COMP 2804: Assignment 2

Due Date: Sunday, February 28th at 11:59PM

School of Computer Science | Carleton University

Your assignment should be submitted online on cuLearn as a single .pdf file. Make the filename YourLastname_YourStudentID.pdf. Indicate clearly your name and student number on the assignment's first page. No late assignments will be accepted. You can type your assignment or you can upload a scanned copy of it. Please, use a good image capturing device. Make sure that your upload is clearly readable. If it is difficult to read, it will not be graded!

Question 1 [10 marks]

A binary tree is

- either one single node
- or a node whose left subtree is a binary tree and whose right subtree is a binary tree.

Prove that any binary tree with n leaves has exactly 2n - 1 nodes.

Question 2 [10 marks]

In Section 4.4, we have seen the recursive algorithm Gossip(n), which computes a sequence of phone calls for the persons P_1, P_2, . . . , P_n, for any integer $n \geq 4$. Give an iterative, i.e., non-recursive version of this algorithm in pseudo-code. Your algorithm must produce exactly the same sequence of phone calls as algorithm Gossip(n).

Question 3 [10 marks]

Is the following algorithm for the Gossip Problem correct? Prove or disprove. What is its complexity (i.e., number of phone calls made)? Is it optimal?

Step 1.: P_1 calls each of the other persons, i.e., P_1 calls P_i, $i > 1$

Step 2.: P_1 calls each of the other persons, i.e., P_1 calls P_i, $i > 1$

Question 4 [10 marks]

The recursive algorithm Fib, shown in Figure 1, takes as input an integer $n \geq 0$ and returns the $n - th$ Fibonacci number f_n.

Algorithm FIB(n)**:**

    if $n = 0$ or $n = 1$
    then $f = n$
    else $f = \text{FIB}(n-1) + \text{FIB}(n-2)$
    endif;
    return f

Figure 1: Fibonacci Algorithm.

Let a_n be the number of additions made by algorithm Fib(n), i.e., the total number of times the +-function in the else-case is called. Prove that for all $n \geq 0$,

$a_n = f_{n+1} - 1$.

The algorithm is not efficient in terms of the total number of operations carried out. Without you having to give the actual such number, can you pin-point exactly where the inefficiency results from?

Question 5 [10 marks]

Let $n \geq 2$ be an integer and consider a sequence $s_1, s_2, ..., s_n$ of n pairwise distinct numbers. The following algorithm, shown in Figure 2, computes the smallest and largest elements in this sequence:

Algorithm $\textsc{MinMax}(s_1, s_2, \ldots, s_n)$**:**

```
    min = s1;
    max = s1;
    for i = 2 to n
    do if s_i < min          (1)
         then min = s_i
         endif;
         if s_i > max         (2)
         then max = s_i
         endif
    endwhile;
    return (min, max)
```

Figure 2: MinMax Algorithm.

This algorithm makes comparisons between input elements in lines (1) and (2). Determine the total number of comparisons as a function of n.

Question 6 [5 marks]

Prove, for example by induction, that for $n \geq 1$, that

$1 + 2 + 3 + ... + n = n(n + 1)/2$.

Question 7 [5 marks]

Prove, for example by induction, that for $n \geq 1$, that

$1^2 + 2^2 + ... + n^2 = n(n + 1)(2n + 1)/6$.

Question 8 [10 marks]

Let $n \geq 66$ be an integer and consider the set S = {1, 2, . . . , n}.

- Let k be an integer with $66 \le k \le n$. How many 66-element subsets of S are there whose largest element is equal to k?
- Use the result in the first part to prove that $\Sigma_{k=66}^{n}\binom{k-1}{65} = \binom{n}{66}$

Question 9 [10 marks]

Consider the recursively defined function, f, below for integers $n \ge 1$. Write a non-recursive version of this function and show that the two versions are defining the same function..

$$\begin{aligned} f(n) = &\quad 1 && \text{if} \quad n = 1 \\ f(n) = &\quad 2f(n-1) + 5 && \text{if} \quad n > 1 \end{aligned} \tag{1}$$

Question 10 [10 marks]

Consider the non-recursively defined function below for integers $n \ge 0$. Write a recursive version of this function and show that the two versions are defining the same function.

$$f(n) = a \cdot b^n \tag{2}$$

Question 11 [10 marks]

Recall that in MergeSort, we divide a list L of size n into two lists L_1 and L_2 of size $n/2$, we sort L_1 and L_2 recursively, and then merge the sorted L_1 and L_2.

Consider a variant of MergeSort, called MergeSortVar, where we divide a list L of size $n = 3^k$, for some non-negative integer k, into three lists L_1, L_2, and L_3 of size $n/3$, we sort L_1, L_2, and L_3 resursively (using MergeSortVar), and then merge the sorted L_1, L_2, and L_3. Assume the merge operation (of L_1, L_2, and L_3) can be done by using n comparison operations. What is the time complexity of MergeSortVar?

Question 12 [10 marks]

Consider a set S defined recursively in the following way:

- $1 \in S$
- If $n \in S$ then $2n \in S$.

Show that every non-negative integer power of 2 is in S (i.e. $\forall n \ge 0, 2^n \in S$).

End of Assignment 2.