Computer Science Foundation Exam

May 18, 2024

Section A

BASIC DATA STRUCTURES

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

PLEASE USE CAPITAL LETTERS IN WRITING YOUR NAME

Last Name: ________________________________

First Name: ________________________________

UCFID: ____________________________________

Question #	Max Pts	Category	Score
1	5	ALG	
2	10	DSN	
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.

1) (5 pts) ALG (Dynamic Memory Management in C)

Given the following C code.

```
int **arr1 = malloc(3 * sizeof(int *));
for(int i = 0; i < 3; ++i)
     arr1[i] = malloc(2 * sizeof(int));
int *arr2 = malloc(3 * 2 * sizeof(int));
```

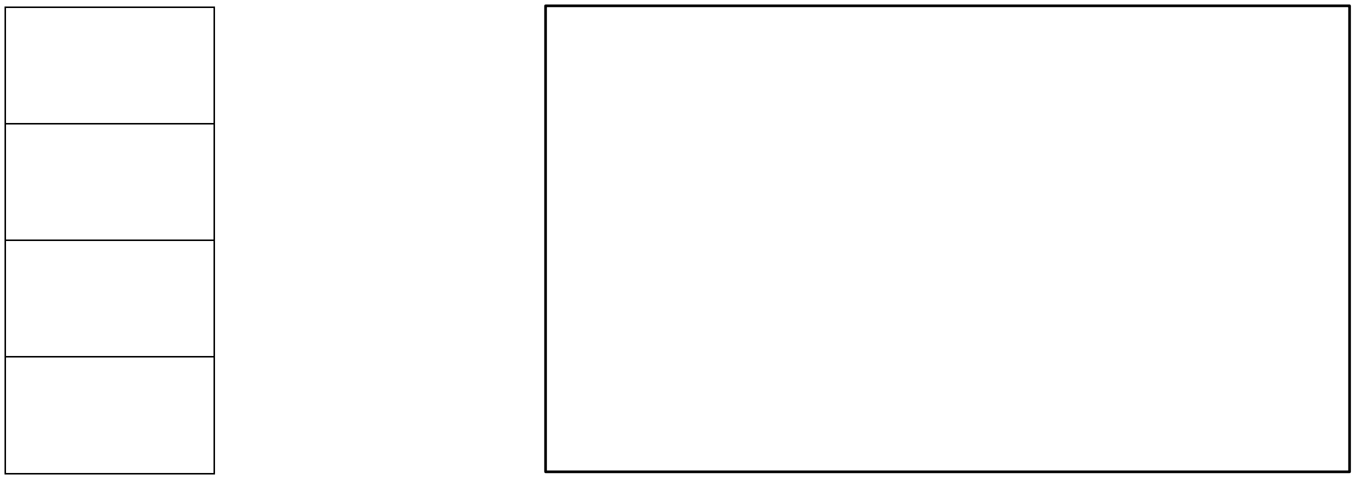
Answer the following questions about the above lines of code.

a) Does `arr1` and `arr2` require the same number of total bytes allocated to be stored in the heap space? Please write yes or no. No reason is needed.

b) Are **<u>all</u>** the addresses associated with `arr1` (excluding `arr1` itself on the stack space) adjacent in memory? Please write yes or no only. No reason is needed.

c) Are **<u>all</u>** the addresses associated with `arr2` (excluding `arr2` itself on the stack space) adjacent in memory? Please write yes or no only. No reason is needed.

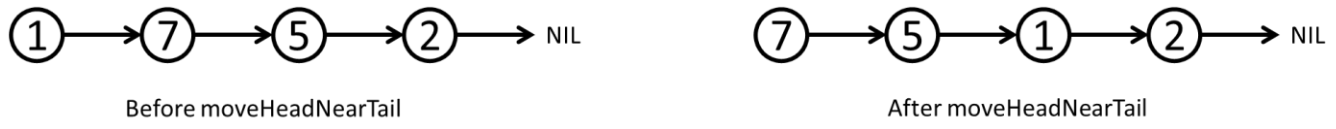
d) Complete the following stack and heap space visual below showing how the memory state looks for both `arr1` and `arr2` from the above lines of code (after all lines execute properly). For each box you draw in the heap space, indicate what type of variable is stored in the box.



Stack Space **Heap Space**

2) (10 pts) DSN (Linked Lists)

Given a singly integer linked list, complete the following user defined function definition `moveHeadNearTail`. The user defined function moves the head node of some singly linked list that is passed to the second last position of the list (the node that comes before the tail node itself). The following figure shows a sample scenario. The function returns the head of the modified linked list. **You may assume the linked list pointed to by head has at least 3 elements in it.**



```
typedef struct node_s {
    int data;
    struct node_s* next;
} node_t;

node_t * moveHeadNearTail(node_t * head) {






















}
```

3) (10 pts) DSN (Stacks)

You are playing a scoring game that uses a LIFO approach for keeping track of scores. Here are how the scores are managed. You are given a character array (string) of moves where each index represents some rule for managing the score. You must go through the array in index order to properly manage the score.

