Collaboration Policy: You are encouraged to collaborate with up to 4 other students, but all work submitted must be your own *independently* written solution. List the computing ids of all of your collaborators in the `collabs` command at the top of the tex file. Do not share written notes, documents (including Google docs, Overleaf docs, discussion notes, PDFs), or code. Do not seek published or online solutions for any assignments. If you use any published or online resources (which may not include solutions) when completing this assignment, be sure to cite them. Do not submit a solution that you are unable to explain orally to a member of the course staff. Any solutions that share similar text/code will be considered in breach of this policy. Please refer to the syllabus for a complete description of the collaboration policy.

Collaborators: list your collaborators

Sources: list your sources

problem 1 *Solving Recurrences*

Prove a (as tight as possible) O (big-Oh) asymptotic bound on the following recurrences. You may use any base cases you'd like.

1. For the following two recurrences, it may be helpful to draw out the tree. However, you should prove the asymptotic bound using induction.
   - $T(n) = T(\frac{n}{2}) + T(\frac{n}{4}) + T(\frac{n}{8}) + n$

     Solution:
   - $T(n) = 2T(\frac{n}{3}) + T(\frac{n}{6}) + n$

     Solution:
2. For the following recurrence relations, indicate: (i) which case of the Master Theorem applies (if any); (ii) justification for why that case applies (if one does) i.e., what is a, $f(n)$, ε, etc; (iii) the asymptotic growth of the recurrence (if any case applies).
   - $T(n) = 7T(\frac{n}{5}) + n \log n$

     Solution:
   - $T(n) = 3T(\frac{n}{3}) + n \log n$

     Solution:

PROBLEM 2 *Climate History*

Scientists call *paleoclimatologists* study the history of climate on earth before instruments were invented to measure temperatures, precipitation, etc. They try to reconstruct climate history using data found from analyzing rocks, sediments, tree rings, fossils, ice sheets, etc.

A group is trying to better understand periods of low precipitation (e.g. dryness) for a time period that spans many thousands of years. Using various data sources, they have assigned a value d_i for the relative dryness for each time-unit (let's say it's measured for every century) for this very long time period. For each time-unit i, one of five values has been assigned as d_i:

- -3 for very high precipitation
- -1 for high precipitation
- 0 for normal (or unknown)
- 1 for low precipitation
- 3 for very low precipitation

The scientists want to find the score for the driest period of consecutive years in their data. Because the effects of a drought are cumulative, the score for a period starting at time i and ending at time j will be the value when all scores from i and j are added. Consider this example with $n = 16$:

d_i	0	3	3	0	-3	-3	1	0	3	-1	-1	0	1	1	3	0
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

The period [1:2] looks high with a score of 6. But the highest score is 7 for this period [6:14]. Note there are some negative values in the middle of this period, but the high values around those make it worth having a period that includes those negative values. (The period [6:15] also has score 7. There can be more than one period with the largest score, but you just need to return the highest score, so the existence of multiple periods with the same best score doesn't matter.)

There are many ways to solve this problem, including a $\Theta(n^2)$ brute-force approach that calculates scores for all combinations of starting and ending values for a period and keeps track of the largest score. You need to do better than that. (Though if you were ever to code your solution, coding the brute-force approach is a good way to test your algorithm. Coding is **not** required for this problem.)

What you need to do: Describe an algorithm (clearly in words or in pseudocode) that uses a divide and conquer algorithm that solves this problem in $\Theta(n \log n)$ time. Your description should make it clear what the base case is, what work is done in dividing and in combining, and what recursive subproblems are solved. Along with your algorithm's description, give a brief but convincing argument that the time complexity is $\Theta(n \log n)$.

The inputs to your algorithm will be n, the number of samples, and a list d that stores n values, where each is either -3, -1, 0, 1 or 3. Your algorithm will output the largest dryness score for a period in the data (as described above).

Solution:

PROBLEM 3 *Fast Transformations*

Computer graphics software typically represents points in n dimensions as $(n+1)$-dimensional vectors. To make transformations on the points (e.g., rotating a modelled figure, zooming in, or making the figure appear as though seen through a fish-eye lens), we use a $(n+1) \times (n+1)$ matrix T which defines the transformation, and then we multiply each vector by this matrix to transform that point. That is, for vector v, the transformed vector is $v' = T \times v$.

