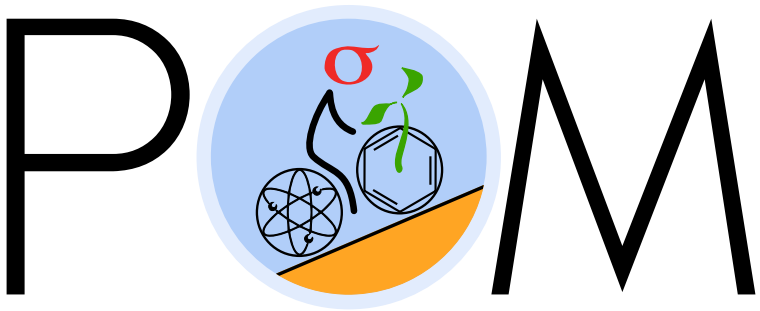


SigmaCamp's *Problem of the Month* Contest

DECEMBER 2025

Video Submissions

Starting this year, **all solutions must be accompanied by short videos explaining your work.**

- Videos must be at most **2.5 minutes long** for 5pt/10pt problems, and at most **6 minutes long** for projects.
- **Solutions must be narrated**, but you do not need to show your face. Acceptable formats include:
  - Screensharing slides or a drawing app (e.g., MS Paint) with narration.
  - Recording a whiteboard, paper, or easel with narration (contents may be pre-written).
  - Speaking directly to the camera.
- **Submit videos as links** (Google Drive, Youtube, Dropbox, etc.). Extra requested files may be submitted as a single PDF file per problem.
- For coding problems, submit your code along with a video explaining your submission. Only **Python-3** (`.py`) and **Java** (`.java`) submissions are accepted.

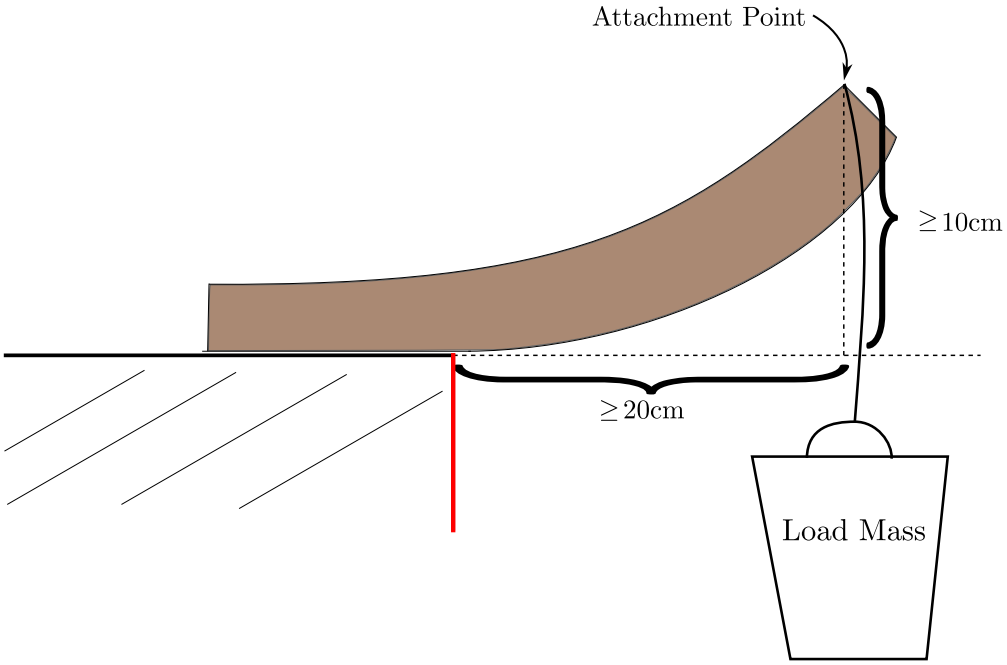
Please see sigmacamp.org/pom for full details on the 2025 POM format change.

Project (30 points)

Physics and Applied Science

Your task for this project is to build a cantilever (see this video for a nice introduction) out of cardboard and adhesive. A cantilever is essentially half of a bridge — while a bridge is supported at both ends, a cantilever is only supported from one side.

Unlike typical cantilevers, we add some additional constraints — the cantilever you build must be anchored to a flat surface (which we refer to as a "table" for simplicity, but you may use any flat surface that ends in a ledge), and must support a mass at least some height above this surface (see the technical specifications below).



Above is a sketch of a valid cantilever submission. **No part of the cantilever may dip below the level of the tabletop (represented by the horizontal dashed line) at any point**, not even to aid in attachment or stability. In particular, this implies no part of the side of the table (represented by the red line in the figure) may be touched by your cantilever. The point where the mass is attached to the cantilever must be at least 20 cm horizontally away from the table and at least 10 cm above the table height.

Provide a theoretical motivation for your design. Include a quantitative prediction (in grams) of the maximum weight your cantilever will be able to support. For any equations you use, you should state the assumptions that the equation is based on and why these assumptions apply to your system.

When making your theoretical prediction, you may conduct additional experiments to determine the properties of your materials and/or design. However, your theoretical prediction should not be based on the results of stress testing your structure to failure.

Technical Specifications

i) The mass should be attached at a point that is at least 20 cm away from the edge of the table.

ii) The mass should be attached at a point that is at least 10 cm above the table surface.

iii) The cantilever should support a mass of at least 100 g. More points will be awarded for larger weight.

iv) The cantilever should have a largest diameter of around 6 cm, or you should be able to wrap a 20 cm long strip of paper around the thickest part of the cantilever.

v) The cantilever should be made entirely out of cardboard and glue/tape of your choice. Absolutely no other materials are allowed. The mass attachment may use other materials, for example twine.

vi) Cantilever should support mass without bending below the table.

Video Submission Details

Provide video confirmation of each of the following:

- Your cantilever supporting the minimal weight (100 g).
- Your construction supporting the largest weight possible before it breaks.

For the items below, you may (if you cannot edit video or for similar technical reasons) include still photos with your written submission instead of video.

- Overall structure of your cantilever.
- How the cantilever was attached to the table.
- Verification that the cantilever fulfills the technical specifications for length (i) and height (ii).
- Verification that the cantilever fulfills the technical specifications for thickness (iv).

Written Submission Details

- Clearly state the theoretical considerations you made when making your design. State and justify your prediction for the maximum mass the cantilever could support.
- You should provide a clear description of your design and provide details on how you constructed your cantilever.
- Provide a list of materials used in the process.
- Describe any additional experiments you may have performed in making your theoretical prediction.
- You should clearly communicate the mass of your weights in units of grams. You may use any objects of your choice as weights, as long as you determine and provide their masses (you can use scales or objects of known weight, such as coins or food items or known volume of water).
- Include photos for anything not clearly documented in your video.

Submission Checklist

Make sure your submission answers the following questions:

- Theoretical discussion — Did you clearly state the assumptions and reasoning in creating your design? Did you make a quantitative prediction of how much mass your structure can support?
- Experimental setup — Did your construction fulfill the technical requirements? Did you design your experiment to minimize uncertainty of measurement using the equipment available?
- Analysis and Discussion — Did your experiment agree with your theoretical prediction? Why or why not? How could you improve your design?
- Documentation — Have you documented all necessary details and requirements in either video or photos?