Here are the rules for managing the score:

- If the move is some character representing an integer (0-9 both inclusive), record the integer itself.
- If the move is the character '+'. You will need to retrieve the last 2 scores recorded and compute the sum. After computing the sum, you will need to add this sum to the recorded list of scores.

Once all moves have been processed, you will need to compute the total sum of all the scores and return this value. For example, if the string passed to the function is "25+3++1", then after processing the first plus sign the corresponding stack of values from bottom to top would be [2, 5, 7]. After processing the second plus sign the corresponding stack of values from bottom to top would be [2, 5, 7, 3, 10]. After processing the string completely, the stack would store [2, 5, 7, 3, 10, 13, 1]. The sum of these values, 41, should be returned. Complete the following function definition that simulates this scoring game. You may assume that the string header file is included. The provided functions and stack structure are here to assist you with completing this function. **Note: There exists a solution that doesn't use the stack and this or any such solution will get full credit, if correctly implemented.** The parameter represents the character array of moves, and is guaranteed to be valid. Namely, the string will consist solely of digits and plus signs, and if the string has any plus signs, they will only appear in an index 2 or greater (meaning that there will be two previous scores to add.)

This code is fairly long, so go ahead and write your code on the following page. The structs and functions you may use are listed on this page, below:

```
typedef struct node_s {
      int data;
      struct node_s * next;
} node_t;
typedef struct {
      node_t * top;
} stack_t;

// Initializes a stack to be empty.
void init(stact_t* s);

// Pushes data onto the stack pointed to by s.
void push(stack_t* s, int data);

// Removes and returns the integer at the top of the stack pointed to by s.
int pop(stack_t* s);

// Returns 1 if and only if the stack pointed to by s is empty. Returns 0
// otherwise.
int empty(stack_t* s);
int computeScore(char * moves) {
















}
```

Computer Science Foundation Exam

May 18, 2024

Section B

ADVANCED DATA STRUCTURES

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

PLEASE USE CAPITAL LETTERS IN WRITING YOUR NAME

Last Name: ________________________________

First Name: ________________________________

UCFID: _____________________________________

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

You must do all 3 problems in this section of the exam.

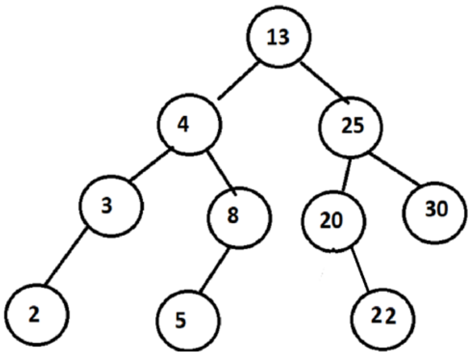
Problems will be graded based on the completeness of the solution steps and not graded based on the answer alone. Credit cannot be given unless all work is shown and is readable. Be complete, yet concise, and above all be neat. For each coding question, assume that all of the necessary includes (stdlib, stdio, math, string) for that particular question have been made.

1) (5 pts) ALG (Binary Trees)

a) Draw **a binary search tree** of with 5 nodes (storing positive integers) that has the maximum possible height. (1 pt)

b) Draw another binary search tree with 7 nodes (storing positive integers) that has the minimum possible height. (1 pt)

c) Re-draw the following binary search tree after deleting the root node. (3 pts)



2) (10 pts) DSN (Heaps)

a) (6 pts) Consider the following struct that represents a binary minheap.

```
typedef struct heap {
    int* elements; //points to the array of heap elements
    int capacity; // total size of the array
    int size;     // actual number of elements in the heap
} heapStruct;
```

Also, the following functions are available to you, and you are free to call them as needed:

```
- int removeMin(heapStruct *h);//removes the smallest item from the heap
pointed to by h.
- int size(heapStruct* h); // returns the number of elements in the heap
pointed to by h.
```

Write a function called heapsort that takes a pointer to a heap, and returns those values in a sorted integer array. At the end of the function, the heap pointed to by h will be empty.

```
int* heapsort(heapStruct* h) { //complete this function








}
```

b) (4 pts) Specify the worst run-time when efficiently implemented for the following operations:

Operation	**Run-time**
Building a binary heap from an unordered array of size **n** using heapify	O(_______)
Inserting an item into a binary heap with **n** items.	O(_______)
Deleting the minimum item from a binary heap with **n** items	O(_______)
Heapsort of **n** items.	O(_______)

3) (10 pts) DSN (Tries)