Let's say we are developing software for very high dimension graphics (n dimensions), and we have a transformation T that we would like to apply to a particular point n times. Develop an algorithm which can multiply this $(n+1)$-dimensional point by T (the $(n+1) \times (n+1)$ transformation matrix) n times in $o(n^3)$ (little-oh of n^3) time. Prove this run time.

Solution:

PROBLEM 4 *Receding Airlines*

You have been hired to plan the flights for Professor Floryan's brand new passenger air company, "Receding Airlines." Your objective is to provide service to n major cities within North America. The catch is that this airline will only fly you East.

You recognize that in order to enable all your passengers to travel from any city to any other city (to the East) with a single flight requires $\Omega(n^2)$ different routes. Prof. Floryan says that the airline cannot be profitable when supporting so many routes. Another option would be to order the cities in a list (from West to East), and have flights that go from the city at index i, to the city at index $i+1$. This requires $\Theta(n)$ routes, but would mean that some passengers would require $\Omega(n)$ connections to get to their destination.

1. Devise a compromise set of routes which requires no passenger have more than a single connection (i.e. must take at most two flights), and requires no more than $O(n \log n)$ routes. Prove that your set of routes satisfies these requirements.

2. After a few years, passengers start demanding routes from East to West, and you decide to support new routes from East to West (in addition to supporting routes from West to East). Show that with routes in *both* directions, it is possible to connect all n cities with just $O(n)$ routes such that no passenger needs more than a single connection to get to their destination.

Note: In class so far, we've used recurrence relations to count an algorithm's time complexity, i.e., how many basic operations are executed. Here we're using one to count the number of flights. While this is a bit different than measuring an algorithm's time-complexity, we can still use a divide and conquer approach and a recurrence relation to answer these questions.

Solution:

PROBLEM 5 *Bazinga!*

Theoretical Physicist Sheldon Cooper has decided to give up on String Theory in favor of researching Dark Matter. Unfortunately, his grant-funded position at Caltech is dependent on his continued work in String Theory, so he must search elsewhere. He applies and receives offers from MIT and Harvard. While money is no object to Sheldon, he wants to ensure he's paid fairly and that his offers are at least the median salary among the two schools' Physics departments. Therefore, he hires you to find the median salary across the two departments. Each school mantains a database of all of the salaries for that particular school, but there is no central database.

Each school has given you the ability to access their particular data by executing *queries*. For each query, you provide a particular database with a value k such that $1 \leq k \leq n$, and the database returns to you the k^{th} smallest salary in that school's Physics department.

You may assume that: each school has exactly n physicists (i.e. $2n$ total physicists across both schools), every salary is unique (i.e. no two physicists, regardless of school, have the same salary), and we define the *median* as the n^{th} highest salary across both schools.

1. Design an algorithm that finds the median salary across both schools in $\Theta(\log(n))$ total queries.
2. State the complete recurrence for your algorithm. You may put your $f(n)$ in big-theta notation. Show that the solution for your recurrence is $\Theta(\log(n))$.
3. Prove that your algorithm above finds the correct answer. *Hint: Do induction on the size of the input.*

Solution:

PROBLEM 6 *Mission Impossible*

As the newly-appointed Secretary of the Impossible Missions Force (IMF), you have been tasked with identifying the double agents that have infiltrated your ranks. There are currently n agents in your organization, and luckily, you know that the majority of them (i.e., strictly more than $n/2$ agents) are loyal to the IMF. All of your agents know who is loyal and who is a double agent. Your plan for identifying the double agents is to pair them up and ask each agent to identify whether the other is a double agent or not. Agents loyal to your organization will always answer honestly while double agents can answer arbitrarily. The list of potential responses are listed below:

Agent 001	Agent 002	Implication
"002 is a double agent"	"001 is a double agent"	At least one is a double agent
"002 is a double agent"	"001 is loyal"	At least one is a double agent
"002 is loyal"	"001 is a double agent"	At least one is a double agent
"002 is loyal"	"001 is loyal"	Both are loyal or both are double agents

1. A group of n agents is "acceptable" for a mission if a majority of them ($> n/2$) are loyal. Suppose we have an "acceptable" group of n agents. Describe an algorithm that has the following properties:
   - Uses at most $\lfloor n/2 \rfloor$ pairwise tests between agents.
   - Outputs a smaller "acceptable" group of agents of size at most $\lceil n/2 \rceil$.
2. Using your approach from Part 1, devise an algorithm that identifies which agents are loyal and which are double agents using $\Theta(n)$ pairwise tests.

Solution: