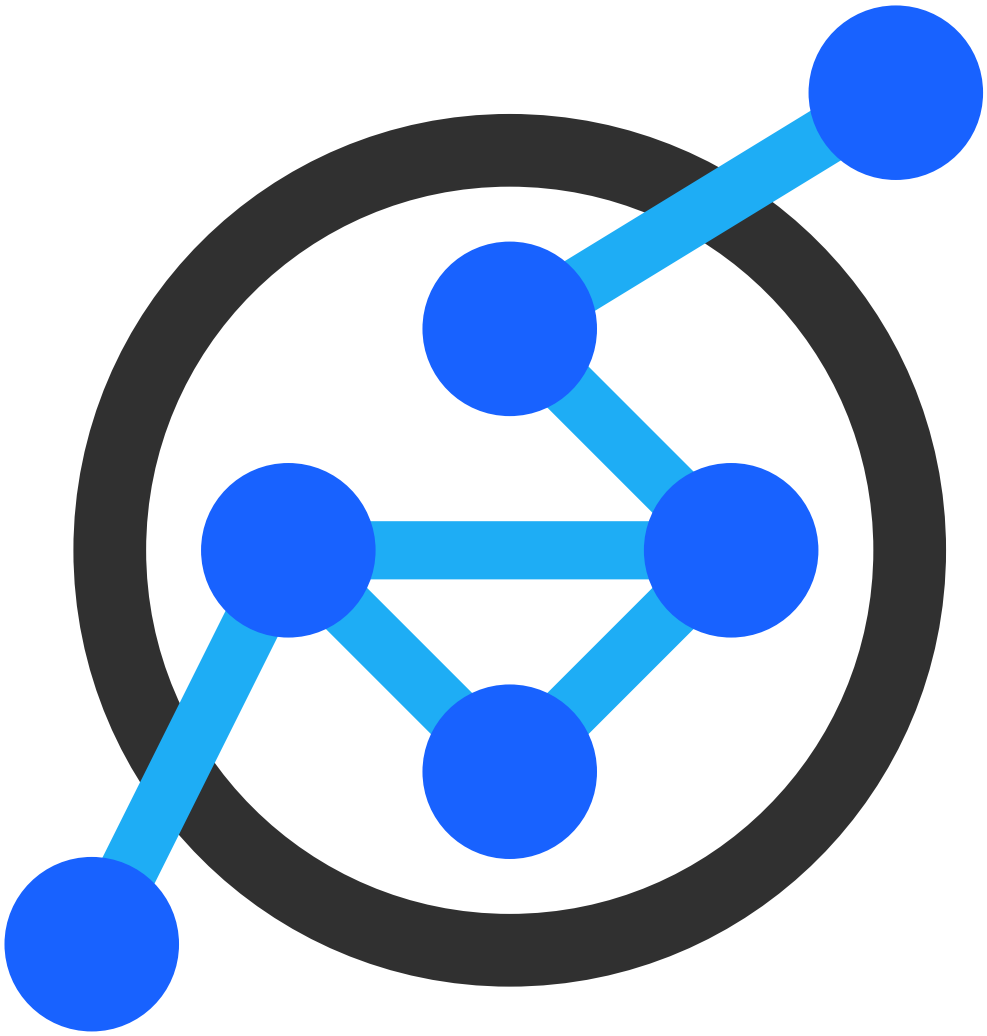
# Second Round Theoretical

# Tasks



Swiss Olympiad in Informatics

March 4, 2023

# Instructions

- Open your exam only after you have been told so. The examination starts for everyone at the same time and lasts 5 hours.
- With the exception of watches (no smart watches), no electronical devices are allowed on your table. Switch off your mobile phone.
- For all *task-related* questions, please use the provided question sheet.
- Start each task on a a separate sheet of paper and write your name on every sheet. Number your sheets and sort them before handing them in.
- Do not use pencil and do not write with red.
- Write legible.

## Grading

The solutions will be graded according to similar criteria as the first theoretical round. The most important criteria are correctness and asymptotic running time. The quality of the description and the arguments asserting the correctness will also be taken into account.

You can always refer to some content of the SOI Wiki or 2H. You don't need to explain why e.g. Dijkstra works, but you should argue why and how it can be applied. In case of Dijkstra you should clearly state on which graph you run it on, and note that the edge weights are not negative.