Hint:

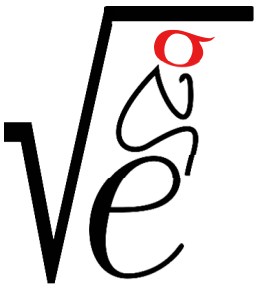
No hint for this month.

Solution:

Featured Solution
by Maxim Lavrov
https://www.youtube.com/watch?v=LfFt-jRZJh8

Featured Solution
by Maria Merzlyakova
https://www.youtube.com/watch?v=tArTdWrD3UM
Highlighted Solution PDF

Mathematics



For all mathematics problems, please provide full justification. **Do not include any code** in your submission unless specified otherwise — all code submissions will be awarded no points.

5 points:

Alice decides to travel, and leaves her home town by getting on the longest flight to another town. She then repeats this process, always taking the longest possible flight to another town, for a total of 177 flights. All the flights between different towns are of different lengths, and the flights' schedule and availability do not change. Also, if there is a flight from city A to city B, then the return flight from B to A exists and is of the same length. Is it possible for her to be home after 177 flights?

Hint:

What happens if we replace 177 with 3?

Solution:

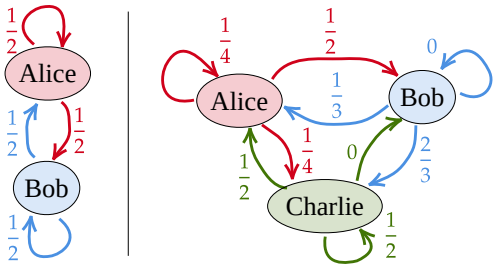
Each time Alice flies, she has the choice of flying back the way she came. Therefore, each flight she takes must be longer or of the same length than the one before it (and the same length only happens if she goes back on the same flight).

Now assume that Alice returns home, and compare her first flight out to her last flight back. The first flight Alice took was the longest available to her, so it must be longer (or the same) as the last flight Alice took. Therefore, all the flights Alice took must be of the same length. Since the flights between different towns are of different lengths, this means that Alice is always taking the same flight, back and forth, coming back after every even number of flights, but being away after any odd number of flights. In particular, she cannot be home after 177 flights.

10 points:

(a) [2 points] Alice and Bob have decided to play a warm-up game, called Warm Tomato, which is similar to hot potato. Every second, if you have the tomato, you can either wait or throw it to another player. Alice and Bob both always throw the tomato to the other player with probability $\frac{1}{2}$ or keep it with probability $\frac{1}{2}$, reevaluating in the next second (these probabilities are shown in the diagram to the left). Alice starts with the tomato at time 0. Find the expected value (or average value) of the time when Alice next has the tomato.

A helpful tool for solving these kinds of problems is to use a to indicate the expected time until Alice next gets the tomato. Similarly, b can indicate the expected time until Alice next gets the tomato, assuming that Bob starts with the tomato. Now consider a move when the tomato ends up in Alice's hands. Think about all possible ways this could happen, and use this to write an equation (or equations) expressing a as some function of a and b. Then you can solve this equation (or a system of equations). If done correctly, the answer should be 2 seconds.



(b) [8 points] Now that they're all warmed up, Alice, Bob, and Charlie are playing Hot Tomato. Every second, if you have the tomato, you can either wait or throw it to another player. The probabilities of each player making a particular decision are shown in the diagram to the right. For example, each time Alice finds herself holding the tomato, she has a $\frac{1}{4}$ chance of keeping the tomato (and reevaluating in one second), a $\frac{1}{2}$ chance of passing it to Bob, and a $\frac{1}{4}$ chance of passing it to Charlie. Alice starts with the tomato at time 0. Find the expected value (or average value) of the time when Alice next has the tomato.

Hint:

No hint this month.

Solution:

Let us call the expected value of time until Alice next gets the tomato a. Let b and c be the expected amounts of time until Alice next gets the tomato, assuming that Bob, or respectively, Charlie, is currently holding the tomato. Let's consider what happens if Alice is holding the tomato. In one second, she can either keep it (with given probability), in which case the time until she next gets it is 1 second, or she can throw it to Bob or Charlie, in which case she is expected to have it back after $b+1$ or $c+1$ seconds, respectively. This allows us to write a in terms of b and c.

(a) In the game of Warm Tomato, we thus get the equations $a = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot (b+1)$. For Bob we get the equation $b = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot (b+1)$. We find that $b = 2$ and $a = 2$, so the answer is 2 seconds.

(b) In the game of Hot Tomato for Alice we get the equation

$$a = \frac{1}{4} \cdot 1 + \frac{1}{2} \cdot (b+1) + \frac{1}{4} \cdot (c+1).$$

For Bob and Charlie, we get the equations

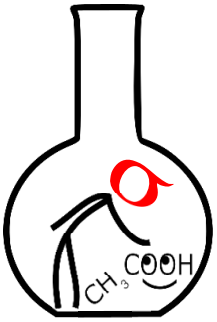
$$b = \frac{1}{3} \cdot 1 + \frac{2}{3}(c+1)$$

and

$$c = \frac{1}{2} \cdot 1 + \frac{1}{2}(c+1).$$

Solving this system of equations, we find that $c = 2$, $b = \frac{7}{3}$, and $a = \frac{8}{3}$. Thus, the answer is $\frac{8}{3}$ seconds.

Chemistry



Acids are compounds that can dissociate to form a hydrogen ion H^{+1}. Dissociation occurs readily when the bond between the hydrogen atom and the rest of the molecule is strongly polarized. "Polarized" means that the electron pair forming the chemical bond between hydrogen and the acidic residue is displaced toward the latter, as shown in Figure 1. In this example, the oxygen atoms—being far more electronegative than sulfur—pull electron density away from sulfur. The sulfur atom, in turn, pulls electrons from the oxygen atom in the OH group, making the O–H bond strongly polarized. The positive charge on the hydrogen increases, making it more prone to dissociation from the H_2SO_4 molecule. This is why sulfuric acid is a strong acid.

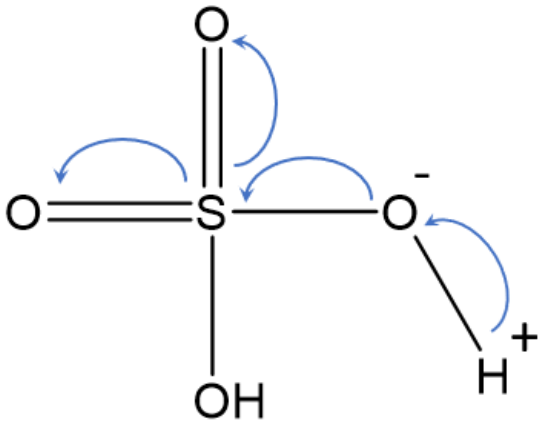


Figure 1: Polarisation of the O-H bond in sulfuric acid

When does this happen? It occurs when strongly electron-withdrawing groups are attached to the oxygen atom in the OH group. In such cases, the acid is strong; otherwise, it is weak. Consider the opposite example: ethyl alcohol (ethanol). In ethanol, the OH group is connected to an ethyl group composed of carbon (an element with moderate electronegativity) and hydrogen (an element that is even less electronegative). Instead of pulling electrons away from the oxygen atom in the OH group, these atoms *donate* electron density to it. As a result, the positive charge on the hydrogen atom in the ethanolic OH group becomes so low that it does not dissociate. This is why ethyl alcohol is a very weak acid.

