CMPU 241 - Analysis of Algorithms

Spring 2019
Assignment 2
Due: Wednesday, Feb. 13th, by 5pm

NAME:

1. (10 points) Consider sorting n numbers stored in array A by first finding the smallest element of A and exchanging it with the number stored in $A[1]$. Then find the second smallest element of A, and exchange it with $A[2]$. Continue in this manner for the first $n-1$ elements of A. This algorithm, called SELECTION-SORT, is given below. Assume the input and output are as specified for the sorting problem on page 16 of our textbook.

```
SELECTION-SORT(A):
1.  n = A.length
2.  for j = 1 to n − 1
3.      smallest = j
4.      for i = j + 1 to n
5.          if A[i] < A[smallest]
6.              smallest = i
7.      exchange A[j] with A[smallest]
```

   (a) (1 point) Give the line number of the basic operation in the algorithm

   The line numbers of the basic operations are lines 4-5. The inner for loop will be run most alongside the if statement. But because the if statement will not always true, line 6 will not run as many times as others.

   (b) (2 points) Why does the outer for loop of the algorithm need to run for only the first $n-1$ elements?

   The outer for loop only runs for n-1 cycles because once there are only two unsorted elements, it is understood that when the lesser of the two is placed at index A[n-1], the largest is now in it's correct position of index A[n] so another check would not be needed.

   (c) (5 points) Are there best-case and worst-case asymptotic running times for SELECTION-SORT? If so, give the best-case and worst-case running times of the algorithm in O and Ω notation. If not, give the worst-case running time in Θ notation. Explain your answer.

   There are not best case and worst case asymptotic running times for selection sort. this is because no matter what we will be looping through the dataset twice and finding the smallest element in the array. Even if the array is sorted these loopings will still need to take place to affirm that the smallest element is indeed in the initial index. This leaves us with a worst case run time of $T(n) = \theta(n^2)$

2. (4 points) Use the definitions of O, Θ, and Ω to determine whether the following assertions are true or false. Briefly justify your answers.

   a. $n^2(n+1)/2 \in O(n^3)$

   Yes, this is true.

   $n^2(n+1)/2 \in O(n^3) = (n^3+n)/2$
   $= \frac{n^3}{2} + \frac{n}{2}$
   $= \frac{n^3}{2}$ ($\frac{n}{2}$ can be eliminated as it is a constant in terms of dataset size)
   $= n^3$ (The $\frac{1}{2}$ can be eliminated as it too is a constant in terms of dataset size)
   $n^3 \in O(n^3)$

   b. $n(n+1)/2 \in O(n)$

   No, this is false.

   $n(n+1)/2 \in O(n) = (n^2+n)/2$
   $= \frac{n^2}{2} + \frac{n}{2}$
   $= \frac{n^2}{2}$ ($\frac{n}{2}$ can be eliminated as it is a constant in terms of dataset size
   $= n^2$ (The $\frac{1}{2}$ can be eliminated as it too is a constant in terms of dataset size)
   $n(n+1)/2 = \Theta(n^2)$
   $\Theta(n^2) \notin O(n)$

   c. $n(n+1)/2 \in \Theta(n^3)$

   No, this is false.

   $n(n+1)/2 \in O(n) = (n^2+n)/2$
   $= \frac{n^2}{2} + \frac{n}{2}$
   $= \frac{n^2}{2}$ ($\frac{n}{2}$ can be eliminated as it is a constant in terms of dataset size
   $= n^2$ (The $\frac{1}{2}$ can be eliminated as it too is a constant in terms of dataset size)
   $n^2 \notin \Theta(n^3)$
   Because Θ is the average runtime, to be in the set above n^2 would need to be equal to n^3, in this case it is clearly not and thus it is false.

   d. $n(n+1)/2 \in \Omega(n)$

   Yes, this is true.

   As found above, $n(n+1)/2 = \Theta(n^2)$.
   Considering Θ is the tight bound, or the average curve sitting bellow the upper bound (O) and lower bound (Ω). Thus because Ω is the lower bound in this case and we have found $n(n+1)/2 = \Theta(n^2)$, Θ is the average bound in this case and can be understood to be in the set of $\Omega(n)$ because Ω is the lower bound.

3. (4 points) For each of the following functions, indicate the class $\Theta(g(n))$ the function belongs to. Show your work in simplifying these expressions to arrive at each answer.

   a. $(n^2+1)^{10}$

   $(n^2+1)^{10} \approx (n^2)^{10} = n^{20}$
   Thus, $(n^2+1)^{10} \in \Theta(n^{20})$

   b. $\sqrt{10n^2+7n+3)}$
   $\sqrt{10n^2+7n+3)} = \sqrt{10n^2} + \sqrt{7n} + \sqrt{3}$
   $= \sqrt{10n^2}$
   $= \sqrt{n^2}$
   $= n$
   Thus, $\sqrt{10n^2+7n+3)} \in \Theta(n)$

   c. $2^{n+1} + 3^{n-1}$
   $2^{n+1} + 3^{n-1} = 2^n * 2 + 3^n * \frac{1}{3}$
   $= 3^n$
   Thus, $2^{n+1} + 3^{n-1} \in \Theta(3^n)$

   d. $\lfloor \lg n \rfloor$
   $\lfloor \log_2 n \rfloor \approx \log_2 n$
   Thus, $\lfloor log_2 n \rfloor \in \Theta(\log_2 n)$

4. (4 points) Answer Yes or No and provide a brief justification for each question below:

   a. Is $2^{n+1} \in O(2^n)$?

   Yes. $2^{n+1} = 2^n * 2 = 2^n \in \Theta(2^n)$
   Because Θ is the average and O is the upper bound, this is true.

   b. Is $2^{2n} \in O(2^n)$?
   No.
   $2^{2n} = (2^n)^2 = 2^n * 2^n$
   In this case you can not eliminate either of the 2^n's as constants and thus $2^{2n} \notin O(2^n)$