Computer Science Foundation Exam

May 6, 2016

Section I A

COMPUTER SCIENCE

**NO books, notes, or calculators may be used,
and you must work entirely on your own.**

Name: _____

UCFID: _____

Question #	Max Pts	Category	Passing	Score
1	10	DSN	7	
2	10	ANL	7	
3	10	ALG	7	
4	10	ALG	7	
5	10	ALG	7	
TOTAL	50		35	

You must do all 5 problems in this section of the exam.

Problems will be graded based on the completeness of the solution steps and <u>not</u> graded based on the answer alone. Credit cannot be given unless all work is shown and is readable. Be complete, yet concise, and above all <u>be neat</u>.

1) (10 pts) DSN (Recursive Functions)

Complete the function below, to create a ***recursive*** function ***productDigits*** that takes in 1 parameter, a non-negative integer, *n*, and returns the product of *n*'s digits. For example *productDigits(232)* will return 12 (2 x 3 x 2), and *productDigits(13999019)* will return 0.

```
int productDigits(int number){




}
```

2) (10 pts) ANL (Summations)

a) (5 pts) Determine the value of the following summation, in terms of n: $\sum_{i=1}^{2n}(4i + 7)$. Express your final answer as a polynomial in the form $an^2 + bn$, where a and b are integers.

_____

b) (5 pts) Determine the value of the summation below:

$\sum_{i=21}^{100}(3i + 1)$

_____

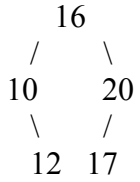
3) (10 pts) ALG (Stacks)

A stack of ***positive integers*** is implemented using the struct shown below. Using this implementation of the stack write the ***push*** and ***peek*** functions. ***Assume that when a struct stack is empty, its top variable is equal to -1.***

```
#define MAX 12

struct stack{
   int top;   /* indicates index of top */
   int nodes[MAX] ;
};

// Attempts to push value onto the stack pointed to by s.
// If the stack is full 0 is returned and no action is taken.
// Otherwise, value is pushed onto the stack and 1 is returned.
int push(struct stack* s, int value){









}

// Returns the value at the top of the stack. If the stack is
// empty, -1 is returned.
int peek(struct stack* s){









}
```

4) (10 pts) ALG (Data Structures Tracing - AVL Trees, Binary Heaps)

a) (5 pts) Show the result of inserting 19 into the AVL tree below. Draw a box around your final resulting tree.

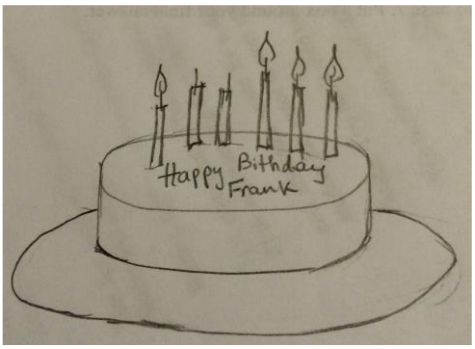


b) (5 pts) In a binary heap of 100 elements, how many elements are at a depth of 6 (lowest level) from the root of the heap? (Note: the depth of an element is the number of links that have to be traversed from the root of the tree to reach it.)

5) (10 pts) ALG (Base Conversion)

a) (5 pts) Convert the hexadecimal number AF2E9 to binary without first converting to the base 10 equivalent.

b) (5 pts) Frank is the team-lead for the software testing team at his job. He is celebrating his birthday. Some of his co-workers have baked a cake for the celebration and thought that it would be really cool to put candles on his cake to represent his age in binary. An unlit candle represents the 0 bit. From the pic of the cake below, how old is Max?



Computer Science Foundation Exam

May 6, 2016

Section I B

COMPUTER SCIENCE

NO books, notes, or calculators may be used,
and you must work entirely on your own.

Name: _____

UCFID: _____

Question #	Max Pts	Category	Passing	Score
1	10	ANL	7	
2	10	ANL	7	
3	10	DSN	7	
4	10	DSN	7	
5	10	ALG	7	
TOTAL	50		35	

You must do all 5 problems in this section of the exam.

Problems will be graded based on the completeness of the solution steps and <u>not</u> graded based on the answer alone. Credit cannot be given unless all work is shown and is readable. Be complete, yet concise, and above all <u>be neat</u>.