Which elements are the most electronegative? Highly electronegative elements are located in the upper-right corner of the Periodic Table (you may google the electronegativity series for more details).

When is the electron-withdrawing effect most pronounced? It is strongest when the electronegative atom is connected to the atom of interest through a single bond. If they are separated by two bonds, the effect weakens; the more bonds between the electron-withdrawing atom and the atom of interest, the less significant the effect. Thus, fluorinated alcohols can be ranked by acidity as follows (from left to right):

This picture demonstrates how the ability of the fluorine atom to pull an electronic density from other atoms decreases when the number of bonds separating these two atoms increases. That demonstrates that not only

[1]This definition applies only to aqueous solutions. A more advanced theory exists, but as long as we are dealing with aqueous solutions, this classical definition works well.

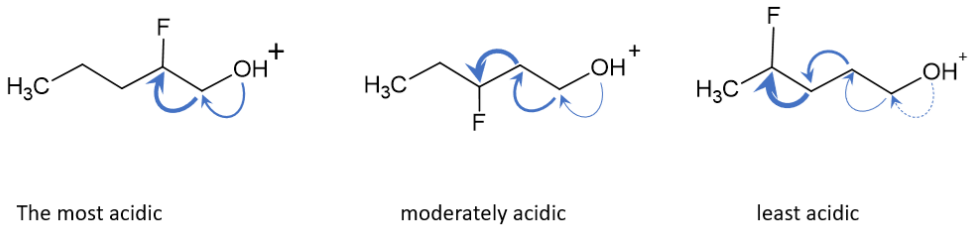


Figure 2: Fluorinated alcohols ranked by their acidity. Arrow's thickness indicates how strongly electrons are being pulled by fluorine.

electronegativity of some atoms, but also its position in the molecule is important when we consider the effect of that atom on the distribution of electrons in the molecule.

5 points:

The figure 1 shows several alcohols. Rank them according to their acidity.

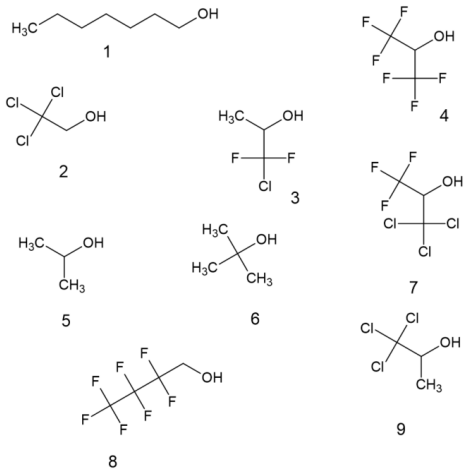


Figure 3

SS

Hint:

The atoms with the highest electronegativity must be as close to the OH group as possible.

Solution:

Answer: Starting from the most acidic: 4, 7, 8, 3, 2, 9, 1, 5, 6.

In this problem, it is necessary to keep in mind that the strongest electron withdrawing groups that are present in these alcohols are trifluoromethyl, then difluorochloromethyl (CF_2Cl), then hexafluoropropyl (which is similar to CF_3, but one fluorine is replaced by a less electron withdrawing C_2F_5), then trichloromethyl,

then hydrogen, then methyl, and the hexyl group is the least electron withdrawing. The effect of each group is nearly additive, so three CH_3 groups in **6** are less electron withdrawing than two H and one hexyl in **1**. In some of these compounds, a strong electron withdrawing effect of one group is partially canceled by an electron donating effect of another group, for example, in **9**, where CCl_3 and CH_3 partially compensate each other's effect. In other compounds, the effect of two groups adds up, for example, in **4**, where two CF_3 withdraw the electronic density from the OH group, whereas in **6** three CH_3 *donate* the electronic density to OH, thereby making this alcohol least acidic.

10 points:

Some compound has an empirical formula $C_6H_5F_2Cl_2BrO_2$. Which isomer demonstrates the strongest acidic properties? Explain your reasoning.

Hint:

The hint to the 5pt problem can be used for this problem too. However, keep in mind that the molecules where a halogen atom and OH group are connected to the same carbon atom have low chemical stability.

Solution:

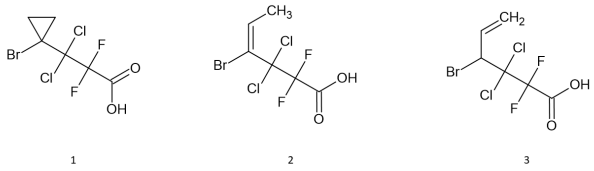
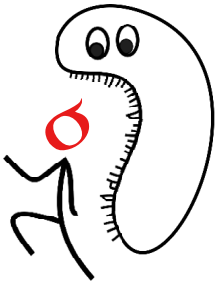


Figure 4: Three the most acidic compounds

It is clear from the formula, that the most probable candidate is some carboxylic acid. In this acid, the most electronegative (and, therefore, the most electron withdrawing) substituents are two fluorine atoms, which must be located as close as possible to the COOH group. That can be achieved if the molecule is linear. Two other subsituents with a very high electron withdrawing effect are chlorines, and they must occupy the next closets position. Therefore, the molecule must be some derivative of 2,2-difluoro-3,3-dichloro propionic acid, as shown at the figure. With regards to the rest, bromine should occupy the position 4, and the remaining alkyl chain should contain either a double bond or a cycle. Since this alkyl tail is very far from the COOH group, its effect on the acidity is very small, so the properties of compounds **1-3** should be nearly identical, although **3** may be a little bit less acidic, because sp^3 is less electronegative than sp^2.

Featured Solution
by Maya Pevzner
https://www.youtube.com/watch?v=hrRrPD4XZZA



Biology

Introduction

For this month's Biology problems, we require you to use a protein structure visualizing tool. There are many tools available online, and you are allowed to use any of them to solve these problems. There are a few visualizers we recommend you use, including Protein Data Bank (PDB), AlphaFold, or Chimera[2]. **You can use any protein structure visualizing tool. Additionally, we ask that you submit a PDF featuring your structures from the 5pt question.**

This month's POM topic is structural biology, and protein structuring tools are widely used by structural biologists for the visualization and analysis of protein structures. These structures are typically determined using a powerful technique called X-ray crystallography. To date, a vast number of protein structures have been solved using this method. All such structures are deposited in a single large database known as the Protein Data Bank (PDB), which is freely accessible to everyone. Each structure has a unique four-character identifier, e.g., 2PYH.

