Exam 1
Computer Science 751
Lehman College– CUNY
Thursday, 17 October 2002

NAME (Printed) ______________________________

NAME (Signed) ______________________________

Login ______________________________

Please show all your work and circle your answers. Your grade will be based on the the work shown.

Question 1		10 points
Question 2		15 points
Question 3		15 points
Question 4		10 points
Question 5		15 points
Question 6		15 points
Question 7		20 points
TOTAL		100 points

Useful Formulas

$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ $\quad$ $\sum_{i=1}^{n} x^i = \frac{x^{n+1}-1}{x-1}$

$\sum_{i=1}^{n} \frac{1}{i} = \ln n + O(1)$ $\quad$ $\sum_{i=0}^{\infty} i x^i = \frac{x}{(1-x)^2}$

$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$ $\quad$ $\lim_{n\to\infty}(1 + \frac{x}{n})^n = e^x$

$n! = \sqrt{2\pi n}(\frac{n}{e})^n(1 + \Theta(\frac{1}{n}))$ $\quad$ $n! = o(n^n)$

$n! = \omega(2^n)$ $\quad$ $\lg(n!) = \Theta(n \lg n)$

$\binom{n}{k} = \frac{n!}{k!(n-k)!}$ $\quad$ $(\frac{n}{k})^k \le \binom{n}{k} \le (\frac{en}{k})^k$

$E[X] = \sum_x x Pr[X = x]$ $\quad$ $Var[X] = E[(X - E[X])^2]$

$= E[X^2] - E^2[X]$

1. True or False (2 point each):

   (a) ___ $\lg n = o(n^2)$.

   (b) ___ $\lg n = O(n^2)$.

   (c) ___ $3n^2 + 2n = \omega(n)$.

   (d) ___ $3n^2 + 2n = \Omega(n)$.

   (e) ___ $n! = \Omega(2^n)$.

   (f) ___ $\lg(n!) = \Theta(\lg(n^n))$.

   (g) ___ $f(n) = o(g(n))$ implies $f(n) = O(g(n))$.

   (h) ___ $f(n) = \Omega(g(n))$ implies $f(n) = \Theta(g(n))$.

   (i) ___ $f(n) = \Theta(g(n))$ implies $f(n) = O(g(n))$.

   (j) ___ $f(n) = \Theta(g(n))$ implies $f(n) = \omega(g(n))$.

2. Assume that every statement takes a constant c time. Give tight bounds on the order of growth and justify your answer:

   (a) What is the output, assuming the following piece of code is embedded in a complete and correct program:

   ```
   for ( int i = 5; i > 0; i--)
   {
      for ( int j = 0 ; j < i; j++)
          cout << '*';
      cout << endl;
   }
   ```

   (b) Assume A is an array of length n:

   ```
   FIND-MAX(A)
      max <- - infinity
      for i <- 1 to n
           do if A[i] > max
              then max <- A[i]
      return max
   ```

   (c) Assume A is an array and the function COMBINE takes $\Theta(n)$ on a sublists `A[p..r]` and `A[r+1..p]` of combined length n:

   ```
   MSORT(A,p,q)
      if ( q - p > 1)
           do MSORT(A,p, q/2);
                MSORT(A,q/2+1,q);
                COMBINE(A,p,q/2,r);
   ```

3. Assume A is an array storing a heap and k is a key:

```
HEAP-INSERT(A,k)
heap-size[A] <- heap-size[A] + 1
i <- heap-size[A]
while i > 1 and A[PARENT(i)] < key
     do A[i] <- A[PARENT(i)]
        i <- PARENT(i)
A[i] <- key
```

   (a) What does the heap look like inserting keys from the sequence: $\{10, 3, 1, 12, 20, 18, 14, 16\}$?

   (b) What is the height of the heap from inserting keys from the sequence: $\{10, 3, 1, 12, 20, 18, 14, 16\}$?

   ____

   (c) Write a function that will take a heap (stored in an array called A) and return the maximum value.

4. Give asymptotic upper and lower bounds for $T(n)$ for the following two recurrences. Make your bounds as tight as possible, and justify your answers: Assume that $T(n)$ is constant for $n \leq 2$:

   (a) $T(n) = 5T(n/3) + 1$

   (b) $T(n) = 10T(n-2) + n$

5. Assume $A[1..n]$ is an array.

```
FIND-KEY(A,k)
for i <- 1 to n
     do if A[i] = k
        then return i
```

   (a) What are tight bounds on the **worst case** order of growth? Justify your answer:

   (b) What are tight bounds on the **best case** order of growth? Justify your answer:

   (c) What are tight bounds on the **average case** order of growth, assuming that all numbers in A are randomly drawn from the interval $[1, n]$? Justify your answer:

6. Suppose that we have an array of n objects to sort and that the key of each record has the value $\{0, 1, \ldots, k\}$. Assume that k is much smaller than n ($k = o(n)$). Give a simple, **linear-time** algorithm for sorting the n-objects.

7. Suppose that we have an array of n objects to sort, and there are no conditions on the keys.

   (a) What is the lower bound on the worst case running time of a comparison sort of the array A?

   (b) Write a sorting algorithm that sorts a list with the worst case running time you stated above: