Computer Science Foundation Exam

September 5, 2020

Section I A

DATA STRUCTURES

ONLINE EXAM

Directions: You may either directly edit this document, or write out your answers in a .txt file, or scan your answers to .pdf and submit them in the COT 3960 Webcourses for the Assignment "Section I A". Please put your *name, UCFID and NID* on the top left hand corner of each document you submit. Please aim to submit 1 document, but if it's necessary, you may submit 2. Clearly mark for which question your work is associated with. If you choose to edit this document, please remove this cover page from the file you submit and make sure your *name, UCFID and NID* are on the top left hand corner of the next page (first page of your submission).

Question #	Max Pts	Category	Score
1	10	DSN	
2	10	DSN	
3	5	ALG	
TOTAL	25		

You must do all 3 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. For each coding question, assume that all of the necessary includes (stdlib, stdio, math, string) for that particular question have been made.

Name: ______________________
UCFID: ______________________
NID: ________________________

1) (10 pts) DSN (Dynamic Memory Management in C)

Suppose we are planning a party and we would like to create an array to store our list of supplies. Currently our list is stored in a text file with the name of each item to be purchased on a line by itself. Write a function called make_grocery_list that reads these items from a file and stores them in a two-dimensional character array. Your function should take 2 parameters: a pointer to the file and an integer indicating the number of grocery items in the file. It should return a pointer to the array of items. Be sure to allocate memory for the array dynamically and only allocate as much space as is needed. You may assume that all of the strings stored in the file representing grocery items are alphabetic strings of no more than 127 characters (so the buffer declared is adequate to initially read in the string).

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

char ** make_grocery_list (FILE *ifp, int numItems) {

     char buffer[128];
     char **list = NULL;
     int i;









}
```

2) (10 pts) ALG (Linked Lists)

Suppose we have a queue implemented as a doubly linked list using the structures shown below. Use head for the front of the queue and tail for the end of the queue.

```
struct node {
    int data;
    struct node* next, *prev;
}

struct queue {
     int size;
     struct node *head, *tail;
}
```

Write a dequeue function for this queue. If the queue is NULL or is already empty, return 0 and take no other action. If the queue isn't empty, dequeue the appropriate value, make the necessary adjustments, and return the dequeued value. **(Note: You must free the node that previously stored the dequeued value.)**

```
int dequeue(queue *thisQ) {





}
```

3) (5 pts) DSN (Stacks)

Convert the following infix expression to postfix using a stack. Show the contents of the stack at the indicated points (1, 2, and 3) in the infix expression.

$$\mathbf{A * B - C} \overset{1}{\quad} \mathbf{+ (D - ((E} \overset{2}{\quad} \mathbf{* F) / G))} \overset{3}{\quad} \mathbf{+ H}$$

1	2	3

Resulting postfix expression:

Computer Science Foundation Exam

September 5, 2020

Section I B

DATA STRUCTURES

ONLINE EXAM

Directions: You may either directly edit this document, or write out your answers in a .txt file, or scan your answers to .pdf and submit them in the COT 3960 Webcourses for the Assignment "Section I B". Please put your *name, UCFID and NID* on the top left hand corner of each document you submit. Please aim to submit 1 document, but if it's necessary, you may submit 2. Clearly mark for which question your work is associated with. If you choose to edit this document, please remove this cover page from the file you submit and make sure your *name, UCFID and NID* are on the top left hand corner of the next page (first page of your submission).

Question #	Max Pts	Category	Score
1	10	DSN	
2	10	ALG	
3	5	ALG	
TOTAL	25		

You must do all 3 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. For each coding question, assume that all of the necessary includes (stdlib, stdio, math, string) for that particular question have been made.

Name: ____________________

UCFID: ____________________

NID: ______________________

1) (10 pts) DSN (Binary Search Trees)

Complete the function on the next page so that it takes the root of a binary search tree (*root*) and an integer (*key*) and, if *key* is in the tree, returns the smallest integer in the BST that is **strictly greater than** *key*. If *key* is not present, or if there is no integer in the tree greater than *key*, simply return *INT_MIN* (which is defined in *limits.h*).

Your function must be **iterative** (not recursive), with a worst-case runtime that does not exceed **O(n)** (so, you can't drop the elements into an array and sort them in order to solve the problem).

You may assume the tree does not contain any duplicate values. However, *root* could be NULL.

For example:

```
        18           next_up(root, 18) would return 20
       /  \          next_up(root, 1) would return 4
      4    20        next_up(root, 4) would return 7
     / \             next_up(root, 10) would return 18
    1  10            next_up(root, 20) would return INT_MIN
       /             next_up(root, 9) would return INT_MIN
      7
```

In your solution, you may make as **single call** to either of the following functions. Assume they're already written and that they each have a worst-case runtime of O(n):

