SCC120 Fundamentals of Computer Science
Term 2 Coursework

Deadline for submission: Sat 3rd June, 2023, at 22:00 (Week 15)

Submission is electronic as **pdf** file on Moodle.

This coursework is worth **20%** of SCC120. There are two sections in this coursework. Please answer all questions. Section A is worth 20 marks and section B is worth 20 marks. The questions in each section tend to be easier at the beginning and then the questions near the end are intended to be more difficult.

You must do this coursework completely by yourself. In this instance, it means you are not even allowed to discuss the coursework questions with other students. Your submitted files will automatically be checked for plagiarism. You are allowed to ask the instructors about these questions. But we can only clarify the questions or give you suggestions if you get stuck, and we will try not to tell you whether you have the correct answer (to be fair to all students).

Section A

1. This question is on searching and sorting, as we studied in class.

   a. Please provide an array with 8 elements, and the element you wanted to search, making sure that *sentinel search on sorted arrays* is faster than *binary search*. **[1 mark]**

   b. Please provide an array with 8 elements, and the element you wanted to search, making sure that *linear search* is faster than *binary search*. **[1 mark]**

   c. Consider the array [-5.3, 0, 8, 10, 14, 18, 20] and say you wanted to search for 16. Let's say you are using binary search. How many iterations of its loop will be executed before the search terminates? **[1 mark]**

   d. Say you are searching for a number in a sorted array. Say you also know that the number will either occur in the first place or the last place of the array. Which of *linear search*, *sentinel search on sorted*, and *binary search* will be the most efficient in the worst case and what is its worst case complexity? **[2 marks]**

   e. For the insertion sort, what is its best case complexity? What is its worst case complexity? Please give an example for each case. **[2 marks]**

2. This question is on the correctness of algorithms and the NP problem, as we studied in class.

   a. **True or False?** Every NP problem could be solved in polynomial time. **[1 mark]**

   b. **True or False?** For insertion sort of an array A of length n, the loop invariant for the outer loop expresses exactly that when we are about to begin its i^{th} iteration, the subarray A [0…i-1] is sorted. **[1 mark]**

   c. **True or False?** Decision problem is always easier than optimization problem. **[1 mark]**

3. For each of the following pairs of functions, state which one grows faster asymptotically for input of size n (in other words, is worse in terms of efficiency).

   a. $f(n) = n\log_2 n + 53$ and $g(n) = 10^8 \log_2 n$ **[1 mark]**
   b. $f(n)=n! + n$ and $g(n)=n^n + n$ **[1 mark]**
   c. $f(n)=n^n$ and $g(n)=2^n + n^3$ **[1 mark]**
   d. $f(n) = \log_2(n)$ and $g(n) = n/10$ **[1 mark]**

4. Answer the following questions about the time complexity of algorithms.

a. **True or False**? The time complexity is more important than space complexity when choosing algorithms. **[1 mark]**

b. **True or False?** The time complexity determines the running time of the algorithm. **[1 mark]**

5. This question is about computing the time complexity of algorithms.

a. Give the worst case Big O characterization of the following algorithm. The input to the algorithm consists of two arrays of strings A & B. Assume *strlen* (a function which returns the length of a string) and *print* are both O(k) in the worst case. **[2 marks]**

```
n = A.length;
m = B.length;
for (i = 0; i < n; i++) {
    for (j = 0; j < m; j++) {
        if(strlen(A[i]) == strlen(B[j])
            continue;
        else
            print (A[i], B[j]);
    }
}
```

b. This question involves doing some research on your own based on things mentioned tangentially in class. Imagine an algorithm that prints both the number of subsets of a set of integers and the subsets themselves. What is the minimum time any such algorithm would take in the worst case? Express using the correct notation, specifically, using one of the following: O, Ω, Θ.

[2 marks]

Section B

In this section, you can assume the most efficient implementation of an ADT (unless otherwise stated). So for example, a queue can be implemented in many ways, but we assume that the remove() and add() operations are O(1) if they are implemented efficiently.

