CMSC 351 Summer 2025 Homework 2

Due Monday 21 July 2025 by 11:59pm on Gradescope.

Directions:

- Homework must be done on printouts of these sheets and then scanned properly, or via Latex, or by downloading, writing on the PDF, and uploading. If you use Latex please do not change the Latex formatting.
- Do not use your own blank paper!
- The reason for this is that Gradescope will be following this template to locate the answers to the problems so if your answers are organized differently they will not be recognized.
- Tagging is automatic, you will not be able to manually tag.

1. Use the appropriate limit theorem to prove that: [10 pts]

$$n^2 + n = \Theta(4n^2 + \ln n + 2)$$

Make sure you specifically say how the theorem applies.

Solution:

2. Here we will construct the proof that:

$$\text{If } \lim_{n\to\infty} \tfrac{f(n)}{g(n)} = L > 0 \text{ then } f(n) = \Omega(g(n))$$

Your job is to identify some details and finish the proof.

Proof Start:

Suppose $\lim_{n\to\infty} \frac{f(n)}{g(n)} = L > 0$. Then by definition of the limit we know:

$$\forall \epsilon > 0, \exists n_0 \geq 0, \text{if } n \geq n_0 \text{ then } L - \epsilon < \frac{f(n)}{g(n)} < L + \epsilon$$

Proof Continues:

(a) Since this is true for all $\epsilon > 0$ write down the $\exists$ part of the statement for $\epsilon = \frac{L}{2}$. [4 pts]

Solution:

(b) We only need one of the inequalities from the right side of the above expression. Rewrite the $\exists$ part of the statement with just that one but solve for $f(n)$. You might have to think a bit to understand which inequality will be the important one. [5 pts]

Solution:

(c) We have now proved $f(n) = \Omega(g(n))$. Which B does the job? [4 pts]

Solution:

3. Here is the divide-and-conquer pseudocode for the MCS, abbreviated - see the notes for the full version.

```
function mcs(A,L,R)
    if L == R:
        return(A[L])
    else:
        C = (L+R) // 2
        Lmax = mcs(A,L,C)
        Rmax = mcs(A,C+1,R)
        Smax = straddle max
        m = max([Lmax,Rmax,Smax])
        print(Lmax,Smax,Rmax,m)
        return(m)
    end if
end function
```

(a) If we call `mcs([-1,4,-2,3,8],0,4)` what will the values be in the order they are printed? [16 pts]

(b) Suppose we call `mcs(A,0,3)` where `A` has length 4 and the output is: [8 pts]

`3,5,2,5,4,3,-1,4,5,9,4,9`

What is the list `A`?

(c) Very briefly explain why it would not be possible for the following sequence to appear in any output: [8 pts]

`...,3,1,2,5,4,10,11,...`

Solution:

4. Here is a modified version of Kadane's Algorithm. Other than the `print` statements the logic is exactly the same as the regular version, just written differently.

```
function kadane(A)
    n = length(A)
    maxoverall = A[0]
    maxendingati = A[0]
    for i = 1 to n-1 inclusive:
        if maxendingati > 0:
            print "A"
            maxendingati = maxendingati + A[i]
        else:
            print "B"
            maxendingati = A[i]
        end if
        if maxendingati > maxoverall:
            print "Y"
            maxoverall = maxendingati
        else:
            print "Z"
        end if
    end for
end function
```

(a) Suppose we call: [7 pts]

`kadane([3,4,-1,-10,-2,5,6,-1])`

What will the output string be?

(b) Suppose we call `kadane(A)` on an unknown list A of length n. and the final two letters printed are `BY`. Prove that at the instant that the `Y` is printed (and before any more code runs) that `maxoverall` $< A[n-1]$. [7 pts]

Solution:

5. The following is some experimental analysis of the average-case time requirements of Bubble Sort. Here is the pseudocode:

```
for i = 0 to n-2 inclusive:
    for j = 0 to n-i-2 inclusive:
        if A[j] > A[j+1]:
            swap A[j] and A[j+1]
        end if
    end for
end for
```

Suppose the `if` statement takes 1 second to check, the `swap` line takes 1 second, and nothing else takes any time at all.

(a) There are essentially only two lists of length 2. How much time does it take to sort each of them and what is the average time? [4 pts]

List	Time
$[1, 2]$	
$[2, 1]$	

Average Time =	

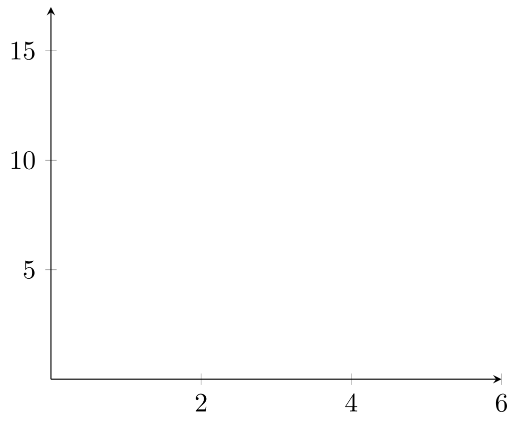
(b) There are essentially only six lists of length 3. How much time does it take to sort each of them and what is the average time? [8 pts]

List	Time
$[1, 2, 3]$	
$[1, 3, 2]$	
$[2, 1, 3]$	
$[2, 3, 1]$	
$[3, 1, 2]$	
$[3, 2, 1]$	

Average Time =	

(c) The average time for a list of length 4 is 9 seconds and for a list of length 5 is 15 seconds. Plot the graph of average time as a function of list length for lengths $2, 3, 4, 5$. [4 pts]

Solution:



6. Here is the pseudocode for Selection Sort with a `print` statement added: [15 pts]

```
for i = 0 to n-2 inclusive:
    minindex = i
    for j = i+1 to n-1 inclusive:
        if A[j] < A[minindex]:
            minindex = j
        end if
    end for
    swap A[i] with A[minindex]
    print(A)
end for
```

If we call Selection Sort on the list `[5,1,10,3,4,7]` what will be printed each time the `print` statement is encountered?

First Time						
Second Time						
Third Time						
Fourth Time						
Fifth Time						