5 points:

Protein structures usually appear very messy, making them difficult to interpret (Fig. 1 as an example). For this reason, scientists often represent them in simplified forms.

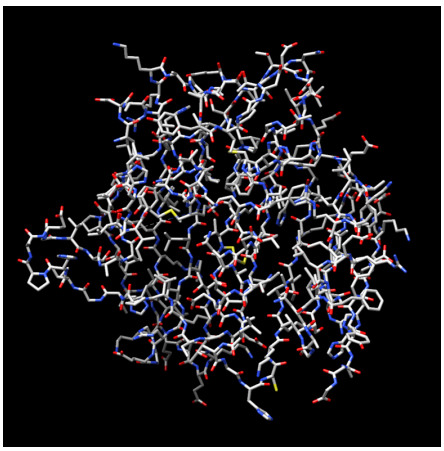


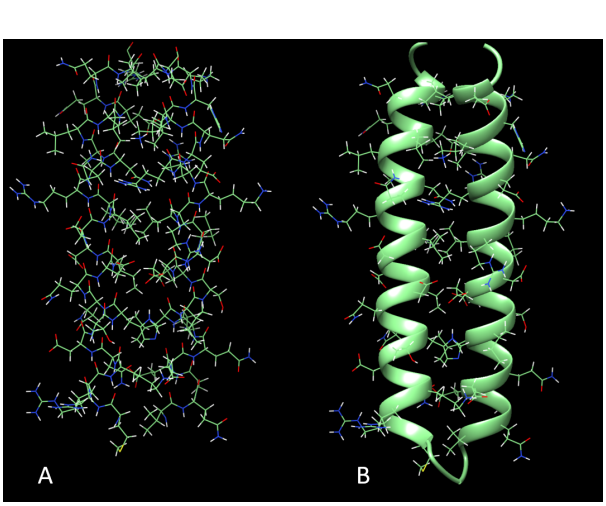
Figure 1: An example of a protein structure where all atoms except hydrogens are shown explicitly

For example, it is sometimes convenient to display only the protein backbone while omitting the amino acid side chains. This makes the structure much clearer, because even those protein structures that appear irregular and complex may show some regular patterns that are otherwise invisible. The most common patterns are spirals and sheets (the latter may bend or fold into more complex shapes). The examples are shown below in Fig. 2.

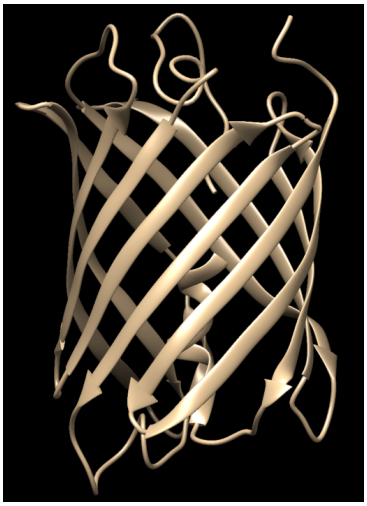
More frequently, protein structure looks like a combination of spirals, sheets, and less regular motifs (Fig.3 shows a typical example).

Interestingly, some proteins have highly regular and visually striking structures. Several examples are shown below (see the figures 4, 5, and 6). As a rule, these structures consist of bundles of spirals and sheets, similar to those shown in the first picture.

[2]For more instruction on using Chimera for both questions, read the section after the 10pt on "Help with Chimera".



(a) An example of a spiral motif with (A) and without (B) explicitly shown atoms (PDB ID 1A93)



(b) An example of a protein structure that represents a sheet folded as a cylinder (PDB ID 4KF4)

Figure 2: Comparisons of spiral and sheet-based protein motifs, as well as a cylindrical structure.

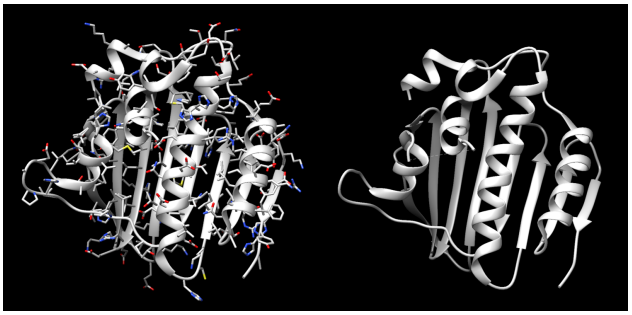


Figure 3: The figure shows the protein shown in Fig. 1, but the backbone is highlighted (left panel). The right panel shows the backbone only.

By browsing the protein visualizing tools, try to find as many beautiful and impressive structures as possible and visualize them using PDB, Alphafold, or Chimera.[3]

Try to read about the spiral and sheet motifs in proteins and explain in your own words why these two motifs are the most common in proteins. Find 3 to 5 protein structures that look the most beautiful, and make the images as spectacular as possible. Try to find some information on these proteins and, if possible, explain how their amazing shape is linked to the role they play in a cell.

In the PDB, this is the PDB ID (as shown in the captions of the figures). In AlphaFold, this is the UniProt code.

Hint:

What do proteins need to do physically, where their shape can affect their performance/function? How can the shapes (spirals, sheets, etc) affect this function?

[3]For additional help using Chimera, see "Help with Chimera" at the bottom of the document

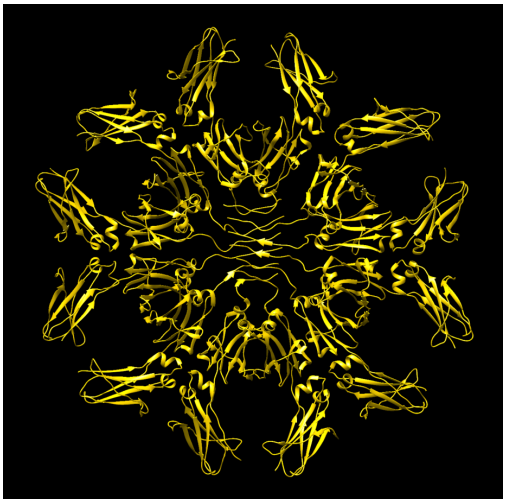
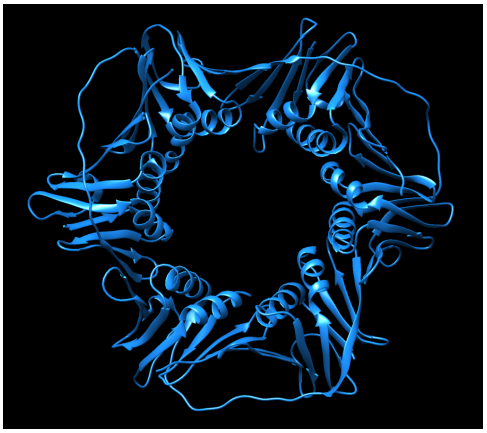


Figure 4: Example 1 (PDB ID 7X13)

(a) Example 2 (PDB ID 5I50)



(b) Example 3 (PDB ID 2HIK)

Solution:

Answer: There were no explicitly wrong or correct answers to this question. Through some searching on the various platforms we provided, you could find beautiful and visually striking protein structures. Here are some examples below.

