COSC 455
Midterm #1 Review Sheet
Spring 2016

1.) For each of the following, indicate whether $f = O(g)$, $f = \Omega(g)$, or $f = \Theta(g)$:

	$f(n)$	$g(n)$
a.	$100n + \log n$	$n + (\log n)^2$
b.	$n^2 / \log n$	$n(\log n)^2$
c.	$n^{1/2}$	$5^{\log_2 n}$
d.	$n!$	2^n
e.	$(\log n)^{\log n}$	$2^{(\log_2 n)^2}$

2.) Solve the following recurrence relations and give a O bound for each:

a.) $T(n) = 7T(n/7) + n$
b.) $T(n) = 9T(n/3) + n^2$
c.) $T(n) = 8T(n/2) + n^3$

3.) Write the recurrence and solve the recurrence for the following function:

```
function f(n):
  if n>1:
     print("still going")
     f(n/2)
     f(n/2)
```

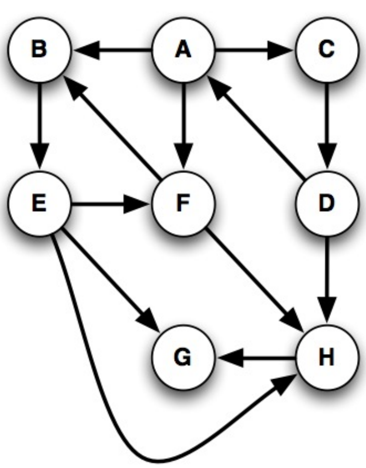
4.) Consider the standard algorithm for mergesort. Give the recurrence for mergesort and solve that recurrence.

5.) Suppose that you have k sorted arrays, each with n elements, and you want to combine them into a single sorted array of kn elements. Create an algorithm that is more efficient than $O(k^2 n)$.

6.) The Dutch Flag Problem is to arrange an array of the characters $\{R, W, B\}$ such that all of the R's come first, then the W's, then the B's. Give a linear-time, in-place algorithm to do this arrangement.

7.) Show that binary search is $O(\log n)$.

8.) Consider the following graph:



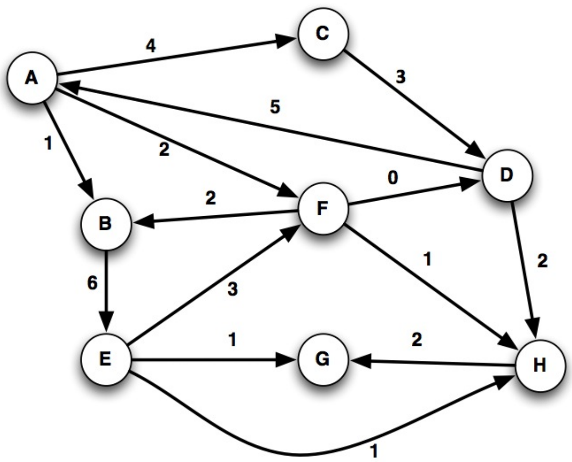
Construct the DFS tree. Add in and clearly identify forward edges, back edges, and cross edges. Assume that if there is a choice, we will expand out in alphabetical order.

9.) Run the strongly connected components algorithm on the graph in #8. Clearly show all steps.

10.) A graph is said to be bipartite if all of its vertices can be partitioned into two disjoint subsets X and Y such that edges only connect a vertex from X to a vertex in Y. There are no edges within the subsets. Using DFS, can I detect if a graph is bipartite? Can I do it with BFS?

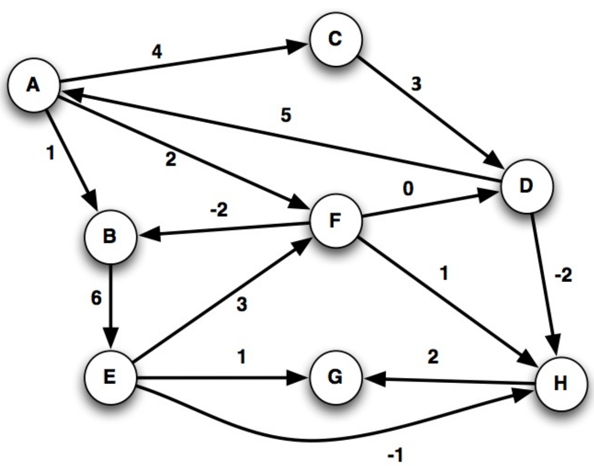
11.) Give a linear time algorithm which takes as input a directed graph, and determines whether or not there exists a vertex $s \in V$ from which all other vertices are reachable.

12.) Consider the following graph:



Run Dijkstra's algorithm on this graph, with A as the source. Show all steps.

13.) Consider the following graph:



Run the Bellman-Ford algorithm on this graph, with A as the source. Show all steps.

14.) How can we use the Bellman-Ford algorithm to detect if there exists a negative cost cycle in my graph?

15.) Consider the problem of determining all-pairs shortest path. For this problem, after your algorithm executes, each vertex s should have an array [1...n] that holds a list of the shortest path distances from s to each other vertex. Basically, determine the shortest path between all pairs of distinct vertices i and j. Give an algorithm that solves this problem. What is the time complexity of your algorithm?

16.) Fully describe and explain the DFS algorithm, including correctness.

17.) Fully describe and explain Dijkstra's algorithm, including correctness.