COMP 3804 — Assignment 1

Due: Monday February 7, 23:59.

Assignment Policy:

• Your assignment must be submitted as one single PDF file through Brightspace.

    Use the following format to name your file:

    LastName_StudentId_a1.pdf

• Late assignments will not be accepted. I will not reply to emails of the type "my internet connection broke down at 23:57" or "my scanner stopped working at 23:58", or "my dog ate my laptop charger".

• You are encouraged to collaborate on assignments, but at the level of discussion only. When writing your solutions, you must do so in your own words.

• Past experience has shown conclusively that those who do not put adequate effort into the assignments do not learn the material and have a probability near 1 of doing poorly on the exams.

• When writing your solutions, you must follow the guidelines below.
  
  – You must justify your answers.
  – The answers should be concise, clear and neat.
  – When presenting proofs, every step should be justified.

Some useful facts:
1. for any real number $x > 0$, $x = 2^{\log x}$.

2. for any real number $x \neq 1$ and any integer $k \geq 1$,

   
   $$1 + x + x^2 + \cdots + x^{k-1} = \frac{x^k - 1}{x - 1}. $$

3. For any integer $n \geq 1$, the harmonic number $H_n = \sum_{i=1}^{n} \frac{1}{i}$ satisfies

   $$\ln n \leq H_n \leq 1 + \ln n.$$
Question 1: Write your name and student number.

Question 2: Solve the following recurrence using the unfolding method that we have seen in class. Give the final answer using Big-O notation.

\[ T(n) = \begin{cases} 1 & \text{if } n = 1, \\ n + 5 \cdot T(n/7) & \text{if } n \geq 7. \end{cases} \]

You may assume that \( n \) is a power of 7.

Question 3: The function \( T(n) \) is recursively defined as follows:

\[ T(n) = \begin{cases} 1 & \text{if } 1 \leq n \leq 2, \\ n + T(n/3) + T(2n/3) & \text{if } n \geq 3. \end{cases} \]

Use the recursion tree method that we have seen in class to prove that \( T(n) = \Theta(n \log n) \).

Question 4: The function \( T(n) \) is recursively defined as follows:

\[ T(n) = \begin{cases} 1 & \text{if } n = 1, \\ \frac{n}{n-1} + \frac{n}{n-1} \cdot T(n-1) & \text{if } n \geq 2. \end{cases} \]

Prove that \( T(n) = O(n \log n) \).

Question 5: Consider the following recursive algorithm \textsc{Mystery}(a, b), which takes as input two integers \( a \) and \( b \) with \( a \geq 1 \) and \( b \geq 0 \):

\begin{algorithm}
\textbf{Algorithm} \textsc{Mystery}(a, b):
\begin{algorithmic}
\If{b = 0}
\State c = 1
\ElsIf{b is even}
\State c = \textsc{Mystery}(a^2, b/2)
\Else
\State c = a \cdot \textsc{Mystery}(a, b - 1)
\EndIf
\EndIf
\State return c
\EndIf
\end{algorithmic}
\end{algorithm}

(5.1) Explain why, for any two integers \( a \geq 1 \) and \( b \geq 0 \), algorithm \textsc{Mystery}(a, b) terminates.

(5.2) Let \( a \geq 1 \) and \( b \geq 0 \) be two integers. What is the output of algorithm \textsc{Mystery}(a, b)? Justify your answer.

(5.3) Let \( a \geq 1 \) and \( b \geq 2 \) be two integers. Prove that the recursion depth of algorithm \textsc{Mystery}(a, b) is \( O(\log b) \).
Let $a \geq 2$ and $b \geq 2$ be two integers, and let $n$ be the output of algorithm $\text{Mystery}(a, b)$. Prove that the recursion depth of algorithm $\text{Mystery}(a, b)$ is $O(\log \log n)$.

**Question 6:** The matrices $H_0, H_1, H_2, \ldots$ are recursively defined as follows:

$$H_0 = (1)$$

and for $k \geq 1$,

$$H_k = \begin{pmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{pmatrix}.$$

Thus, $H_0$ is a $1 \times 1$ matrix whose only entry is 1,

$$H_1 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

and

$$H_2 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}.$$

**(6.1)** Let $k \geq 0$ be an integer and let $n = 2^k$. How many entries does the matrix $H_k$ have? Express your answer in terms of $n$.

**(6.2)** Describe a recursive algorithm $\text{Build}$ that has the following specification:

**Algorithm** $\text{Build}(k)$:

**Input:** An integer $k \geq 0$.

**Output:** The matrix $H_k$.

For any positive integer $n$ that is a power of 2, say $n = 2^k$, let $T(n)$ be the running time of your algorithm $\text{Build}(k)$. Derive a recurrence for $T(n)$. Use the Master Theorem to give the solution to your recurrence.