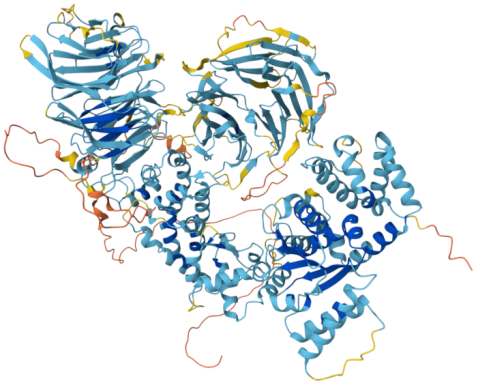


Figure 6: Apoptosome protein Dark from AlphaFold (UniProt Q7KLI1)

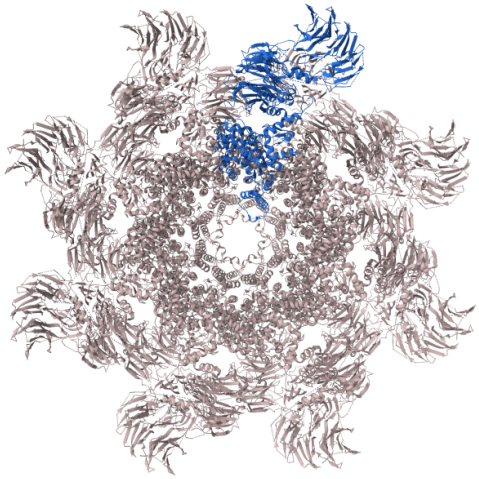


Figure 7: Apoptosome protein Dark (PDB 3J9L)

Figure 8: B-DNA Dodecamer (PDB 1BNA)

The reason why proteins fold into common motifs like sheets and spirals is primarily for stability. A protein's backbone must fold tightly to push H_2O out of its core, but it also has to leave just enough space for its atoms to fit along the curved chain and move slightly without colliding. The names of these common motifs, spirals and sheets, are called α-helix and β-sheets.

The connection between protein structure and cellular function can be very difficult. Put simply, the conformation of a protein affects where it can bind, where it fits, and what it can do in the cell. Proteins are often flexible, and there are many changes to their conformation that could change their function, most times registering the protein nonfunctional. There are also various levels of protein structure, from primary (amino acid sequence) to quaternary (multiple folded polypeptide chains).

Read here for more information about protein folding, read these detailed references from Penn State and LibreTexts Bio or LibreTexts Chem.

10 points:

Introduction

Most biological processes rely on interactions between proteins and small molecules (ligands or substrates). These interactions are governed primarily by three types of noncovalent forces: (i) electrostatic interactions between oppositely charged groups, (ii) hydrophobic interactions between nonpolar surfaces, and (iii) hydrogen bonds.

Hydrogen bonds form when a hydrogen atom covalently bound to an electronegative atom (typically oxygen or nitrogen, and more rarely sulfur) interacts with another electronegative atom bearing a lone pair of electrons. Although individually weak—typically one to two orders of magnitude weaker than covalent bonds—hydrogen bonds are highly directional and play a key role in molecular recognition.

Water provides a classic example. In liquid water or ice, each molecule participates in four hydrogen bonds: two as a donor via its hydrogen atoms and two as an acceptor via the oxygen lone pairs. In water, the O–O distance across a hydrogen bond is approximately 3 angstroms, compared with an O–H covalent bond length of about 1 angstroms.

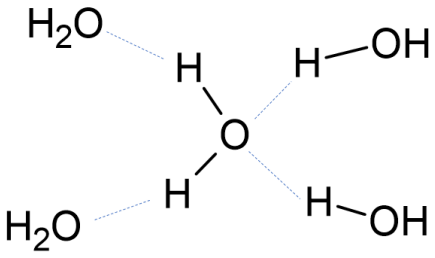


Figure 9: Hydrogen bonds in water (shown as thin dashed lines).

Understanding these interactions is central to structural biology. For example, biotin (vitamin B_7) binds very tightly to streptavidin. Structural analysis reveals that biotin fits into a pocket on the protein surface, forms extensive hydrophobic contacts, and establishes multiple hydrogen bonds with polar protein groups.

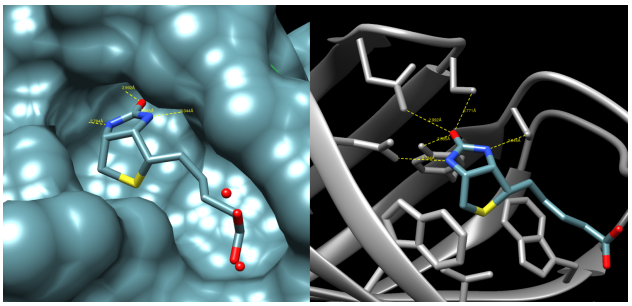


Figure 10: Biotin–streptavidin complex (PDB ID 1STP). The left panel shows the protein surface; the right panel highlights hydrogen bonds (yellow dashed lines).

Because most drugs are small molecules that bind specific proteins, analyzing protein–ligand interactions is essential for rational drug design.

The Problem

During the COVID-19 pandemic, extensive efforts focused on inhibiting the replication of the SARS-CoV-2 virus. One promising strategy targeted the viral main protease, Mpro, an enzyme essential for viral replication. A research group identified compound **A** (Fig. 11) as a moderate Mpro inhibitor.

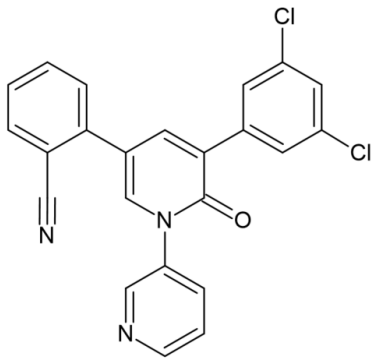


Figure 11: Structure of compound **A**. Bonds connecting the six-membered rings can rotate.

The structure of compound **A** bound to Mpro was subsequently determined (Fig.4). This structure can be downloaded from the Protein Data Base (PDB ID 7L10). Analysis of this complex shows that **A** occupies a binding pocket on the protein surface, makes extensive hydrophobic contacts, and forms two hydrogen bonds that stabilize the complex.

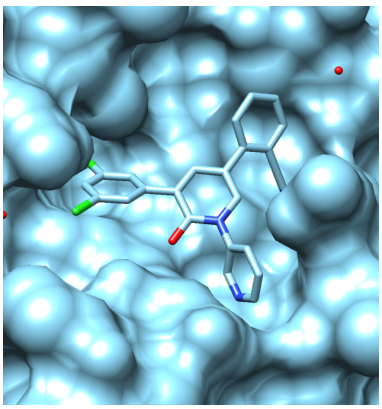


Figure 12: Structure of compound **A** bound to Mpro (PDB ID 7L10). Only a portion of the protein surface is shown.

Your task is to identify the atoms in compound **A** that participate in hydrogen bonding with Mpro. Indicate which parts of the molecule are essential for binding and which do not appear to contribute significantly. Finally, based on the structure of the complex, propose positions in compound **A** that could be modified to improve binding affinity while preserving key interactions.