The algorithm should be described in enough detail that it is easy to convert the description into a program. Also write down which data structures you would choose. Usually it is best to just write a short pseudocode.

To describe an algorithm, you should structure your solution as according to the following guideline:

1. Describe the idea for an algorithm that solves the problem.
2. Give pseudocode or explain how one would implement the algorithm.
3. Argue about the correctness of the approach.
4. Indicate asymptotic running time and memory usage.

If some part of your solution can be used for multiple subtasks, it suffices to write it down only once and refer to it from other parts. Note, however, that a general algorithm might be able to solve the previous subtasks, but not necessarily optimally.

# Goulash

As a preparation for the IOI in Hungary, Mouse Stofl wants to cook a big pot of goulash (a traditional Hungarian soup) for all his friends. Unfortunately, he was too quick to start and forgot to get the ingredients from the basement. Now that he has started, he has to keep stirring the goulash and cannot leave the pot unattended. Of course, his friends have volunteered to help him fetch the ingredients from the basement.

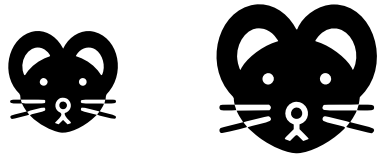
There are $N$ ingredients needed for the goulash and Stofl has $M$ friends to help fetch the ingredients from the basement. The ingredients in Stofl's basement are ordered on shelves. Each ingredient $i$ is located at a specific heigth $h_i$ and each of Stofl's friends $j$ has a certain maximum heigth $m_j$ they can reach.

As everyone is very hungry, they want to finish the goulash as soon as possible. The goulash will be done once all the needed ingredients are added to the pot. Note that the order in which the ingredients are added does not matter. Each of Stofl's friends can carry exactly one ingredient at a time. It is not possible for them to bring up multiple items in one trip to the basement. Conveniently, all of Stofl's friends walk at the same speed and it takes them 1 minute to go to the basement and 2 minutes to come back with whatever they are carrying.

Given a list of ingredients and Stofl's friends can you help them to figure out whether they can make the goulash and calculate how long it will take them to finish?

An example of how this can look can be seen below.

| $i$ | Ingredient | Height |
|---|---|---|
| 0 | Salt | 2 |
| 1 | Tomato | 4 |
| 2 | Potato | 3 |



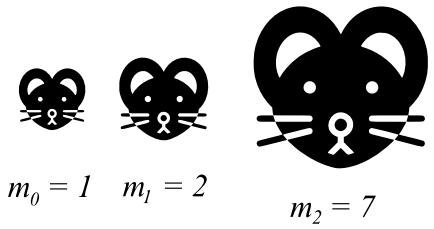
$m_0 = 1$ $m_1 = 4$

In this example, Stofl has two friends and there are three ingredients needed for the goulash. Unfortunately, friend 0 is not able to reach any of the ingredients. Thus friend 1 needs to go to the basement three times to get the three ingredients taking $3 \cdot (1 + 2) = 9$ minutes.

## Subtask 1: Solve an example (10 points)

Solve the problem for the following goulash recipe with the help of the three depicted friends of Stofl.

| $i$ | Ingredient | Height |
|---|---|---|
| 0 | Olive oil | 1 |
| 1 | Steak | 4 |
| 2 | Onion | 7 |
| 3 | Garlic | 1 |
| 4 | Tomato | 3 |
| 6 | Paprika | 5 |
| 7 | Parsley | 2 |




Write down exactly when which mice go to the basement and what they carry up. You do not need to explain your answer in this subtask.

## Subtask 2: Check whether it is possible (10 points)

Design an algorithm that determines whether it is possible to cook the goulash. This algorithm should output "*Yes*" if it is possible for Stofl's friends to get all the ingredients from the basement and "*No*" otherwise.

## Subtask 3: Solve the general case (80 points)

Design an algorithm to help Stofl determine the minimum time his friends need to get all the ingredients from the basement. You can assume $N > M$ to judge the efficiency of your algorithm.

# Jogging Map

Mousarnen is a quiet and secluded village, with $L$ houses surrounded by huge gardens, each with **exactly a single** driveway connecting to a location. *Roads* in Mousarnen, each of them precisely 1 mouse meter long, connect two *locations*, where a location is either one of the $L$ *houses* or one of the $Y$ *intersections* in Mousarnen. (Therefore, a *driveway* is just a special type of *road*.) The inhabitants of Mousarnen don't care much about road infrastructure, so there are no more roads than necessary. If you ask a Mousarner about it, he will just shrug his shoulders and say: *"Why should we have more? No matter where I am in Mousarnen, I can get to any other place, so that should be enough."* That is why there are exactly $L + Y - 1$ roads in Mousarnen.