**(6.3)** If $x$ is a column vector of length $2^k$, then $H_kx$ is the column vector of length $2^k$ obtained by multiplying the matrix $H_k$ with the vector $x$.

Describe a recursive algorithm $\text{Mult}$ that has the following specification:

**Algorithm** $\text{Mult}(k, x)$:

**Input:** An integer $k \geq 0$ and a column vector $x$ of length $n = 2^k$.

**Output:** The column vector $H_kx$ (having length $n$).

**Running time:** must be $O(n \log n)$.

Explain why the running time of your algorithm is $O(n \log n)$. You are allowed to use the Master Theorem.

*Hint:* The input only consists of $k$ and $x$. The matrix $H_k$ is not given as part of the input.
**Question 7:** Let \( n \geq 2 \) be an integer, and let \( A[1 \ldots n] \) be an array storing \( n \) pairwise distinct numbers.

It is easy to compute the two smallest numbers in the array \( A \): Using \( n - 1 \) comparisons, we find the smallest number in \( A \). Then, using \( n - 2 \) comparisons, we find the smallest number among the remaining \( n - 1 \) numbers. The total number of comparisons made is \( 2n - 3 \). By a similar argument, we can find the smallest and largest numbers in \( A \) using \( 2n - 3 \) comparisons. In this question, you will show that the number of comparisons can be improved.

(7.1) Consider the following algorithm \texttt{TwoSmallest}(\( A, n \)), which computes the two smallest numbers in the array \( A \):

```plaintext
Algorithm \texttt{TwoSmallest}(A, n):
endif;
for \( i = 3 \) to \( n \)
do if \( A[i] < \) smallest (**)
    then secondsmallest = smallest; smallest = \( A[i] \)
    else if \( A[i] < \) secondsmallest (***)
        then secondsmallest = \( A[i] \)
    endif
endif
endfor;
return smallest and secondsmallest
```

In each of the lines (*), (**), and (***), the algorithm \textit{compares} two input numbers.

Assume that \( A \) stores a uniformly random permutation of the set \( \{1, 2, \ldots, n\} \). Let \( X \) be the total number of \textit{comparisons} made when running algorithm \texttt{TwoSmallest}(\( A, n \)). Observe that \( X \) is a \textit{random variable}. Prove that the expected value of \( X \) satisfies

\[
\mathbb{E}(X) = 2n - \Theta(\log n).
\]

**Hint:** For each \( i = 3, 4, \ldots, n \), use an indicator random variable \( X_i \) that indicates whether or not line (***)) is executed in iteration \( i \).

(7.2) Consider the following algorithm \texttt{SmallestLargest}(\( A, n \)), which computes the smallest and largest numbers in the array \( A \):

...
Algorithm SmallestLargest($A, n$):
endif;
for $i = 3$ to $n$
do if $A[i] <$ smallest  (**)
then smallest = $A[i]$
else if $A[i] >$ largest  (***)
then largest = $A[i]$
endif
endif
endfor;
return smallest and largest

Observe that this algorithm is very similar to algorithm TwoSmallest($A, n$).

Assume that $A$ stores a uniformly random permutation of the set $\{1, 2, \ldots, n\}$. Let $Y$ be the total number of comparisons made when running algorithm SmallestLargest($A, n$). Observe that $Y$ is a random variable. Argue, in a few sentences, that the same analysis as for (7.1) implies that $\mathbb{E}(Y) = 2n - \Theta(\log n)$.

(7.3) Assume that $n \geq 2$ is a power of 2. Furthermore, assume that the array $A[1 \ldots n]$ stores an arbitrary sequence of $n$ pairwise distinct numbers. Describe, in English, an algorithm that computes the two smallest numbers in the array $A$, and that makes $n + \log n - 2$ comparisons. Justify your answer.

Hint: Consider a tennis tournament with $n$ players. The players are numbered from 1 to $n$. For each player $i$, the number $A[i]$ is the ATP-ranking of this player.

The $n$ players play as they do in any Grand Slam tournament: They play against each other in pairs; after any game, the winner goes to the next round and the loser goes home. This is a special tournament: If player $i$ plays against player $j$, then the player with the smaller $A$-value is guaranteed to win.

In this way, the smallest number in $A$ corresponds to the tennis player who wins the tournament. The second smallest number in $A$ corresponds to the second best player. This second best player must lose some game. Who can beat the second best player?

(7.4) Assume that $n \geq 2$ is an even integer. Furthermore, assume that the array $A[1 \ldots n]$ stores an arbitrary sequence of $n$ pairwise distinct numbers. Describe, in English, an algorithm that computes the smallest and largest numbers in the array $A$, and that makes $\frac{3}{2}n - 2$ comparisons. Justify your answer.

---

1 All players are fully vaccinated; thus, Djokovic does not play in this tournament.