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 *Tug-of-War Fraud*

Your country will be competing in the upcoming Tug-of-War World Cup. You are part of the coaching staff and have carefully screened all perspective athletes and divided them in to several groups based on the amount of force they exert when pulling the rope. Each group consists of athletes who can individually pull with exactly the same amount of force. All athletes also wear weighted vests to insure that they all weigh exactly the same amount when competing. So choosing any number (even-number) of athletes from a single group, dividing them in to two equal sized teams, and having them complete against each other in a tug-of-war contest will always result in a draw (neither side wins).

One day you notice that one of the groups is missing a member and one of the other groups has an extra member. The group missing a member, group A, has a lower individual pulling force than the group that has an extra member, group B. So you know now that everyone in Group B has the same pulling force, except for one athlete (from Group A), who has a lower pulling force. The athlete from Group A is a fraud and must be removed.

You decide to find the fraud by holding tug-of-war contests within Group B. If the fraudulent athlete is in a contest with teams of equal size, their team will lose. There are now n athletes in Group B, including the fraudster. You could just hold $n/2$ one-on-one contests to find the fraudster, but that's a lot of contests to hold and you think you could do better. Create a strategy that consists of holding contests with subgroups of athletes from Groups B that will eventually be able to identify the fraudster.

An algorithm that involves holding $n/2$ one-on-one contests has a time complexity of $\Theta(n)$. Your algorithm must have a lower time complexity than that. (Note: There is an algorithm that would require only 4 contests when $n = 80$. You don't need to find that specific algorithm. You're OK as long as yours is better than $\Theta(n)$)

Solution:

problem 2 *Sorry-Oh*

You are playing a new video game about a character named "Sorry-Oh". Sorry-Oh is currently at a point in the game where he needs to find a way to get from one corner of a rectangular room to the opposite corner. Unfortunately, some cells contain lava. When Sorry-Oh steps into a cell containing lava, he says "Sorry" and takes a certain amount of extra damage. The room can be visualized as a rectangular grid of cells whose size is x by y. For each move, Sorry-Oh can only step to a cell that is either horizontally or vertically adjacent to his current cell. To get from the starting corner to the opposite corner, Sorry-Oh is only allowed to make the minimum number of moves, a total of $x + y - 2$ moves.

Sorry-Oh starts in the top-left corner cell with a damage value of 0. Each cell contains a number representing a certain amount of additional damage specific to that cell. Cells that do not contain lava, have a value of 0. Cells that do contain lava have a positive integer value. When moving from one cell to the next, any existing damage that Sorry-Oh already had will first be doubled (previous lava wounds continue to hurt for a long time), then the damage value of the new cell will be added to the total.

Your goal is to get Sorry-Oh from the starting cell on the top-left (this will always be indicated by "*S*", to the ending cell on the bottom-right, and to have the lowest possible damage value after arriving at that ending cell.

Given the following example grid -

S	2	4
1	3	0
2	2	1

An example path could be computed as follows:

1. Start cell, damage = 0
2. move right, d = 2 ((0 * 2) + 2))
3. move right, d = 8 ((2 * 2) + 4))
4. move down, d = 16 ((8 * 2) + 0))
5. move down, d = 33 ((16 * 2) + 1))
6. Total damage from the moves R-R-D-D is 33.

Another example path:

1. move down, d = 1 ((1 * 2) + 1))
2. move right, d = 5 ((1 * 2) + 3))
3. move down, d = 12 ((5 * 2) + 2))
4. move right, d = 25 ((12 * 2) + 1))
5. Total damage from the moves D-R-D-R is 25.

The grid is given to you in the form of an unweighted, undirected graph. Each node represents a cell and contains a value representing the additional damage associated with that cell (except the starting cell, which only contains an "S"). The edges of the graph represent connections to the adjacent cells (each node will have 2, 3, or 4 edges).

Describe an algorithm that can find a path through the grid that has the lowest possible damage value after arriving at the final cell. Explain the time complexity of your algorithm.

Solution:

PROBLEM 3 *Ancient Population Study*

Historians are studying the population of the ancient civilization of *Algorithmica*. Unfortunately, they have only uncovered incomplete information about the people who lived there during Algorithmica's most important century. While they do not have the exact year of birth or year of death for these people, they have a large number of possible facts from ancient records that say when a person lived relative to when another person lived.

These possible facts fall into two forms:

- The first states that one person died before the another person was born.
- The second states that their life spans overlapped, at least partially.

The Algorithmica historians need your help to answer the following questions. First, is the large collection of uncovered possible facts internally consistent? This means that a set of people could have lived with birth and death years that are consistent with all the possible facts they've uncovered. (The ancient records *may not be accurate*, meaning all the facts taken together cannot possibly be true.) Second, if the facts are consistent, find a sequence of birth and death years for all the people in the set such that all the facts simultaneously hold. (Examples are given below.)

We'll denote the n people as $P_1, P_2, \ldots, P_n$. For each person P_i, their birth-year will be b_i and their death-year will be d_i. (Again, for this problem we do not know and cannot find the exact numeric year value for these.)

The possible facts (input) for this problem will be a list of relationships between two people, in one of two forms:

- P_i *prec* P_j (indicates P_i died before P_j was born)
- P_i *overlaps* P_j (indicates their life spans overlapped)

If this list of possible facts is not consistent, your algorithm will return "not consistent". Otherwise, it will return a possible sequence of birth and death years that is consistent with these facts.

Here are some examples:

- The following facts about $n = 3$ people are **not** consistent: P_1 *prec* P_2, P_2 *prec* P_3, and P_3 *prec* P_1.
- The following facts about $n = 3$ people **are** consistent: P_1 *prec* P_2 and P_2 *overlaps* P_3. Here are two possible sequences of birth and death years:
  $b_1, d_1, b_2, b_3, d_2, d_3$
  $b_1, d_1, b_3, b_2, d_2, d_3$
  (Your solution only needs to find one of any of the possible sequences.)

Your answer should include the following. Clearly and precisely explain the graph you'll create to solve this problem, including what the nodes and edges will be in the graph. Explain how you'll use one or more of the algorithms we've studied to solve this graph problem, and explain why this leads to a correct answer. Finally, give the time-complexity of your solution.

Solution: