Name: _____

ID Number: _____

CSC 225 Midterm Exam
Tuesday Oct. 18, 2011

Instructions:

1. Put your name on every page of the exam.
2. No calculators or other aids. Closed book.
3. Read through the entire exam before beginning. You should have 8 pages including this header page.

Question	Value	Mark
1	20	
2	30	
3	20	
4	30	
Total	**100**	

Recall that you need at least 40% (40/100) in order to write the final exam in this course. Suggested strategy: read through the exam before starting, and begin with the questions which are easiest for you.

1. [20] Consider the following recurrence relation defined only for $n = 2^k$ for integers k such that $k \geq 1$: $T(2) = 7$, and for $n \geq 4$, $T(n) = n + T(n/2)$.

   Three students were working together in a study group and came up with this answer for this recurrence: $T(n) = n * \log_2(n) - n - \log_2(n) + 8$.

   Determine if this solution is correct by trying to prove it is correct by induction. If you cannot complete the induction proof, explain what goes wrong.

2. Consider the following sum: $S(n) = \sum_{i=1}^{n} (i^5 \; n^2)$.

(a) [5] Give a simple function $f(n)$ so that the sum $S(n)$ is in $\Theta(f(n))$.

(b) [5] State the definition of Big Oh.

(c) [10] Use your definition of Big Oh from (b) to prove that $S(n)$ is in $O(f(n))$ where $f(n)$ is your answer to part (a).

(d) [10] Prove that $S(n)$ is in $\Omega(f(n))$ where $f(n)$ is your answer to part (a).

3. [20] Suppose that a program uses an array A[0 ... NMAX] for storing *n* distinct integer data items. The data is stored in positions A[0 ... (n-1)]. The value of NMAX is selected to be large enough so that there is always enough space in the array for all of the *n* data items.

   Consider these approaches to ordering the data in A, where *n* is given:

   1. The values are unsorted.
   2. The values are sorted from the minimum value to the maximum value.
   3. The values are stored as per a max-heap as done in class for HeapSort.

   Consider these operations on the data:

   1. FIND_MAX: Find the array position of the maximum value.
   2. DELETE_MAX: You are given the array position of the maximum value. Delete this value and update the array to maintain the required properties of the data structure.
   3. INSERT: Insert a new value into the array (retaining the required properties of the data structure).
   4. FIND_MIN: Find the array position of the minimum value.

   Fill in this chart indicating the time complexity using Θ notation that each operation takes in the **worst case** on each of these data structures using an algorithm for the problem which is as fast as possible.

Data Structure	FIND_MAX	DELETE_MAX	INSERT	FIND_MIN
Unsorted Array				
Sorted Array				
Max-Heap				

Grading scheme: You start with 20 points and get -2 for each blank entry and -3 for each incorrect entry.

4. Consider the code covered in class for a recursive maxSort routine:

```
public void maxSort(int size) {int i, t, maxPos;
    if (size <= 1) return;
    maxPos=0;
    for (i=1; i < size; i++)
        if (A[i] >= A[maxPos]) maxPos=i;
    t= A[maxPos];
    A[maxPos]= A[size-1];
    A[size-1] = t;
    maxSort(size-1);
}
```

The goal in this question is to count the number of times that an access is made to an entry of A. A statement like: *t= **A**[maxPos];* counts as one array access whereas this one: ***A**[maxPos]= **A**[size-1];* counts as two. Define $T(n)$ to be the number of times that an entry of A is accessed for a problem of size n.

(a) [5] Is the operation of accessing an array entry a good choice for a proxy operation to be used for measuring the time complexity of this algorithm? Justify your answer.

(b) [5] Determine a recurrence relation for $T(n)$. Explain where each part of your recurrence is coming from in terms of the maxSort method.

[Question 4 continued]

(c) [10] Solve your recurrence relation from part (b) by repeated substitution to get a closed formula.

Your recurrence relation is:

[Question 4 continued]

(d) [10] Prove that the answer you get from part (c) is the correct formula for the number of times the maxSort program accesses an entry of the array A. You can get up to 5 marks for part (d) if your answer to (c) is wrong and you continue the induction proof until it is clear that the proof fails.

Your answer from part (c):

Use this page if you need more space.
Clearly indicate the question you are answering.