```
// Takes the root of a binary search tree (possibly the root of a
// subtree within a larger BST) and returns the smallest value in that
// (sub)tree. If the tree is empty, it returns INT_MAX.
int find_min(node *root);

// Takes the root of a binary search tree (possibly the root of a
// subtree within a larger BST) and returns the largest value in that
// (sub)tree. If the tree is empty, it returns INT_MIN.
int find_max(node *root);
```

An incomplete version of the function and *node* struct are provided on the following page, along with ten blanks for you to fill in to complete the solution. **Note that one of these blanks ought to be left blank and has been included so that part of the solution isn't given away.** Thus, each blank is worth one point, and for at least one of the ten blanks, leaving it blank is the only way to get credit for it.

(…*continued from previous page*)

```
// Assume these are included from limits.h.
#define INT_MAX 2147483647
#define INT_MIN -2147483648

typedef struct node
{
   struct node *left;
   struct node *right;
   int data;
} node;

int next_up(node *root, int key)
{

   node *parent = ________ ;

   while (root != NULL)
   {
      if (key < root->data)
      {
          ______________________ ;

          ______________________ ;

      }
      else if (key > root->data)
      {
          ______________________ ;

          ______________________ ;
      }
      else
      {
         if ( ___________________________ )

            return _______________________________ ;

         else if (parent != NULL)

            return ________________ ;
         else
            return _________________ ;
      }
   }

   return ___________________ ;
}
```

2) (10 pts) ALG (Heaps)

a. (3 pts) Fill in the array representation of a heap that meets all of the following conditions:

   i. The minheap contains exactly eight integer values, without any duplicate values.
   ii. The minheap does *not* contain the value 14.
   iii. Inserting 14 into the minheap would cause us to incur the **best**-case possible runtime for insertion.

Note: Index 0 is omitted because in the traditional storage of a heap using an array, that index is not used.

Index	1	2	3	4	5	6	7	8
Value								

b. (3 pts) Fill in the array representation of a heap that meets all of the following conditions:

   i. The minheap contains exactly eight integer values, without any duplicate values.
   ii. The minheap does *not* contain the value 98.
   iii. Inserting 98 into the minheap would cause us to incur the **worst**-case possible runtime for insertion.

Note: Index 0 is omitted because in the traditional storage of a heap using an array, that index is not used.

Index	1	2	3	4	5	6	7	8
Value								

c. (4 pts) Is it possible to draw a single minheap that simultaneously meets all of the conditions given in parts (a) **and** (b) of this problem? If so, draw such a minheap. If not, explain why not. If you're giving an explanation, be brief and clear, but also complete.

If possible, fill in this table, if not, explain why it's not possible below the table and leave it blank:

Index	1	2	3	4	5	6	7	8
Value								

3) (5 pts) ALG (Tries)

Consider inserting the following words into an initially empty trie, where nodes are created ONLY if they are necessary. Note that an empty trie is a NULL pointer and has zero nodes. How many nodes will be created after all these words are inserted, including the root node?

cat
part
do
art
cart
car
dog
dart
ark
park

No need to include any drawing, just give your final answer, but be very careful, since the grading is completely based on your numeric answer and no work at all.

Total number of nodes = _______

Computer Science Foundation Exam

September 5, 2020

Section II A

ALGORITHMS AND ANALYSIS TOOLS

ONLINE EXAM

Directions: You may either directly edit this document, or write out your answers in a .txt file, or scan your answers to .pdf and submit them in the COT 3960 Webcourses for the Assignment "Section II A". Please put your *name, UCFID and NID* on the top left hand corner of each document you submit. Please aim to submit 1 document, but if it's necessary, you may submit 2. Clearly mark for which question your work is associated with. If you choose to edit this document, please remove this cover page from the file you submit and make sure your *name, UCFID and NID* are on the top left hand corner of the next page (first page of your submission).

Question #	Max Pts	Category	Score
1	10	ANL	
2	10	ANL	
3	5	ANL	
TOTAL	25		

You must do all 3 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. For each coding question, assume that all of the necessary includes (stdlib.h, stdio.h, math.h, string.h) for that particular question have been made.

Name: ______________________
UCFID: ______________________
NID: ________________________

1) (10 pts) ANL (Algorithm Analysis)

Consider the following function that takes a list of **unique** integers for input

1. The list is empty return 0
2. Choose a random integer, `x`, from the list
3. Place every value from the list less than the value `x` in list 1 every value greater than `x` in list 2.
4. Run this function on list 1 and store the result in a variable called `left_answer`.
5. Run this function on list 2 and store the result a variable called `right_answer`.
6. Add to the answer the function run on list 2
7. Return the value `left_answer + right_answer + x`

What is the best case runtime and the worst case runtime for the above function, in terms of the input list size, **n**? What standard algorithm taught in COP 3502 is this algorithm closest to? In terms of that standard algorithm, what does the return value of this function represent? Provide proof of the runtimes, but in doing so, you may use results about known algorithms from COP 3502 without proving those results.

2) (10 pts) ANL (Algorithm Analysis)

For a certain known data structure a lookup takes O($\sqrt{n}$) time, where n is the number of stored items. For a data set of 8,000,000 items the runtime for a look up was approximately 10ms. On a different data set the look up took 40ms. About how many **items** do you expect to be stored in the second data set?

3) (5 pts) ANL (Summations)

What is the closed form of the following summation? Express your result as a polynomial, in n, in standard form.

$$\sum_{i=0}^{n^4} 6i$$

Computer Science Foundation Exam

September 5, 2020

Section II B

ALGORITHMS AND ANALYSIS TOOLS

ONLINE EXAM

Directions: You may either directly edit this document, or write out your answers in a .txt file, or scan your answers to .pdf and submit them in the COT 3960 Webcourses for the Assignment "Section II B". Please put your *name, UCFID and NID* on the top left hand corner of each document you submit. Please aim to submit 1 document, but if it's necessary, you may submit 2. Clearly mark for which question your work is associated with. If you choose to edit this document, please remove this cover page from the file you submit and make sure your *name, UCFID and NID* are on the top left hand corner of the next page (first page of your submission).

Question #	Max Pts	Category	Score
1	10	DSN	
2	5	ALG	
3	10	DSN	
TOTAL	25		

You must do all 3 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. For each coding question, assume that all of the necessary includes (stdlib, stdio, math, string) for that particular question have been made.

Name: ______________________
UCFID: ______________________
NID: ________________________

1) (10 pts) DSN (Recursive Coding)

If Annabelle plays a video game for t minutes, she'll earn f(t) points. (The function f is provided below.) f is a non-decreasing function. (This means that if a < b, then f(a) ≤ f(b).) Annabelle wants to earn at least ***target*** number of points, but since she gets in trouble if she plays too much, she would like to play the least number of minutes that allows her to score at least ***target*** number of points. Complete the function below to be ***recursive*** and efficient (max # of recursive calls should be around 30) to solve the problem. You may assume that 0 < target < 60000, thus it's guaranteed that $0 < t < 10^9$ (for the function f given below.)

