CMSC 351 Summer 2025 Homework 1

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. This question is really review of sums from CMSC250 (or an equivalent course). We'll be doing a lot of sums so you really want to get these down! Answers should be simplified.

$7 \cdot (105-5+1)$
$= 7 \cdot 101 = 707$

Sum	**Result**	
(a) $\sum_{i=5}^{105} 7$	707	[2 pts]
(b) $\sum_{i=5}^{105} i$ $\quad \frac{101\ (105-5+1)}{2} = \frac{101 \cdot 110}{2}$	5555	[3 pts]
(c) $\sum_{i=n}^{2n} i$ with $n \geq 0$	$\frac{3}{2} n(n+1)$	[4 pts]
(d) $\sum_{i=1}^{n=117} i^2 = \frac{117 \cdot 118 \cdot 235}{6} =$	540735	[5 pts]
(e) $\sum_{i=0}^{7} [1 + 2^{3i}]$	2,396,753	[6 pts]

$(n \cdot (n+1)) \cdot$

$$\sum_{i=0}^{7} 1 + \sum_{i=0}^{7} 8^i = 8 + \frac{8^8 - 1}{8-1} = 8 + \frac{16777216 - 1}{7} = 8 + 2396745$$
$$= \underline{2396753}$$

2. A currency has only three coins: 1¢, 4¢, and 6¢.

(a) For $1 \leq n \leq 15$, in the **Greedy** = column fill in the number of coins required using the greedy algorithm. In the **Dynamic Expression** = column fill an expression of the form min{...} and in the **DE** = column the final value which demonstrates the result of the dynamic algorithm. A few have been done for you for clarification. [15 pts]

Suboptimal Spots

n	Greedy =	Dynamic Expression =	DE =
0	0	N/A	0
1	1	min{1}	1
2	2	min{2}	2
3	3	min{3}	3
4	1	min{4, 1	1
5	2	min{2, 2}	2
6	1	min{3, 3, 1}	1
7	2	min{2, 4, 2}	2
8	3	min{3, 2, 3}	2
9	4	min{3, 3, 4}	3
10	2	min{4, 2, 2}	2
11	3	min{3, 3, 3}	3
12	2	min{4, 3, 2}	2
13	3	min{3, 4, 3}	3
14	4	min{4, 3, 3}	3
15	5	min{4, 4, 4}	5

min{1+a[x-1], 1+a[x-4], 1+a[x-6]}

min{1+a[1]}

min{1+3, 1+0}

$a[\overset{2+1}{6-1}], a[\overset{2+1}{6-4}], a[\overset{0+1}{6-6}]$

a[6]+1, a[3]+1, a[1]+1

(b) Is the Greedy algorithm optimal? If your answer is "yes" explain why, and if your answer is "no" then use specific examples to explain why not. [5 pts]

Solution:

I would say that No, the Greedy Algorithm is Not Optimal because there were 3 spots (n = 8, 9, 14) where we grabbed more coins that necessary, when compared to the Dynamic Programming method meaning that the Greedy algorithm is suboptimal (in this example at least).

3. Here is a modified version of the pseudocode for the optimal solution to the Coin Changing problem:

```
A = empty list which can grow as needed
A[0] = 0
C = list of coin denominations
for x = 1 to n inclusive:
    howmanycoins = infinity
    for each coin in C with coin <= n:
        if x - coin >= 0:
            howmanycoins = min(howmanycoins,1+A[x-coin])
        end if
    end for
    A[x] = howmanycoins
end for
```

Suppose we know that the code before the outermost **for** loop takes time c_1, the body of the innermost **for** loop takes time c_2, and nothing else takes any time at all.

For each of the following, write down a simplified expression for the total time taken by the algorithm and then write down a one of our nice functions $g(n)$ so $T(n) = \Theta(g(n))$.

