

3. Return the adjacency matrix to the user. Go to step 4.

4. Stop the algoritm.