Help with Chimera (optional, only if you are using Chimera)

Chimera is a widely used tool by structural biologists, and learning how to use it could be very useful for you in the future. Below are some notes that could help you to learn how to use it, but if you have further questions, feel free to email us.

Sometimes, instead of one protein molecule, the structure downloaded from PDB contains several identical molecules grouped together. In that case, you may select some of them and hide them using one command from the Actions menu. Once you have identified a structure, Chimera can fetch it directly using its four-character identifier (PDB ID). Chimera allows many different ways to represent protein structures: you can display the full atomic structure, only the molecular surface, or just the backbone, among other options. You can also experiment with different colors and lighting, for example by moving the light source

to illuminate the structure from different angles. For better visualization of structures, we recommend that you use the Actions menu (Atoms/Bonds, Ribbon, etc; the Focus and Set Pivot options may be helpful, too). To select some specific part of the molecule, use Ctrl-left mouse. You may also find useful Tools and Favorite sub-menus.

We also recommend using large language models (LLMs), such as ChatGPT, to obtain additional instructions on how to install and use Chimera.

For the 10pt problem, download the structure 7L10 from the Protein Data Bank (Chimera can do this directly using the "Fetch by ID" command in the File menu). When the structure opens, the protein is typically displayed as cartoons (showing only the backbone), whereas the ligand is shown in full detail. You can visualize more or fewer structural features by selecting specific parts of the molecule (Ctrl + left-click) and using the commands in the Action menu. You may measure distances and perform other analyses using the Tools menu (especially the Measurement submenu). Keep in mind that most PDB structures do not display hydrogens, so to identify hydrogen bonds, you should measure distances between heteroatoms (for example, between a carbonyl oxygen and an amide nitrogen).

Hint:

No hint this month.

Solution:

The optimal length of a hydrogen bong is about 1.8 angstrom, and the length of a NH of OH bond is approximately 1 angstrom. Therefore, it is reasonable to expect that if some hydrogen bond acceptor (e.g., a carbonyl oxygen or an aromatic nitrogen) and a hydrogen bond donor (e.g., an amide nitrogen in the peptide backbone, or a nitrogen in a histidine side chain) are separated by 3 angstroms or less, it is reasonable to expect there is a hydrogen bond between them. To measure distances in Chimera, you can use the "Tools-Structure Analysis-Distances" menu. Keep in mind that to measure a distance, you have to select two atoms. Selection of one atom can be done by a ctrl-left mouse; to select two or more atoms, you must also hold the shift key. As soon as you have selected a pair of atoms, the rest is obvious.

However, there is even a simple way to do that: the Chimera program has an automatic procedure for H-bond detection. To use it, go to the Tools menu and find the "FindHBonds" option, as shown on the figure below.

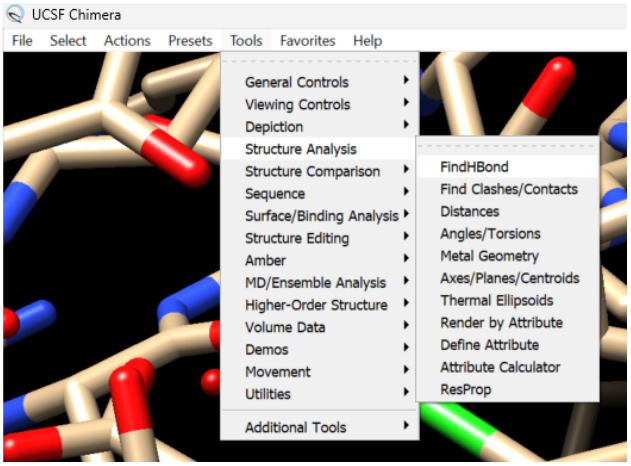


Figure 13: "Find H-bonds" tool

A dialogue window appears (see below), where you can use the default options.

After you press "Apply", all possible hydrogen bonds appears as thin lines (see the figure).

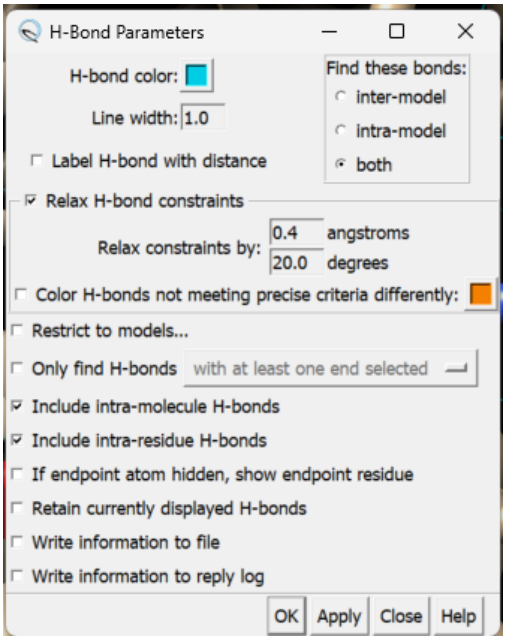


Figure 14: FindHBonds options

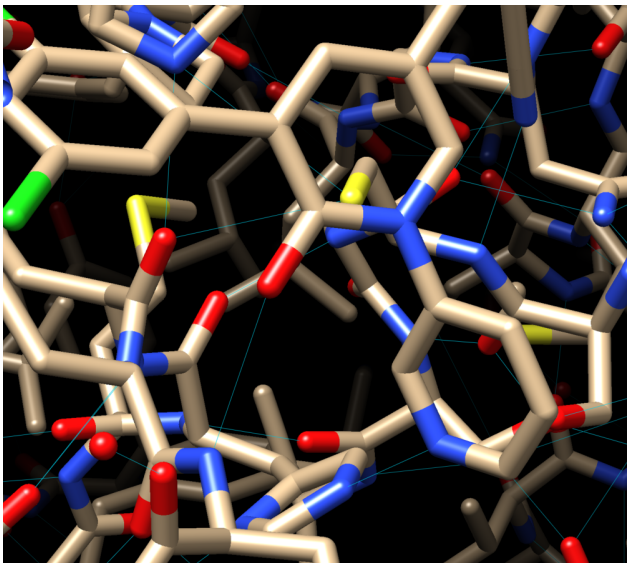


Figure 15: Autodetected hydorgen bonds

The bonds that form between our drug candidate and the enzyme are shown on the figure below (I used the "Distances" tool as described at the very beginning): It is easy to see that the two distances shown as yellow dashed lines are approximately 2.8 angstroms, which is an optimal length for an H-bond.

Furthermore, if you move a cursor to the amino acid residue, the residue's name appears on the screen. On the figure, the residue is His 163.A, which means the residue name is histidine, its number is 163, and it belongs to the protein molecule A (sometimes, a PDB structure contains more than one protein molecule). If you put a mouse on the second residue, you will see that it is Glu 166.A, i.e., a glutamic acid residue number 166.

