CS 170 Homework 1

Due **Tuesday 9/5/2023, at 10:00 pm (grace period until 11:59pm)**

1 Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write "none".

2 Course Policies

(a) What dates and times are the exams for CS170 this semester? Are there planned alternate exams?

Note: We will make accommodations for students in faraway timezones.

(b) Homework is due Mondays at 10:00pm, with a late deadline at 11:59pm. At what time do we recommend you have your homework finished?

(c) We provide 2 homework drops for cases of emergency or technical issues that may arise due to homework submission. If you miss the Gradescope late deadline (even by a few minutes) and need to submit the homework, what should you do?

(d) What is the primary source of communication for CS170 to reach students? We will send out all important deadlines through this medium, and you are responsible for checking your emails and reading each announcement fully.

(e) Please read all of the following:

(i) **Syllabus and Policies:** `https://cs170.org/syllabus/`

(ii) **Homework Guidelines:** `https://cs170.org/resources/homework-guidelines/`

(iii) **Regrade Etiquette:** `https://cs170.org/resources/regrade-etiquette/`

(iv) **Forum Etiquette:** `https://cs170.org/resources/forum-etiquette/`

Once you have read them, copy and sign the following sentence on your homework submission.

"I have read and understood the course syllabus and policies."

3 Understanding Academic Dishonesty

Before you answer any of the following questions, make sure you have read over the syllabus and course policies (`https://cs170.org/syllabus/`) carefully. For each statement below, write *OK* if it is allowed by the course policies and *Not OK* otherwise.

(a) You ask a friend who took CS 170 previously for their homework solutions, some of which overlap with this semester's problem sets. You look at their solutions, then later write them down in your own words.

(b) You had 5 midterms on the same day and are behind on your homework. You decide to ask your classmate, who's already done the homework, for help. They tell you how to do the first three problems.

(c) You look up a homework problem online and find the exact solution. You then write it in your words and cite the source.

(d) You were looking up Dijkstra's on the internet, and inadvertently run into a website with a problem very similar to one on your homework. You read it, including the solution, and then you close the website, write up your solution, and cite the website URL in your homework writeup.

4 Math Potpourri

The following subparts will cover several math identities, tricks, and techniques that will be useful throughout the rest of this course.

(a) Simplify the following expressions into a single logarithm (i.e. in the form $\log_a b$):

(i) $\frac{\ln x}{\ln y}$

(ii) $\ln x + \ln y$

(iii) $\ln x - \ln y$

(iv) $170 \ln x$

(b) Give a simple proof for each of the following identities:

(a) $x^{\log_{1/x} y} = \frac{1}{y}$

(b) $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$

(c) $\sum_{k=0}^{n} ar^k = \begin{cases} a\left(\frac{1-r^{n+1}}{1-r}\right), & r \neq 1 \\ a(n+1), & r = 1 \end{cases}$

5 Recurrence Relations

For each part, find the asymptotic order of growth of T; that is, find a function g such that $T(n) = \Theta(g(n))$. Show your reasoning and do not directly apply any master theorems.
In all subparts, you may ignore any issues arising from whether a number is an integer.

(a) $T(n) = 3T(n/4) + 10n$

(b) $T(n) = 97T(n/100) + \Theta(n)$

(c) An algorithm $\mathcal{A}$ takes $\Theta(n^2)$ time to partition the input into 5 sub-problems of size $n/5$ each and then recursively runs itself on 3 of those subproblems. Describe the recurrence relation for the run-time $T(n)$ of $\mathcal{A}$ and find its asymptotic order of growth.

(d) $T(n) = 3T(n/3) + \Theta(n)$

(e) $T(n) = T(3n/5) + T(4n/5)$ (We have $T(1) = 1$)
Hint: Try to guess a $T(n)$ of the form an^b and then use induction to argue that it is correct.

6 In Between Functions

In this problem, we will find a function $f(n)$ that is asymptotically worse than polynomial time but still better than exponential time. In other words, f has to satisfy two things,

- For all constants $k > 0$, $f(n) = \Omega(n^k)$ (1)
- For all constants $c > 0$, $f(n) = O(2^{cn})$ (2)

(a) Try setting $f(n)$ to a polynomial of degree d, where d is a very large constant. So $f(n) = a_0 + a_1 n + a_2 n^2 \cdots + a_d n^d$. For which values of k (if any) does f fail to satisfy (1)?

(b) Now try setting $f(n)$ to a^n, for some constant a that's as small as possible while still satisfying (1) (e.g. 1.000001). For which values of c (if any) does f fail to satisfy (2)?
Hint: Try rewriting a^n as 2^{bn} first, where b is a constant dependent on a.

So far we have found that the functions which look like $O(n^d)$ for constant d are too small and the functions that look like $O(a^n)$ are too large even if a is a tiny constant.

(c) Find a function $D(n)$ such that setting $f(n) = O(n^{D(n)})$ satisfies both (1) and (2). Give a proof that your answer satisfies both.
Hint: Make sure $D(n)$ is asymptotically smaller than n.

7 [Coding] Decimal to Binary

Given the n-digit decimal representation of a number, converting it into binary in the natural way takes $O(n^2)$ steps.

(a) Give a divide-and-conquer algorithm to do the conversion in $O(n^{\log_2 3} \log n)$ time.

Hint: refer to lecture slides and take inspiration from Karatsuba's algorithm!

(b) Write out the recurrence for your algorithm from part (a) and use it to derive the desired runtime.

(c) Code your solution! Go to `https://github.com/Berkeley-CS170/cs170-fa23-coding` and follow the directions in the `README.md` to complete this week's coding homework in `hw01`.

Notes:

- *Submission Instructions:* Please download your completed `decimal_to_binary.ipynb` file and submit it to the gradescope assignment titled "Homework 1 Coding Portion".

- *OH/HWP Instructions:* While we will be providing conceptual help on the coding portion of homeworks, OH staff will not look at your code and/or help you debug.
- *Academic Honesty Guideline:* We realize that code for some of the algorithms we ask you to implement may be readily available online, but we strongly encourage you to not directly copy code from these sources. Instead, try to refer to the resources mentioned in the notebook and come up with code yourself. That being said, we **do acknowledge** that there may not be many different ways to code up particular algorithms and that your solution may be similar to other solutions available online.