1) (10pts) ANL (Algorithm Analysis)

Determine the ***best case*** run time in terms *n* for each of the following functions/operations.

a) Finding the maximum value in an unsorted linked list of *n* elements _____

b) Inserting an item into a binary search tree of *n* elements _____

c) Inserting an item into a binary heap of *n* elements _____

d) Sorting an array of *n* elements using Merge Sort _____

e) Deleting an element from a circular linked list of *n* elements _____

Determine the ***worst case*** run time in terms of *n* for each of the following functions/operations.

f) Deleting an item from an AVL tree of *n* elements _____

g) Deleting the minimum item from a binary min heap of *n* elements _____

h) Inserting an item into a binary search tree of *n* elements _____

i) Sorting an array of *n* elements using Heap Sort _____

j) Deleting an element from a doubly linked list of *n* elements _____

2) (10 pts) ANL (Algorithm Analysis)

a) (5 pts) Given that a function has time complexity $O(n^2)$, if the function takes 338 ms for an input of size 13000, how long will the same function take for an input of size 8000?

b) (5 pts) What is the run-time of the segment of code below, in terms of the variables *n* and *k*? Please provide a Big-Oh bound and briefly justify your answer. (Assume k has already been defined as set to a value prior to the code segment shown.)

```
int i, total = 0;
for (i=0; i<n; i+=2) {
    int start = k;
    while (start > 0) {
        total += ((k|i) & start);
        start /= 2;
    }
}
```

3) (10 pts) DSN (Linked Lists)

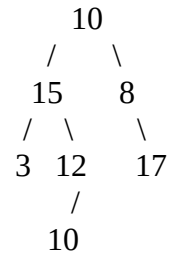
Write a function, `mode`, that takes in a pointer to the front of a linked list storing integers, and returns the mode of the list of integers. Recall that the mode of a list of values is the value that occurs most frequently. You may assume that all of the integers in the list are in between 0 and 999, inclusive. If there is more than one mode, your function must return the smallest of all of the modes. (For example, if the list contains the values 2, 4, 3, 2, 2, 4, 1, and 4, your function must return 2 and should NOT return 4, since both 2 and 4 occur three times in the list but 2 is smaller than 4.) Hint: declare an auxiliary array inside of the mode function. ***You may assume that the list pointed to by front is non-empty.***

Use the struct definition provided below.

```
#include <stdlib.h>
#include <stdio.h>
#define MAX 1000

typedef struct node {
    int value;
    struct node* next;
} node;

int mode(node* front) {




}
```

4) (10 pts) DSN (Binary Trees)

For this problem you will write a modified inorder traversal of a binary tree. In a regular inorder traversal, you simply visit the nodes of the tree "in order". Consider the added task of adding up all of the numbers as you see them in the traversal, one by one, and printing out each intermediate sum. For example, if the input binary tree was:



the corresponding inorder traversal would visit 3, 15, 10, 12, 10(root), 8 and 17, in that order. For the added task, the traversal should print out 3, 18, 28, 40, 50, 58 and 75, respectively, the running sums after visiting each value.

Complete the function below, ***recursively***, so that is performs the given task. One way to accomplish this is to have the function take in a second value, `prevSum`, representing the previous sum of values prior to visiting the given node, and also to have the function return the sum of the nodes in its subtree.

```
typedef struct treenode {
    int value;
    struct treenode *left;
    struct treenode *right;
} treenode;

int inorderSum(treenode* root, int prevSum) {

    if (root == NULL) _____________________________ ;

    int sum = 0;

    sum += _______________(_______________, _________________);

    sum += ____________________________;

    printf("%d ", ____________________);

    sum += _______________(_______________, ______________________);

    return ____________________;

}
```

5) (10 pts) ALG (Sorting)

For this question, implement the (very slow) sorting algorithm described below:

1. Randomly choose two array indexes, i and j, with i < j. (If i and j are equal, choose again.)
2. If array[i] > array[j], swap the two values.
3. Check if the array is sorted. If it's not, go back to step 1. If it is, return.

Recall that the function call `rand()` returns a random non-negative integer, so `rand()%n` will equal a random integer in between 0 and n-1.

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

int min(int a, int b) {if (a < b) return a; return b;}
int max(int a, int b) {if (a > b) return a; return b;}