These are the two hydrogen bonds that are the most important for the interaction between the drug candidate

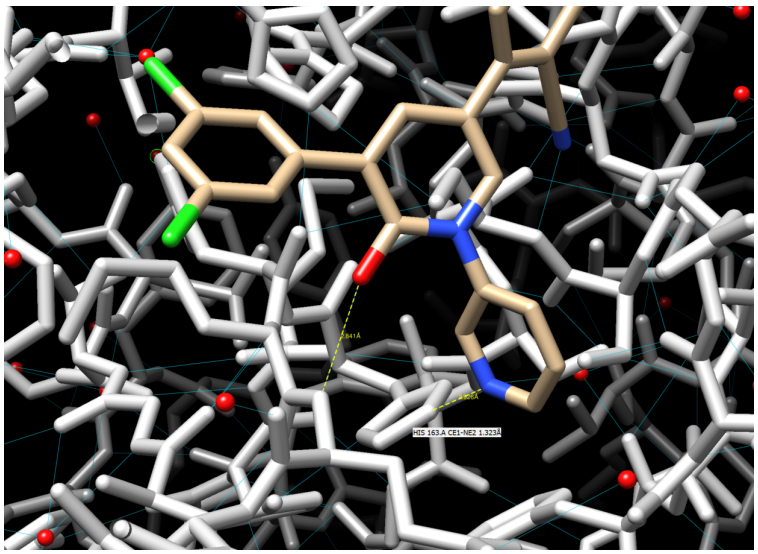


Figure 16: Hydrogen bonds between Mpro and the drug candidate. The protein molecule is gray.

and Mpro.

It is also easy to see that the dichlorobenzene ring occupies a deep pocket in the enzyme molecule, and one of the chlorine atoms is involved in multiple interactions with hydrophobic amino acid chains. Another chlorine is mostly exposed to the solvent, so it is much less important for the interaction. Clearly, it can be substituted with other groups. That is exactly what happened in reality: the group of researchers who was working on this project replaced one chlorine with a bigger group and created a new compound that appeared to be a much more efficient binder (see the figure below).

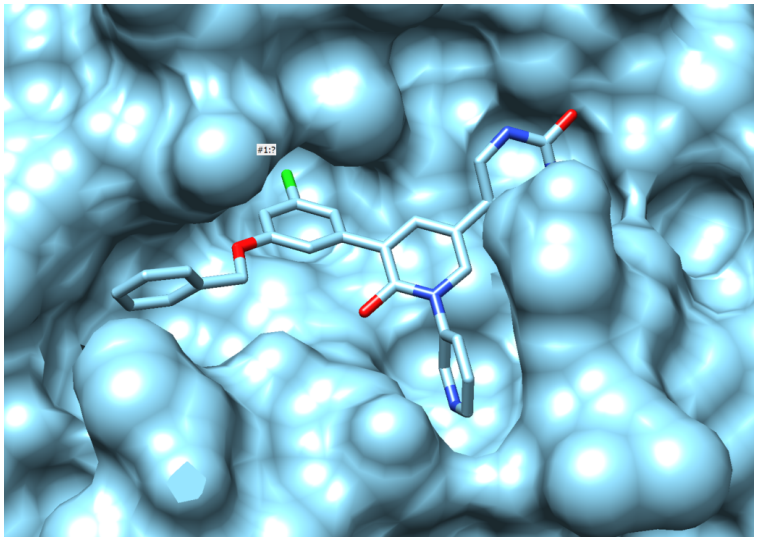


Figure 17: A better ligand that was obtained by a chemical modification of the compound **A** (PDB ID 7L12)

Computer Science

- Your program should be written in Java or Python-3.
- No GUI should be used in your program (e.g. `easygui` in Python).
- All the input and output should be done through files named as specified in the problem statement.
- Java programs should be submitted in a file with extension `.java`; Python-3 programs should be submitted in a file with extension `.py`. **No `.txt`, `.dat`, `.pdf`, `.doc`, `.docx`, etc. Programs submitted in the incorrect format will not receive any points!**

The game of Chomp starts on a board with n columns and m rows, where players take turns selecting a square on the board. Whenever a square is selected, all squares below and to the right of the selected square are removed from the board. The player who removes the last square in the board looses.

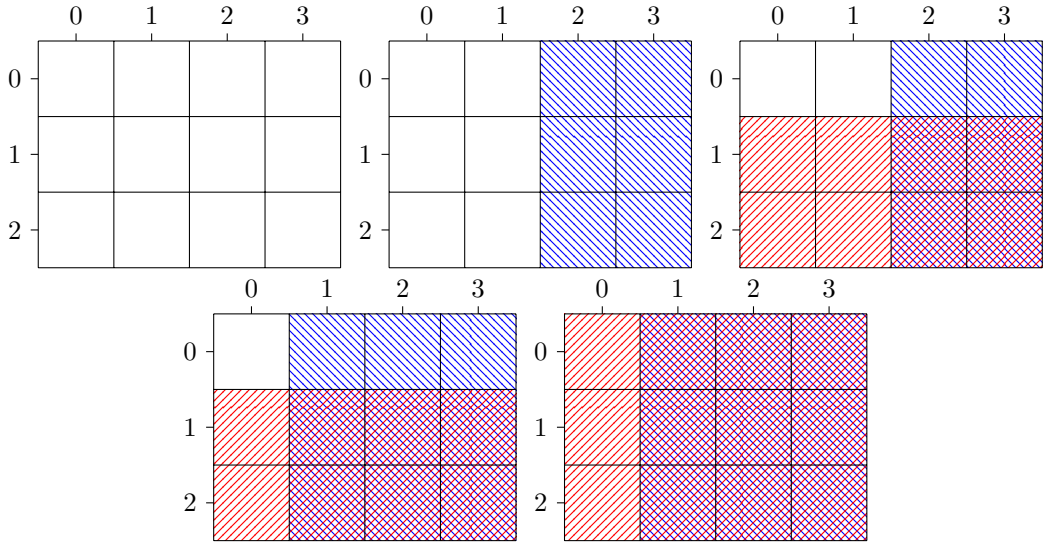


Figure 18: An example game of Chomp with $(n, m) = (4, 3)$. Alice (blue) and Bob (red) take turns selecting squares on the board, and the red and blue regions of the grid are no longer considered part of the board. The game proceeds as follows: Alice selects (2, 0); Bob selects (0, 1); Alice selects (1, 0); Bob selects (0, 0). Since Bob removes the last square of the board, he loses.

5 points:

Given a starting board size and several moves that have been made, your task is to compute the number of squares left on the board. **Note:** The board may have trillions of squares, but at most 1000 moves will have been made.

As an example, consider the 5×4 board below, which results from the moves (0, 3), (3, 2), (2, 1). Note that we no longer color the moves differently, as who made each move does not affect the result of the problem.

After the three moves, the board has 9 squares left. Note that sorting the moves by their x (or y)-coordinate may be helpful!

