Computer Science Foundation Exam

December 16, 2005

Computer Science

Section 1B

No Calculators!

KEY

Solutions and Grading Criteria

Name: _____

SSN: _____

Score: ____ /50

In this section of the exam, there are four (4) problems. You must do all of them.

The weight of each problem in this section is indicated with the problem.

Partial credit cannot be given unless all work is shown and is readable.

Be complete, yet concise, and above all be neat.

1. [16 points – 10 points part (a), 6 points part (b)]

Ackermann's function grows faster than any primitive function. It is defined below. (a) Construct a recursive function that will correctly produce the Ackermann number for values of **m** and **n**. (b) Trace the execution of your function for the values **m = 1** and **n = 5** and determine the value produced by the function.

$$A(0,n) = n+1 \quad for\ n \geq 0$$
$$A(m,0) = A(m-1,1) \quad for\ m > 0$$
$$A(m,n) = A(m-1,\ A(m,n-1)) \quad for\ m,n \geq 0$$

(a) One possible solution is:

```
#include <stdio.h>

int A(int a, int b);

int main (void)
{
int a = 1; int b = 5; int result;
result = A(a,b);
printf("\n%s%d\n","A[1,5] returns: ", result);
return 0;
}

int A(int m, int n)
{
   if ((m == 0) && (n >= 0))
   {
      return(n+1);
   }
   else if ((m > 0) && (n == 0))
   {
       return(A(m-1,1));
   }
   else if ((m >= 0) && (n >= 0))
   {
       return(A(m-1, A(m, n-1)));
   }
}
```

There is certainly more than one way to solve this problem, the solution given is simply one technique. Make sure their code is recursive and deduct the entire 9 points if it is not recursive. Be sure to check the stopping condition.

For part (b), their trace is based on their function, so if they messed up the function, the trace will probably also be incorrect, but it should help them discover any errors in their code as well.

(b) One possible tracing mechanism:

A(1,5) = A(0, A(1,4))
= A(0, A(0, A(1,3)))
= A(0, A(0, A(0, A(1,2))))
= A(0, A(0, A(0, A(0, A(1,1)))))
= A(0, A(0, A(0, A(0, A(0, A(1,0))))))
= A(0, A(0, A(0, A(0, A(0, A(0,1))))))
= A(0, A(0, A(0, A(0, A(0, 2)))))
= A(0, A(0, A(0, A(0, 3))))
= A(0, A(0, A(0, 4)))
= A(0, A(0, 5))
= A(0, 6)
= 7

Value produced by A(1,5) is: 7

2. [10 points – 5 points part (a), 5 points part (b)]

Find the closed from expression in terms of the parameter **n,** or an exact value if the summation limits are known, for each of the following summations. Show all of your work…an answer alone is not sufficient to receive full credit for a problem.

(a) Find the closed form for: $\sum_{i=6}^{4n-2} 3i$ ANSWER: $24n^2 - 18n - 42$

$$\sum_{i=6}^{4n-2} 3i = 3\sum_{i=1}^{4n-2} i - 3\sum_{i=1}^{5} i = \frac{3(4n-2)(4n-1)}{2} - \frac{3(5)(6)}{2}$$

$$= \frac{48n^2 - 36n - 84}{2} = 24n^2 - 18n - 42$$

Deduct 1 point for simple arithmetic errors. Since calculators are not available, they may leave the solutions in any correct form that no longer includes any summations without penalty.

(b) Find the closed form for: $\sum_{i=0}^{3n+1} 4i - 3$ ANSWER: $18n^2 + 9n - 2$

$$\sum_{i=0}^{3n+1} 4i - 3 = 4\sum_{i=0}^{3n+1} i - 3\sum_{i=0}^{3n+1} 1 = \frac{4(3n+1)(3n+2)}{2} - 3(3n+2)$$

$$= \frac{36n^2 + 36n + 8}{2} - 9n - 6 = 18n^2 + 9n - 2$$

3. [12 points – 6 points part(a), 6 points part (b)]

Answer each of the following "timing" questions concerning an algorithm of a particular order and a data set of a particular size. Assume that the run time of the algorithm is affected only by the size of the data set and not its composition and that **N** is an arbitrary integer. Show all of your work…an answer alone is not sufficient to receive full credit for a problem.

(a) Assume that an $O(n^2)$ algorithm runs 20 msec on a data set of size 8. What size problem will require 125 msec to execute using the same algorithm?

ANSWER: Data set size = 20

$$\frac{N_{old}^2}{T_{old}} = \frac{N_{new}^2}{T_{new}} = \frac{8^2}{20} = \frac{N_{new}^2}{125} = N_{new}^2 = \frac{64 \times 125}{20} = \frac{8000}{20} = 400, \quad so \;\; N = 20$$

(b) Your friend tells you that her data set required 48 seconds to run on an $O(\log_2 n)$ algorithm. When you execute the same algorithm with your data set of size 16 the algorithm took 32 seconds to execute. What size was her data set?

ANSWER: Data set size =64

$$\frac{N_{old}}{T_{old}} = \frac{N_{new}}{T_{new}} = \frac{\log_2 N_{old}}{48} = \frac{\log_2(16)}{32} = \frac{4 \times 48}{32} = 6, \quad so \;\; 2^6 = 64$$

For part (a), a common mistake will be to forget that the equation is solved for N^2 and they will not return the square root of this value. Similarly, for part (b), a common mistake will be to forget that the equation is solved for $\log_2 N$ and they will forget to return the proper power of 2. For each of these types of mistakes deduct 3-4 points from the problem. The remaining 2-3 points for the problem should be assigned to setting up the equation properly.

4. [12 points – 10 points part(a), 2 points part (b)]

Answer parts(a) and (b) below assuming a binary tree with the following node structure

```
struct treenode {
int data;
struct treenode * left, *right;
};
```

(a) Write a recursive function which adds 50 to the data value for every node having only a single child and returns the tree.

(b) What is the Big-Oh complexity of your function in terms of the number of nodes in the tree?

Solution to part (a)

```
struct treenode * change (struct treenode *p) {
        if (p == NULL)
           return 0;
         if ((p->right==NULL) != (p->left == NULL))
                p->data = p->data +50;
        change(p->left);
        change(p->right);
 }
```

Solution to part (b)

O(n) where *n* is the number of nodes in the tree.