As an afficionado of Wordle, you're curious how many five letter words there are in a dictionary stored in a trie. Write a recursive function that takes in a pointer to a trie node and an integer k, representing the depth of the node in the trie, and **returns the number of five letter words** stored within that subtrie. A wrapper function is provided which makes the initial recursive call on the root node of the trie storing the dictionary. Please use the struct shown below. Assume all necessary includes.

```
typedef struct trieNode {
    int isWord;
    struct trieNode* children[26];
} trieNode;

int num5LetterWrapper(trieNode* root) {
    return num5Rec(root, 0);
}

int num5Rec(trieNode* root, int k) {









}
```

Computer Science Foundation Exam

May 18, 2024

Section C

ALGORITHM ANALYSIS

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

PLEASE USE CAPITAL LETTERS IN WRITING YOUR NAME

Last Name: ________________________________

First Name: ________________________________

UCFID: ____________________________________

Question #	Max Pts	Category	Score
1	5	ANL	
2	10	ANL	
3	10	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 <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>. For each coding question, assume that all of the necessary includes (stdlib, stdio, math, string) for that particular question have been made.

1) (5 pts) ANL (Algorithm Analysis)

What is the **worst-case** Big O runtime for the following function, in terms of the input parameter, **n**? (You may assume that the array pointed to by list is of length **n**.) In order to receive full credit, you must use words to explain your reasoning AND arrive at the correct answer.

```
int mystery(int* list, int n) {

    int i = 0, j = 1;

    if (n < 2) return 0;

    while (j < n) {

        while (i < j && list[i] < list[j]) i++;
        j++;
    }

    return i;
}
```

REASON:

RUN-TIME: (____________)

2) (10 pts) ANL (Algorithm Analysis)

An algorithm that processes a list of size n takes $O(\sqrt{n}lgn)$ time. On Shannon's computer, when she runs the algorithm on a list of size $\mathbf{n = 2^{16}}$**,** her computer takes **c** milliseconds. (Shannon is very secretive, so she hasn't told you the value of **c** unfortunately!) **In terms of c,** how long, **in milliseconds,** should we expect the algorithm to take on her computer when she is processing a list of size $\mathbf{2^{20}}$? (Your answer should be of the form kc, where k is a positive real number.)

3) (10 pts) ANL (Recurrence Relations)

Using the iteration technique, determine the Big-Oh solution to the recurrence relation below, in terms of ***n***.

$$T(n) = 2T\left(\frac{n}{2}\right) + n^3, \text{ for } n > 1$$
$$T(1) = 1$$

Computer Science Foundation Exam

May 18, 2024

Section D

ALGORITHMS

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

PLEASE USE CAPITAL LETTERS IN WRITING YOUR NAME

Last Name: ________________________________

First Name: ________________________________

UCFID: ____________________________________

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.

1) (10 pts) DSN (Recursive Coding)

Write a recursive function that determines if player X, who goes first, can win a game of tic-tac-toe (played on a 3 by 3 int board). You can use the following helper functions. You don't have to implement any of the listed helper functions. **Functions that do not use recursion will receive 0 points**. Note: board is a 3 by 3 array of integers, storing either EMPTY(0), X(1), or O(2). The three given functions return the current state of the board, as it is, from player X's perspective, not what "could happen" in the future. myTurn is 1, when it's X's turn and 0 when it's O's turn. You will have to place and unplace X's and O's in your solution. Finally, the intention here is that player 'O' plays pessimistic. Namely, if there is ANY set of moves that players X and O could make from the current state of the board that result in X winning, the function should return 1.

```
#define EMPTY 0
#define X 1
#define O 2

int XWin(int ** board); // returns 1 if I have won and 0 otherwise
int XLose(int ** board); // returns 1 if I have lost and 0 otherwise
int tied(int ** board); // returns 1 if the board is in a tied state

int canXWin(int ** board, int myTurn) {

























}
```

2) (10 pts) ALG (Sorting)

(a) (1 pt) Which of the sorting algorithms (listed in part d) could encounter problems if an array can contain duplicates? (Specifically, for four of the algorithms, whether or not there are duplicates in the array don't alter the run-time of the algorithm on individual cases, but one of the algorithms, in its original form, is definitively affected.)

(b) (2 pts) What problem could be encountered?

(c) (2 pts) Pick one of the algorithms that aren't affected by duplicates and explain why it runs similarly with or without duplicates.

(d) (5 pts) What is the worst case runtime for the following sorting algorithms on an array with ***n*** distinct values? Please list your answers with Big-Oh notation, using proper conventions.

Quick	_______
Bubble	_______
Insertion	_______
Merge	_______
Selection	_______

3) (5 pts) ALG (Base Conversion)

Convert 277 in base 8 to base 16. **Please show your work and put a box around your final answer.**