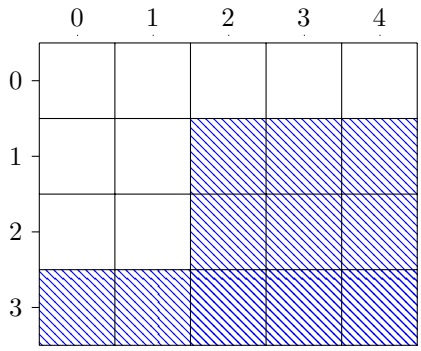


Figure 19: An example board after several moves

To perform this task, your program should read the file `input.txt`, which will contain the following:

- Line 1: Will contain the integers n, m, a, where a is the number of moves that will have been made.
- Lines 2 to $a + 1$: Each line will have position (x, y) on which a player made a move.

Note that the board is 0-based; the top-left square of the board is $(0, 0)$, and initially the bottom-right square is $(n - 1, m - 1)$.

Your program should write the number of squares remaining on the board to `output.txt`

First Sample `input.txt`:

```
5 4 3
0 3
3 2
2 1
```

First Sample `output.txt`:

```
9
```

Second Sample `input.txt`:

```
100 200 3
10 90
80 20
10 50
```

Second Sample `output.txt`:

```
5900
```

Hint:

No hint this month.

Solution:

After creating suitable representations for moves and sorting them by their x-coordinate, we can compute the total area by breaking it up into rectangles whenever the height of the remaining squares on the board changes.

```python
from dataclasses import dataclass

@dataclass
class Move:
    x: int
    y: int

def compute_remaining_squares(n, m, moves: list[Move]) -> int:
    moves = [Move(0, m)] + moves + [Move(n, 0)]
    moves.sort(key= lambda move: move.x)

    #Prunes away moves that don't affect the 'path' of the board state (ie. they are too low)
    i = 0
    while i < len(moves) - 1:
        if moves[i+1].y >= moves[i].y:
            moves.pop(i+1)
        else:
            i += 1

    total_area = 0
    for i in range(len(moves) - 1):
        height = moves[i].y
        width = moves[i+1].x - moves[i].x
        total_area += height * width

    return total_area

def solve_5pt_dec(input_file: str, output_file: str):
    moves = []

    with open(input_file, "r") as file:
        n, m, a = file.readline().split(' ')
        n = int(n)
        m = int(m)
        for _ in range(int(a)):
            x, y = file.readline().split(' ')
            move = Move(int(x), int(y))
            moves.append(move)


    result = compute_remaining_squares(n, m, moves)

    with open(output_file, "w") as file:
        file.write(str(result))

if __name__ == "__main__":
    solve_5pt_dec("input.txt", "output.txt")
```

10 points:

Given a board state in Chomp, we can categorize it as *winning* if the player who's turn it currently is can guarantee a win with perfect play, and *losing* if their opponent can guarantee a victory with perfect play.

Equivalently, we can recursively define *winning* and *losing* states as follows:

- The state with only one square left is *losing*, since the player's only move will be to remove the last square.
- A state is *winning* if there exists some move that will place the opponent in a *losing* state.
- A state is *losing* if every possible move places the opponent in a *winning* state.

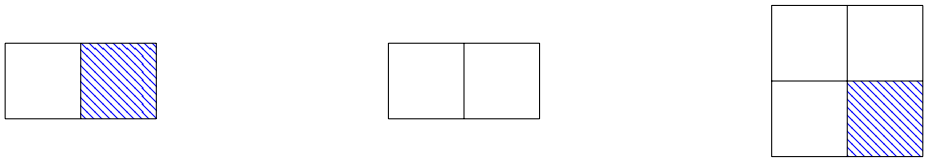


Figure 20: From left to right, these boards are: *losing*, since only one square is left; *winning*, since the current player can play at (1, 0) to put their opponent in a *losing* state; and *losing*, since any move will either lose immediately, or put the opponent into a state equivalent to the middle one, which is *winning*

For this problem, your task is to take in a board state (a starting rectangular board and several moves that have been made) from the file `input.txt`, which will have the exact same format as in the 5pt. However, in the 10pt problem, **the starting board will have $\mathbf{n}, \mathbf{m} \leq \mathbf{8}$**. Then, you should:

- Write `losing` to `output.txt` if the state is *losing*.
- Otherwise, write `winning` to `output.txt`.

We provide some sample inputs and outputs.

First Sample `input.txt`:

```
5 5 1
```

First Sample `output.txt`:

```
winning
```

Second Sample `input.txt`:

```
2 2 1
1 1
```

Second Sample `output.txt`:

```
losing
```

Potentially Helpful Information. Firstly, note that starting on an $n \times m$ board, for every square that's been removed, the squares below or to the right must also have been removed. So, by corresponding board states to the path traversing the bottom tiles in the state, we can see that the number of possible board states is equal to the number of paths that only go up and right from the bottom left vertex to the top right vertex, as shown in Figure 21.

Any such path goes up m times and right n times, so the number of such paths is $\binom{m+n}{n}$, so the number of possible board states is $\binom{m+n}{n}$. Note that this means that for the 10pt question, the largest possible number of board states will be $\binom{8+8}{8} = 12870$.

Secondly, it is actually possible to prove that if $(n, m) \neq (1, 1)$, then when starting on an $n \times m$ board the first player always has a winning move:

Let S be the set of all squares other than $(n-1, m-1)$. Note that since $(n, m) \neq (1, 1)$, S contains at least one square. Then,

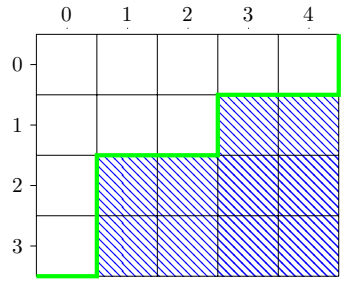


Figure 21: An example on a 5×4 grid. The 'path' corresponding to the board state is highlighted in green.

- If selecting some square in S would guarantee a victory for the first player, then the first player has a winning move.
- Otherwise, since selecting any square in S is a losing move, all of the resulting boards are *winning*. So, the first player can select the square $(n-1, m-1)$. On the next move the second player must pick some square in S, which will place the first player in a *winning* position. Note that the board that results from first selecting $(n-1, m-1)$, then selecting the square $(x, y) \in S$ is the same board that results from just selecting (x, y). So, selecting $(n-1, m-1)$ is a winning move for the first player.

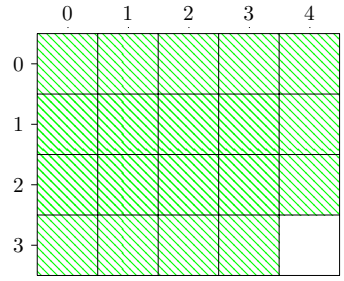


Figure 22: An example of the argument above on a 5×4 grid. If selecting any of the green squares would be a winning move, the current player can select them and win. Otherwise, they are all losing. So, selecting the bottom right square (4, 3) will force the second player to make a losing move on the next turn.

Note that this argument does not actually specify *which* move is winning for the first player; just that one exists!

Similar mathematical arguments may help with other classes of board states. We encourage (but do not expect) you to use such techniques to help you either validate the correctness of your algorithm, or to cut down on the amount of work your algorithm has to perform!

Hint:

No hint this month.

Solution:

Featured Solution
by Zachary Lubashev
https://www.youtube.com/watch?v=GbKNZ1gn57w