void randomSort(int* array, int length) {




}
```

Computer Science Foundation Exam

May 6, 2016

Section II A

DISCRETE STRUCTURES

NO books, notes, or calculators may be used,
and you must work entirely on your own.

Name: _____

UCFID: _____

Question	Max Pts	Category	Passing	Score
1	15	PRF (Induction)	10	
2	10	PRF (Logic)	7	
3	15	PRF (Sets)	10	
4	10	NTH (Number Theory)	7	
ALL	50		34	

You must do all 4 problems in this section of the exam.

Problems will be graded based on the completeness of the solution steps and not graded based on the answer alone. Credit cannot be given unless all work is shown and is readable. Be complete, yet concise, and above all be neat.

1) (15 pts) PRF (Induction)

Use strong induction to prove that, for every non-negative integer, n, F_n is even if and only if 3 divides n. Here F_n is the Fibonacci sequence given by the recurrence:

$$F_0 = 0$$
$$F_1 = 1$$
$$F_n = F_{n-1} + F_{n-2}$$

2) (10 pts) PRF (Logic)

Here we have discovered some "rules of inference" that aren't valid. Invalidate them by finding counter-examples that make each premise true but make the conclusion false.

(a) $p \vee q$
$\neg p \vee \neg q$
$\therefore p$

(b) $(p \rightarrow q) \rightarrow r$
$\neg r$
$\therefore \neg p$

(c) $p \rightarrow (q \rightarrow r)$
$\neg r$
$\therefore p$

3) (15 pts) PRF (Sets)

Show for finite sets A, B, C that if $B \cup C \subseteq A$ and $A \times B \subseteq A \times C$, then $B \subseteq C$.

Disprove for finite sets A, B, C that if $A \times B \subseteq A \times C$, then $B \subseteq C$.

4) (10 pts) NTH (Number Theory)

Prove that for any two distinct primes a and b there exists some prime c distinct from both a and b such that $c \mid (a + b)$.

Computer Science Foundation Exam

May 6, 2016

Section II B

DISCRETE STRUCTURES

**NO books, notes, or calculators may be used,
and you must work entirely on your own.**

Name: _____

UCFID: _____

Question	Max Pts	Category	Passing	Score
1	15	CTG (Counting)	10	
2	10	PRB (Probability)	7	
3	15	PRF (Functions)	10	
4	10	PRF (Relations)	7	
ALL	50		34	

You must do all 4 problems in this section of the exam.

Problems will be graded based on the completeness of the solution steps and <u>not</u> graded based on the answer alone. Credit cannot be given unless all work is shown and is readable. Be complete, yet concise, and above all <u>be neat</u>.

1) (15 pts) CTG (Counting)

Please leave your answers in factorials, permutations, combinations and powers. Do not calculate out the actual numerical value for any of the questions. Justify your answers.

(a) (5 pts) If seven people are to be seated in a single row with ten chairs, how many unique arrangements are possible? (For example, if the people are labeled P_1 through P_7 and we use B to denote an empty chair, then one possible arrangement is P_1, B, B, P_2, P_4, P_3, P_7, P_5, B, P_6. Notice that the people are distinguishable but the empty chairs aren't.)

(b) (10 pts) How many of the arrangements from part (a) guarantee that none of the empty chairs are adjacent to one another?

2) (10 pts) PRB (Probability)

A mad scientist has created a probability-driven lock that can be permanently affixed to all manner of things (gym bags, lockers, vehicles, door locks, and so on). The lock has a five-digit LED display and a single button, and works as follows:

Once the lock is closed, if you want to open it, you press the button, and it generates a random sequence of five digits on its display. (Each digit is a random integer on the range zero through nine.) If the product of all the digits is odd, the lock opens. Otherwise, you have to press the button again to generate another random sequence of digits.

If you press the button three times without getting the lock to open, it seals itself shut forever.

If you put one of these locks on something, what is the probability that you'll actually be able to get it to open again?

3) (15 pts) PRF (Functions)

(a) (4 pts) Define finite sets A and B that satisfy ***all three*** of the following criteria simultaneously:

1. A and B are non-empty.
2. There exists a surjective function from A to B.
3. Every possible surjective function from A to B is also injective.

A = _____

B = _____

(b) (4 pts) Define finite sets A and B that satisfy ***all three*** of the following criteria simultaneously:

1. A and B are non-empty.
2. There exists an injective function from A to B.
3. It is **impossible** to define an injective function from A to B that is also surjective.

A = _____

B = _____

(c) (4 pts) Define finite sets A and B that satisfy ***all three*** of the following criteria simultaneously:

1. A and B are non-empty.
2. There exists an injective function from A to B.
3. It is **possible** to define an injective function from A to B that is also surjective.

A = _____

B = _____

(d) (3 pts) Define finite sets A and B that satisfy ***both*** of the following criteria simultaneously:

1. A and B are non-empty.
2. It is **impossible** to define a function from A to B that is **not** injective.

A = _____

B = _____

4) (10 pts) PRF (Relations)

(a) (3 pts) What three properties must a relation satisfy in order to be a partial ordering relation?

(b) (7 pts) Consider the relation $\mathcal{R}$ on $\mathbb{Z}^+$ defined as follows: For all positive integers x and y, $(x, y) \in \mathcal{R}$ if y is divisible by x. Prove or disprove that $\mathcal{R}$ is a partial ordering relation.