(a) The country issues k different coins where at least one is a 1¢ coin. [4 pts]

$T(n) = \boxed{c_1 + k \cdot c_2 \cdot n}$ which is Θ of $\boxed{n}$

(b) The country issues infinitely many coins with denominations 1¢, 3¢, 5¢, 7¢, ... [4 pts]

$T(n) = \boxed{c_1 + \sum_{x=1}^{n} \frac{n}{2} c_2}$ which is Θ of $\boxed{n^2}$

(c) The country issues infinitely many coins with denominations 1¢, 10¢, 100¢, 1000¢, ... [4 pts]

$T(n) = \boxed{c_1 + 10 \sum_{x=1}^{n} n \cdot c_2 = 10n}$ which is Θ of $\boxed{n}$

(d) Explain why the `with coin <= n` part of the code is necessary for (b) and (c) but not for (a). [4 pts]

Solution:

because in (b) and (c) the coin sets are infinite, so you need to check first if the coin is <= n, you would continue to check through the infinite set of coins and that loop would never terminate because of the infinite possibilities so you would keep running forever.

4. Prove the following using the definition of Ω. All work should be clear and explained with sentences. [15 pts]

$$n^3 - n^2 + n - 100 = \Omega(n^3)$$

Solution:

Factor n^3 out:

$$n^3 - n^2 + n - 100 = n^3\left(1 - \frac{1}{n} + \frac{1}{n^2} - \frac{100}{n^3}\right)$$

Find lower bound for large n:

@ $n \geq 10$: $\frac{1}{n} \leq 0.1$, $\frac{1}{n^2} \leq 0.01$, $\frac{100}{n^3} \leq 0.1$

for $n \geq 10$: $1 - \frac{1}{n} + \frac{1}{n^2} - \frac{100}{n^3} \geq 1 - 0.1 + 0.0 - 0.1 = 0.8$

$= 1 - \frac{1}{n} + \frac{1}{n^2} - \frac{100}{n^3} \geq 0.8$

Pick $c = 0.8$

$n_0 = 10$

$\exists n \geq 10$: $n^3 - n^2 + n - 100$

$= n^3[\ldots] \geq 0.8 \cdot n^3 = \underbrace{c \cdot n^3}$

is $\underline{\Omega(n^3)}$

5. Consider the statement:

$$n^2 + 100n = \mathcal{O}(n^2)$$

(a) This can be proved using $C = 2$. [8 pts]
What would the smallest possible corresponding n_0 be?

Smallest n_0 for $C = 2$ is	100

(b) This can be proved using $n_0 = 20$. [12 pts]
What would the smallest possible corresponding C be?

Smallest C for $n_0 = 20$ is	6

Scratch Work; Not graded:

a) $C=2$, find smallest n_0:

$n^2+100n \leq 2n^2 \iff 100n \leq n^2 \iff \underline{n \geq 100}$

b) $n^2+100n \leq C \cdot n^2$ for all $n \geq 20$

6. Prove using the definition of $\mathcal{O}$ that: [20 pts]

$$3^n \neq \mathcal{O}(2^n)$$

Note: this is not difficult but does require that you properly negate the definition and then prove that this negation is true. : Contradiction

Solution:

Assume that 3^n does equal $O(2^n)$

$\hookrightarrow$ $3^n = O(2^n)$

So, then constant $C > 0$ such that

$3^n \leq C \cdot 2^n$ for all $n >$ some constant n_0

$$\frac{3^n}{2^n} \leq \frac{C \cdot 2^n}{2^n} = \left(\frac{3}{2}\right)^n \leq C$$

but the expression $\left(\frac{3}{2}\right)^n$ continues to grow beyond any constant C.

7. It is a common mistake that if $f(n) \neq \Omega(g(n))$ then $f(n) = \mathcal{O}(g(n))$ and if $f(n) \neq \mathcal{O}(g(n))$ then $f(n) = \Omega(g(n))$ but these are false. [9 pts]

Draw an example specifically for $g(n) = 1$. In other words draw a function $f(n)$ such that $f(n) \neq \Omega(1)$ and $f(n) \neq \mathcal{O}(1)$.

Note that your function needs to be inclusive enough to clearly communicate its behavior.

Solution:

: $f(x) = x \cdot (-1)^x$

$f(n) \neq \Omega(1)$: for odd numbers @ x,
$f(x) = x \cdot (-1)^x = \underline{-x}$ which approaches $-\infty$
and alternates between $+\infty$ and $-\infty$
so cannot stay above any certain value so the lower bound is invalid.

$f(n) \neq O(1)$: -

Again, because of the (-1)^x, the function alternates between positive and negative values and is unbounded. The absolute value of the function approaches infinity along with its input (x), so there is no limit/ceiling that you can put on the values in the function, therefore the upper bound is invalid.