CSE 21 Spring 2022
Homework 5
Due date: Thurs, May 12, 2022 at 11:59pm

Instructions

This homework should be done in **groups** of one to four students, without assistance from anyone besides the instructional staff and your group members. Homework must be submitted through Gradescope by a **single representative** of your group and received by **11:59pm** on the due date. There are no exceptions to this rule.

You will be able to look at your scanned work before submitting it. You must **type** your solutions. (hand-drawn diagrams are okay.) Your group representative can resubmit your assignment as many times as you like before the deadline. Only the most recent submission will be graded.

Students should consult their textbook, class notes, lecture slides, podcasts, group members, instructors, TAs, and tutors when they need help with homework. You may ask questions about the homework in office hours, but questions on Piazza should be private, visible only to instructors.
This assignment will be graded for not only the *correctness* of your answers, but on your ability to present your ideas clearly and logically. You should explain or justify, present clearly how you arrived at your conclusions and justify the correctness of your answers with mathematically sound reasoning (unless explicitly told not to). Whether you use formal proof techniques or write a more informal argument for why something is true, your answers should always be well-supported. Your goal should be to **convince the reader** that your results and methods are sound.
Key Concepts recursive algorithm: correctness and runtime, Recurrence relations, Master Theorem, unraveling.

---

1. (8 points) Consider the following recurrence relation:

$$A(1) = 1, \quad A(n) = 2A(n-1) + 2^{n-1}$$

Use the method of unraveling to find a closed form for $A(n)$.

2. (12 points) **PowerOfTwo** Given an integer $x \geq 0$, this algorithm returns 2^x.

```
   procedure $FastPower$ ($x$ : positive integer with $x \geq 0$)
1.     if $x == 0$:
2.         return 1
3.     if $x$ is odd:
4.         return $2 * FastPower(x-1)$
5.     if $x$ is even:
6.         return $FastPower(x/2)^2$
```

(a.) Prove *FastPower* correctly returns 2^x for any input $x \geq 0$.

(b.) Assuming that $x = 2^k$ for some integer $k \geq 0$. In terms of k, how many recursive calls are necessary to reduce x down to 0 (base case)?

(c.) Assuming that $x = 2^k - 1$ for some integer $k \geq 0$. In terms of k, how many recursive calls are necessary to reduce x down to 0 (base case)?

3. (20 points) Suppose you are given a pile of n coins where n is a power of 2. A coin is authentic with probability p and counterfeit with probability $1-p$.

An authentic coin weighs exactly 1 gram. A counterfeit coin weighs slightly less than an authentic coin but you don't know exactly by how much and furthermore, two counterfeit coins may have different weights.

You wish to throw away all of the counterfeit coins and keep all of the authentic coins.

You have access to a digital scale that can use to weigh as many coins as you want.

Consider the following recursive algorithm to do this:

```
Counterfeit(pile of $n$ coins) such that $n$ is a power of 2.
1) weigh the entire pile of $n$ coins.
2) if the pile weighs exactly $n$ grams, then:
3)     Keep all of the coins and end the process.
4) else:
5)     if there is only one coin in the pile:
6)         throw it away.
7)     else:
8)         Split the pile of coins into two piles each with $n/2$ coins.
9)         Recursively run Counterfeit on each pile.
```

(a) Prove using induction that this algorithm will keep all authentic coins and throw away all counterfeit coins.

(b) What is the minimum number of times you use the digital scale?
Describe a scenario when you achieve the minimum.
What is the probability this particular scenario will happen?

(c) What is the maximum number of times you use the digital scale?
Describe a scenario when you achieve the maximum.
What is the probability this particular scenario will happen?

(d) Let the random variable S_n be the number of times you use the scale for a pile of n coins. Explain why the expected value of S_n satisfies the recurrence relation:

$$E[S_1] = 1, \qquad E[S_n] \leq 1 + (1 - p^n)(2E[S_{n/2}])$$

i. Let C be the event that there is at least one counterfeit coin in the pile, then

$$E[S_n] = P\left(\overline{C}\right) E\left[S_n|\overline{C}\right] + P(C)E[S_n|C]$$

Compute $P\left(\overline{C}\right)$,$P(C)$ and $E\left[S_n|\overline{C}\right]$

ii. Make an argument why $E[S_n|C] \geq 1 + 2E[S_{n/2}]$ when n is a power of 2 greater than 1.

iii. Put it all together to get the recurrence above.

(e) If you had 16 coins and $p = 8/9$, what does the recurrence relation tell you about the expected number of times you would use the digital scale?

4. **Master Theorem.** (12 points) For each situation below, first give a recurrence for the running time of the algorithm; you don't need to explain this part of your answer. Then use the Master Theorem, if possible, to find this time up to order, clearly explaining how the Master Theorem applies (e.g., giving values of variables and identifying the appropriate case).

If it is not possible to use the master theorem then solve the recurrence in another way.

a. Suppose an algorithm solves a problem of size n by recursively calling 6 subproblems, each subproblem is of 1/3 the size of the original input. Then the non-recursive part of the algorithm takes $O(n^2)$ time.

b. Suppose an algorithm solves a problem of size n by recursively calling 8 subproblems, each of subproblem is of 1/4 the size of the original input. Then the non-recursive part of the algorithm takes $O(n\sqrt{n})$ time.

c. Suppose an algorithm solves a problem of size n by recursively calling 3 subproblems, each of subproblem is of 1/4 the size of the original input. Then the non-recursive part of the algorithm takes $O(n^{1/2})$ time.