**Hyo Lee**
**Extra Credit Submission**
**02/23/2016**

# Homework (Extra Credit)

Many of these questions are taken from the review problems for the midterm.

## Algorithmic Complexity

1. a) For the following recursive function, find f(5): **= 15**

```
int f(int n) {

if (n == 0)

     return 0;

else

     return n + f(n - 1);

}
```

```
f(0) = 0
f(1) = 1
f(2) = 3
f(3) = 6
f(4) = 10
f(5) = 15
```

b) For the function in Question a), find f(0). **= 0**

c) For the function in Question a), suppose + is changed to * in the inductive case. Find f(5). **= 0**

d) For the function in Question a), what happens with the function call f (-1)? **= Negative would give an error, beyond base condition**

2. Compute the following sum

1+ 1/2 + 1/4 + 1/8 …… = **Sn = 2 - 1/2^(n-1) ≈ 2**

3. Rank the following time complexities starting from the least to the greatest: O($n^2$), O(log n), O(log log n), O(n), O (n log n) **= O(log log n), O(log n), O(n), O(n log n), O(n^2)**

4. Algorithm: What problem does this algorithm solve? Find the time complexity of the algorithm. **= The algorithm is a selection sort that sorts in ascending order at O(n^2)**

```
for i= 1 to n do
// find min element in A[i...n]
// and put it in the i'th position (i.e. at A[i])

     min_index <-- i

     //locate min
     for j= i+1 to n do

          if A[j] < A[min_index] then min_index <-- j

     //put the min where it belongs
     swap( A[i], A[min_index] )
```

5. Consider the following three algorithms for determining whether anyone in the room has the same birthday as you.

*Algorithm 1*: You say your birthday, and ask whether anyone in the room has the same birthday. If anyone does have the same birthday, they answer yes.

*Algorithm 2*: You tell the first person your birthday, and ask if they have the same birthday; if they say no, you tell the second person your birthday and ask whether they have the same birthday; etc, for each person in the room.

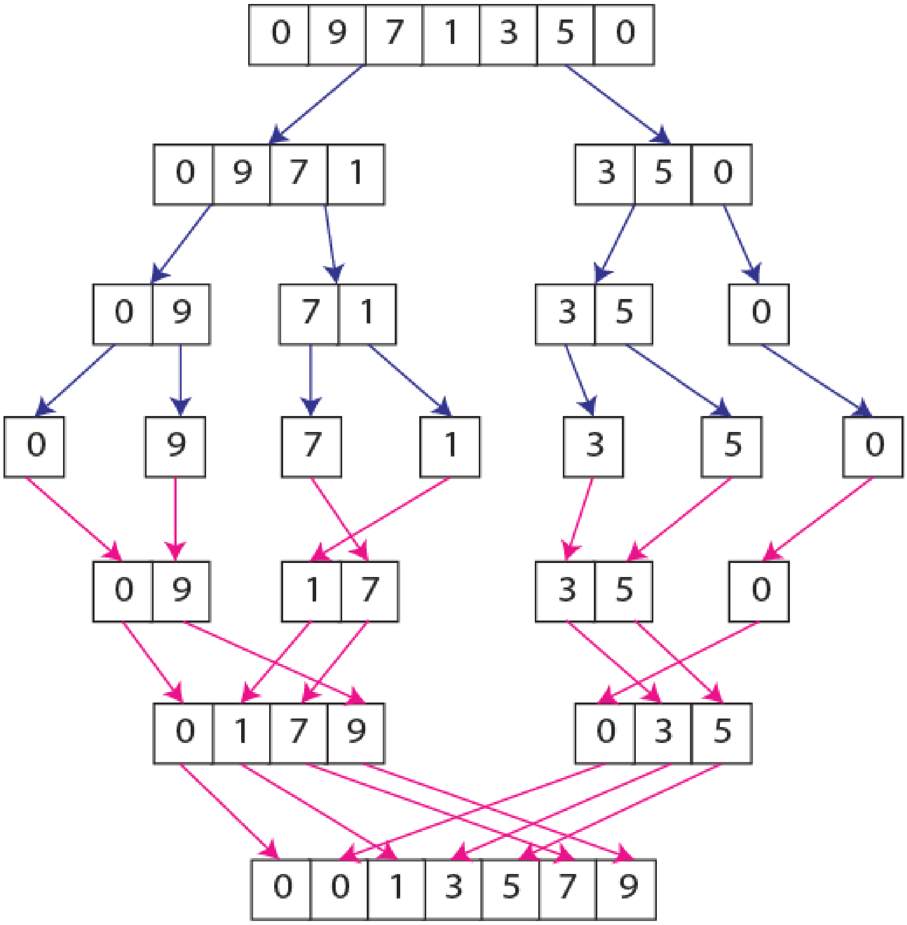
*Algorithm 3*: You only ask questions of person 1, who only asks questions of person 2, who only asks questions of person 3, etc. You tell person 1 your birthday, and ask if they have the same birthday; if they say no, you ask them to find out about person 2. Person 1 asks person 2 and tells you the answer. If it is no, you ask person 1 to find out about person 3. Person 1 asks person 2 to find out about person 3, etc.

Question 1: For each algorithm, what is the factor that can affect the number of questions asked (the "problem size")? **= The number of people at the party who need to be asked.**

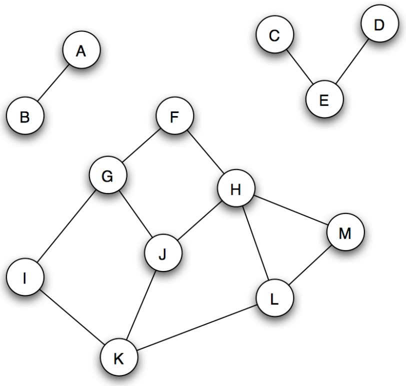
Question 2: In the worst case, how many questions will be asked for each of the three algorithms? **= 1) 1 time, 2) n times, 3) n(n + 1) / 2 times**

Question 3: For each algorithm, say whether it is constant, linear, or quadratic in the problem size in the worst case. **1) constant, 2) linear, 3) quadratic**

6. Sort the following numbers [0, 9 7, 1, 3, 5 0] using merge sort. Draw a diagram to clearly illustrate how merge sort works.



7. Consider the following graph. Represent it using an adjacency matrix.



$$\begin{array}{c|ccccccccccccc}
 & a & b & c & d & e & f & g & h & i & j & k & l & m \\
\hline
a & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
b & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
c & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
d & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
e & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
f & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
g & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
h & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
i & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
j & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
m & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0
\end{array}$$