CSC 212 Spring 2019 Midterm Exam 2 Study Guide

On Monday we will be holding a review session for the midterm where we will go over answers to these questions.

1. Pointers

   (a) Write a small function which takes a reference to an int and a regular int as an argument, sets the referenced int equal to the other int then returns void.

   ```
   void set(int *a, int val)
   ```

   (b) Declare an array of 12 ints on the *heap*. Fill it with 1-12 in reverse order using your above function then deallocate it.

(c) Describe the difference between ”dot notation” vs ”arrow notation”
ie. myList.print() vs myList→ $print()$

2. Recursion

   (a) What is recursion? What types of problems are good for recursion?

   (b) Draw out a call tree for the following function given foo(5). What is the Big-O runtime of this function?

   ```
   void foo(int n) {
       if(n < 2) {
           return n;
       } else {
           return foo(n-1) + foo(n-2);
       }
   }
   ```

   (c) Write a recursive function which takes an array if ints and its length as arguments and prints out the odd numbers. What is the runtime of this function? Justify your answer.

   ```
   void onlyOdd(int arr[], int n)
   ```

3. Binary Search

   (a) What is the runtime of binary search? Is it possible to improve the runtime why or why not?

4. Unimodal Array

   (a) Describe the difference between a strongly and weakly unimodal array.

   (b) How can you most efficiently find the maximum value of each type of array?

5. Recurrences
   Solve the following relations using the unrolling method
   (a) T(0)=1, T(n)=T(n-4)-4
   (b) T(1)=1, T(n)=T(n/3)+1
   (c) T(1)=1, T(n)=2T(n/2)+n
   (d) T(0)=1, T(n)=T(n-1)+2
   (e) Define and solve the recurrence of the following function. Assume each print statement takes 1 time.

```
void bar(int n) {
    if(n == 0) {
        std::cout << "fin" << std::endl;
    } else {
        std::cout << n << " ";
        bar(n-1);
        bar(n-1);
        bar(n-1);
    }
}
```

   (f) What is the Big-O runtime of this function?

6. Mergesort
   (a) What is the worst case runtime of mergesort?
   (b) What is the main advantage of mergesort vs quicksort?

7. Quicksort
   (a) Illustrate the state of the array below after applying the quicksort partition method from the lecture slides, once.

```
[13, 15, 19, 20, 9, 11, 8, 6, 4, 14, 1, 10]
```

   (b) What index value will be returned by partition?
   (c) What is the worst case of this particular partition method? What input would cause this behavior?
   (d) What is one partition choice which is less likely to encounter worst-case behavior?
   (e) In what situation would you be likely to choose quicksort over mergesort?

8. Linked Lists

(a) You are implementing a CDLL with members *head* and *tail* and Nodes with members *data, next*, and *prev.* Write a method which appends a node to the end of the list. What is the runtime of this function?

```
void CDLL::append(int val)
```

(b) Write a method for a SLL class with members *head* and *tail*, which returns the average of the elements in the list. Nodes have members *next* and *data* (which stores a double). What is the runtime of this function?

```
double SLL::average()
```

(c) What is one advantage of a linked list vs a traditional array? One weakness?

9. Stacks

(a) Draw out the contents of each stack after the function has concluded

```
int main() {
    stack<int> p;
    stack<int> q;

    p.push(1);
    p.push(3);
    q.push(2);

    for(int i = 0; i < p.size(); i++) {
        q.push(p.top());
        q.push(i);
    }
    q.pop()

    /* evaluate stack contents at this point */

    return 0;
}
```

(b) Explain the difference between LIFO and FIFO data structures. Which of these is stack?