COP 3502 Section 2 Final Exam Summer 2020

1) (10 pts) A class has ***n*** students, and the i^{th} student has taken s_i tests. This information is stored in a file where the first line has the value of ***n*** and the following ***n*** lines have information about each student. On the i^{th} of these lines, the first value is s_i. This is followed by s_i integers, all in between 0 and 100, inclusive, representing the test scores. Here is a sample file:

```
4
10 100 80 90 100 90 90 95 80 100 95
3 87 93 90
5 100 90 90 100 90
8 90 90 90 90 80 80 85 85
```

Write a segment of code that reads in this information from standard input (assuming file redirection on the command prompt as shown in class) into an array of arrays that is dynamically allocated to have precisely the correct number of slots to store the data. Name your array studentscores. All necessary variables have been declared for you below. Please use only these in your solution. **Note: you are ONLY allocating the memory and reading in the information. No need to do anything else.** Also, the way this is written, you wouldn't have stored the length of each separate array, but don't worry about that for the purposes of this question.

```
int** studentscores;
int i, j, numStudents, numScores;
```

2) (10 pts) Write a function that takes in a pointer to a linked list of nodes storing integers and a variable named value, and returns the number of nodes in the list storing that value. For example, if a list pointed to by listPtr stores 2, 6, 2, 3, 4, 2, 6, and 6 and value = 6, your function should return 3, since 6 appears in the list 3 times. Please use the struct and function prototype provided below:

```
typedef struct node {
    int data;
    struct node* next;
} node;

int countInList(node* listPtr, int value) {
    // fill in code
}
```

3) (5 pts) What is the value of the following post-fix expression? (Note: You will be graded solely on your final response.)

3 9 + 4 * 12 2 / / 2 8 5 - * +

4) Write a ***recursive*** function that counts and returns the number of nodes in a binary tree with the root `root`, that store an even value. Please use the struct shown and function prototype shown below. (For example, if the tree rooted at root stored 2, 3, 4, 8, 13, 18 and 20, the function should return 5, since there are five even values [2,4,8,18,20] stored in the tree.

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

int numEvenNodes(node* root) {
    // Fill in code
}
```

5) Consider inserting the following values into a min heap, in this order: 12, 3, 19, 2, 1. Show the final locations for each value in the array storing the heap. (Recall that we store heaps in arrays using 1-based indexing and typically leave the 0 index blank.) Note: Only the answer will be graded for this question.

index	1	2	3	4	5
value					

6) Write a function that takes in a root node of a trie and returns the length of the longest word stored in that trie. Use the struct given and function prototype given below.

```
typedef struct trienode {
    int isWord;
    struct trienode* next[26];
} trienode;

int maxWordLength(trienode* root) {
    // Fill in code
}
```

7) What are the **worst case run times** of each of the following operations? Make sure to list your answer in terms of the appropriate variables in the prompt. Note that on occasion, some of the run times won't be dependent on some of the variables listed in the prompt.

(a) Inserting an item to the front of a linked list of ***n*** elements. _____

(b) Sorting ***n*** integers using Quick Sort. _____

(c) Merging a sorted list of ***a*** elements with a sorted list of ***b*** elements. _____

(d) Inserting an item into a binary heap of ***n*** elements. _____

(e) Deleting an item from a binary search tree of ***n*** elements. _____

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

(g) Printing out each permutation of ***n*** elements, where printing one value takes O(1) time. _____

(h) Calculating $\mathbf{a^b \bmod c}$ where individual multiplications and mods take O(1) time, using fast modular exponentiation. _____

(i) A floodfill on a grid with ***r*** rows and ***c*** columns. _____

(j) Dequeuing an item from a queue of ***n*** elements. _____

8) An algorithm answers a query on a database of n elements in $O(\sqrt{n})$ time. For a database of size $n = 10^6$, it takes 2 seconds to perform 5,000 queries. How many seconds should it take to answer 3,000 queries on a database of size $n = 10^8$? (Note: all credit is based on the work and not the answer.)

9) Determine the following summation in terms of n. Please express your answer in the form, $an^2 + bn + c$, for constants a, b and c.

$$\sum_{i=n}^{3n-1} (5i + 7)$$

10) Consider the problem of finding the mode (most frequently occurring value) in an array. John attempts to solve the problem recursively. His strategy is to recursively call his mode function on the left half of the array and the right half of the array, and then use these answers to calculate the mode of the whole array. Assuming that his function only returns the mode and nothing else, why is his strategy doomed to fail?

11) Sorting

(a) (5 pts) In a Merge Sort of 8 elements, the Merge function gets called 7 times. Consider a Merge Sort being executed on the array shown below. What does the array look like right **AFTER** the sixth call to the Merge function completes?

index	0	1	2	3	4	5	6	7
value	40	27	12	18	11	99	31	16

Index	0	1	2	3	4	5	6	7
Value								

(b) (5 pts) Consider sorting the array below using a Bubble Sort. Show the contents of the array after each iteration of the algorithm completes. (Note: after the first iteration, the maximum array value should be in its correct spot.)

Index	0	1	2	3	4	5
Original	33	18	22	9	15	14
After 1st						
After 2nd						
After 3rd						
After 4th						
After 5th						

12) For the purposes of this question, a permutation of size n is any ordering of the integers 0, 1, 2, …, n-1. We define a spaced-out permutation of size n to be a permutation such that two consecutive terms in the permutation differ by at least 2. For example, [0, 2, 4, 1, 3] is a spaced out permutation of size 5, and [5, 2, 4, 0, 3, 1] is a spaced out permutation of size 6, but [3, 0, 2, 1] is not a spaced out permutation since the 2 and 1 are adjacent and differ by only 1. Fill in the blanks below so that all spaced out permutations of size N are printed out. (Note: You may use the abs function from the math library. The function takes in a single integer and returns its absolute value.)

```
#include <stdio.h>
#include <math.h>
#define N 6

int main(void) {
    int perm[N], used[N];
    for (int i=0; i<N; i++) used[i] = 0;
    printSpaced(perm, 0, used);
    return 0;
}

void printSpaced(int perm[], int k, int used[]) {

    if (k == N) {
        for (int i=0; i<N; i++) printf("%d ", perm[i]);
        printf("\n");
        return;
    }

    for (int i=0; i<N; i++) {

        if ( ____ || ___________________________________________ )) {

            used[i] =  ____;

            perm[k] =  ____;

            printSpaced( ______, ______, ______);

            used[i] = 0;
        }
    }
}
```