True or False. You only have to write the answer (i.e. no explanations needed). [6 marks total for this part, 1 mark each]

1) In a Stack ADT, the push() method removes an item from the top of the stack.

2) For a Queue implemented with a circular buffer, there is no limited size to the number of items you can add.

3) A reduced graph of a graph G always has the same set of nodes as those in G.

4) If a full undirected graph G has n nodes, then a minimal spanning tree of G always has n-1 edges.

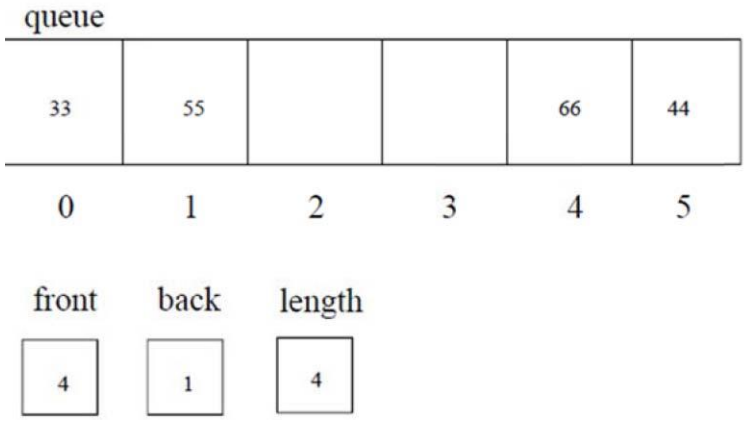
5) In a strict binary tree with four levels, there is a minimum of nine nodes in the whole tree.

6) It is impossible to have a tree where the pre-order traversal, in-order traversal, and post-order traversal of nodes all give same results.

For this part, you only have to write or draw the answers to the following questions (i.e. no explanations needed). [6 marks total for this part, 1 mark each]

7) You have a Stack implemented as a linear array (see figure below). The "top" variable is 4. What happens after five successive calls of pop()? Draw the updated array in your answer and also indicate the updated value of "top".

8) You have a Queue implemented as a circular buffer (see figure below). What happens after two successive calls of remove()? Draw the updated array in your answer and also indicate the updated values of the three variables, *front*, *back*, and *length*.



9) Draw one example of a directed graph with five nodes that has a density of 0.2 (there can be many possible graphs satisfying this, so just draw one example).

10) Draw one example of an undirected graph with six nodes where you would *not* be able to find a minimal spanning tree (i.e. you would *not* be able to "span" through or visit every node).

11) For a binary tree, the pre-order traversal is "abcdefg", the in-order traversal is "cbdaegf", try to draw the tree.

12) In a tree T with maximum out-degree 4, if there are 10 nodes with an out-degree of 4, 10 nodes with an out-degree of 3, 1 node with an out-degree of 2, and 10 nodes with a out-degree of 1, what is the number of leaf nodes in tree T?

Write the O() complexities for each case below. No explanations are needed. [6 marks total for this part, 1 mark each]

13) For a Set implemented with a linked list, what is the O() for the add() method?

14) For a Queue implemented as a circular linked list (as given in the lecture slides), what is the O() for removing an element?

15) In a priority queue (implemented as a linked list with front and back pointers and nothing else), you usually remove the item that has the highest priority. If you want to remove the item with the highest priority, what is O ()?

16) There is a graph G with N nodes and E edges, and there is a potential subgraph H (that has a smaller number of nodes and edges than G). If we want to check if H is really a subgraph of G (and return yes or no), what is the O() complexity for doing this?

17) If a binary tree with height D, what is the number of nodes in O() notation? Note that here O() is not related to the runtime.

18) For the “adjacency matrix” representation of trees, if there are N nodes in a tree, What is the time complexity of post-order traversal in O() notation?

Programming Question

19) How to calculate the height of a tree? Please explain your idea at first, and then provide the pseudo code. **[2 marks]**