

## Midterm Exam

NAME:

February 27

If you have any questions about a problem, quietly come to the front of the classroom and ask me. To pace yourself, you should allow on more than 1 minute/point. For example, on a 10 point problem, don't spend more than 10 minutes. Good Luck.

1. (15 points) Give tight asymptotic bounds for  $T(n)$  in each of the following recurrences (i.e. it's sufficient to use  $\Theta$  notation so use the master method whenever you can). You can assume that  $T(1) = \Theta(1)$ . As part of showing your work, you are required to give the value of " $\ell$ " and " $k$ ". You are welcome to do the rest in your head if you want but the more you show the more we can give partial credit if you made a mistake.

(a)  $T(n) = 3T(n/9) + 10 \ln n + 20$

(b)  $T(n) = T(3n/4) + \log_{10} n$

(c)  $T(n) = 4T(n/4) + 5n(\log_2 n)^2 + n\sqrt{n}$

(d)  $T(n) = 9T(n/3) + 10^{\log_{10} n} + \frac{1}{2}n^2 + n(\ln n)^{10}$

Take each of the four answers above and put each one in one of the blanks below so that the resulting statement is true.

\_\_\_\_\_ =  $O$ (\_\_\_\_\_) =  $O$ (\_\_\_\_\_) =  $O$ (\_\_\_\_\_)

2. (10 points) Below is most of a divide-and-conquer algorithm to compute the maximum element in an array of  $n$  elements. Complete the below pseudo-code and then analyze the asymptotic complexity of the resulting algorithm.

```
int maxValue(int[] A,int left,int right){  \\find max value of A[left]...A[right]
    if (left==right)
        return A[left];
    int mid = (int) (left+right)/2
    int ans1 = maxValue(left,mid);
    int ans2 = maxValue(mid+1,right);

}
```

Here is my analysis for the worst-case asymptotic time complexity of the above algorithm.

3. (10 points) In this problem you are to compute the expected number of array elements examined by a binary search when searching in array  $A[0..6]$  for an element  $x$  where:

array position	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	$x$ not in A
probability $x$ in that position	.05	.15	.1	.15	.05	.3	.1	.1

While the arithmetic should be easy to do without a calculator you are welcome to just leave your answer in a form like  $5 \times .4 + 2 \times .1 + \dots$ .

4. (15 points) In this problem you are to analyze the asymptotic time complexity for the following divide-and-conquer algorithm. **Do NOT waste your time figuring out what this does. It does not matter!**

Answer the questions below the code.

```
int foo(int[] A){
    n = A.length;
    if (n==1)
        return A[0];

    /** divide ***/
    int half = (int) n/2
    int[] A1 = new int[half];
    int[] A2 = new int[n-half];
    int[] A3 = new int[half]

    for (int i=0; i < half; i++){
        for (int j=0; j <= 1; j++){
            if (j==0) A1[i] = A[2i];
            else A2[i] = A[2i+1];
        }
    }

    A2[n-half-1] = A[n-1];

    for (int i=0; i < half; i++)
        A3[i] = A1[i]*A2[i];

    /** end of divide ***/

    b1 = foo(A1);
    b2 = foo(A2);
    b3 = foo(A3);

    if (b3 > b1*b2)
        return b3;
    else
        return b1*b2;
}
```

- What is the asymptotic time complexity of the code (marked above) to divide the original input into the subproblems? Use the space on the right above to briefly show your work or explain how you got your answer.
- Write a recurrence relation to express the time complexity  $T(n)$  of `foo` on an input of  $n$  elements.
- Solve the recurrence using the master method.

5. (15 points) You are given  $n$  coins identical in appearance; either all are genuine or exactly one is fake. It is unknown whether the fake coin is heavier or lighter than the genuine ones. Your model of computation to solve this problem is as follows. You can only learn about the coins through a provided `weigh(set1, set2)` method that takes as input two sets of coins `set1` and `set2` and returns one of the three possibilities: the two sets of coins have the same weight, `set1` is heavier, or that `set2` is heavier.

Using the decision tree lower bound technique give the best lower bound you can on the number of times `weigh` must be called in the worst-case to determine if any coin is fake, and if so which coin is fake and whether it is heavier or lighter than the genuine ones.

6. (20 points) You have an inventory system in which each item has unique barcode that is an 8 digit (base-10) number along with associated information such as the item name, number in stock, etc. Design a data structure to implement the following methods in the stated time. You should clearly describe your data structure and how each of the below methods will be implemented. Be sure to go over the expected time complexity for your implementation of each method.

Operation	Expected Time Requirement
<code>insertItem(int barcode, itemData data)</code>	$O(1)$
<code>removeItem(int barcode)</code>	$O(1)$
<code>findItem(int barcode)</code>	$O(1)$
<code>printSortedInventory</code>	$O(n)$

where  $n$  is the number of items currently in the system. The semantics of `insertItem` and `removeItem` should be clear. The method `findItem` should return the data provided when the given barcode was inserted. Finally, `printSortedInventory` should print a list of all items sorted by barcode along with the associated information.

More Space for Problem 6:

Problem	Points Possible	Points Received
1	15	
2	10	
3	10	
4	15	
5	15	
6	20	
total	85	