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):

```
int f(int n) {

if (n == 0)

     return 0;

else

     return n + f(n - 1);

}
```

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

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

d) For the function in Question a), what happens with the function call f (-1)?

2. Compute the following sum

1+ 1/2 + 1/4 + 1/8 ……

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)

4. Algorithm: What problem does this algorithm solve? Find the time complexity of the algorithm.

```
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")?

Question 2: In the worst case, how many questions will be asked for each of the three algorithms?

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

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.