```
#include <math.h>

int f(int t) {
    return (int)(2*sqrt(t)+log(t));
}

int minPlay(int target) {
    return minPlayRec(target, 0, 1000000000);
}

int minPlayRec(int target, int low, int high) {




















}
```

2) (5 pts) ALG (Sorting)

Show the result after each iteration of performing Insertion Sort on the array shown below. For convenience, the result after the first and last iteration are provided. The first row of the table contains the original values of the array.

Iteration	Index 0	Index 1	Index 2	Index 3	Index 4	Index 5	Index 6	Index 7
0	13	6	9	27	3	15	1	12
1	6	13	9	27	3	15	1	12
2								
3								
4								
5								
6								
7	1	3	6	9	12	13	15	27

3) (10 pts) DSN (Bitwise Operators)

A company is looking to hire an employee based on answers to a questionnaire. The questionnaire has 20 yes/no questions (labeled from question 0 to question 19) and an applicant's set of responses is stored as a single integer such that the i^{th} bit is set to 1 if the response to question i is yes, and it's set to 0 if the response to question i is no. For example, if an applicant answered yes to questions 0, 2, 3, 5, 8, and 10 and no to the other 14 questions, her responses would be stored as the single integer 1325, since $00000000010100101101_2 = 1325$. For all questions, the company prefers yes answers to no answers. However, since it's unlikely that a candidate will answer yes to all of the questions, they have developed a modified scoring system for each candidate. The company has ranked each of the 20 questions in order of preference from most important to least important. This ranking is an array, ***preferences***, that stores a permutation of the integers from 0 to 19, inclusive. For example, if preferences[0] = 8, preferences[1] = 10 and preferences[2] = 1, then the company cares most about the answer to question 8, second most about the answer to question 10 and third most about the answer to question 1. A candidate's score is simply the maximum number of the most important questions they answered yes without a single no. For example, the sample candidate described above would have a score of 2, because she said yes to questions 8 and 10, but no to question 1, which was third on the preference list. Any candidate who said no to question 8 would be assigned a score of 0. Complete the function below so it returns the score of the applicant whose answers are stored in the formal parameter ***applicant***. The formal parameter ***preferences*** is an array of size 20 which stores the order of importance of the questions as previously described. **(Note: You must use bitwise operators to earn full credit for this question.)**

```
#define SIZE 20

int score(int preferences[], int applicant) {





}
```