This is the place where Mouse Binna lives and where the famous MOI [1] camp takes place. Mouse Binna and Mouse Stofl were able to qualify for the camp this year. They are very good friends, but couldn't be more different in one respect. While Mouse Binna takes a daily jog around the village, Mouse Stofl rarely finds the motivation to engage in any kind of physical activity.

The leaders of the MOI provide their participants with tons of snacks, and although Stofl enjoys them very much, it finally occurred to him that maybe he should start doing some exercise for the sake of his health. He wants to take an example from Mouse Binna and intends to start jogging as well. To find the perfect jogging route, Stofl needs a map of Mousarnen. More precisely, he wants to know which locations are connected to each other.

Fortunately, Binna is coming back from a jog. While she doesn't have a map of Mousarnen, she can look up the distance between any two of the $L + Y$ locations, thanks to her state-of-the-art wristwatch. Due to the cryptic proprietary format, it is unfortunately not possible to distinguish directly from the watch, if the location is a house or an intersection. Though, our mice know the number of houses $L$ and the number of intersections $Y$. Mouse Binna wants to continue jogging as soon as possible, so she asks Stofl to keep his questions to a minimum.

You can assume that the locations are numbered from $0, 1 \ldots L + Y - 1$. You are able to ask queries of the form $d(a, b)$ which gives you the number of roads you pass through while jogging from location $a$ to location $b$.

## Subtask 1: Solve an example (15 points)

Construct all possible maps with locations $0, 1, 2, 3, 4, 5$ given the following questions and answers:

- $d(4, 2) = 1$
- $d(3, 3) = 0$
- $d(3, 0) = 2$
- $d(3, 5) = 2$
- $d(0, 2) = 3$

Shortly explain, why other maps are not possible.

## Subtask 2: Solve the general case (65 points)

Can you help Stofl construct the map of Mousarnen with the least amount of questions asymptotically in the worst case scenario?

[1] Mouse Olympiad in Informatics

Your priority is to optimize for the number of questions first, then the runtime and at last the memory. All of them asymptotically, meaning in big-$O()$, and in the worst case.

With worst case we mean for the worst map possible and that the watch doesn't need to choose its location labels before you ask the first question. It can adapt its *'solution' map* in the background, as long as it stays consistent with the answers already given. [2].

In order to get full points, you need to determine the map of Mousarnen in $O(Y \cdot L)$ queries. Asking $O((L+Y)^2)$ questions can get you up to 10 points. Note that you can get more than 10 points, if your number of questions lie between $O((L+Y)^2)$ and $O(Y \cdot L)$.

For solving the special case of maps where $L = 2$, meaning that all the locations lie on one line, you can get up to 10 points.

Note: You **can** get both types of partial points in this subtask. If you solve the special case efficiently and the general case in $O((L+Y)^2)$ questions correctly you can get 20 points.

## Subtask 3: Show a Lower Bound (20 points)

Mouse Binna is a bit annoyed that she had to wait so long and claims, that Mouse Stofl was asking his question inefficiently. To defend himself after Binna comes back from her next jog, Stofl wants to show Binna an example of a map where you need to ask a certain number of questions to be certain to have a correct map.

Given the number of locations $N$, can you help him construct a map where any algorithm would need at least $\frac{N^2}{2023}$ questions in their worst case? You are free to choose the number of houses $L$ and intersections $Y$, as long as they fulfill $N = L + Y$. You can assume being unlucky with the labels, i.e. the watch changing labels of not yet asked locations.

To get full points you need to *reason why* your chosen map cannot be constructed in less than $\frac{N^2}{2023}$ questions.

[2] That way the company can sell the improved watch the next year

# Coding Accident

Mouse Binna is participating in a programming contest. She was just about to submit her final full score solution to the last task of the contest. However, while submitting, she was involved in an unfortunate *tab completion accident*.[1]

As a result, the order of the lines in her code has been randomly shuffled. This being a programming competition, her code unfortunately had not been under version control and she now has to unshuffle the file contents manually.

Mouse Binna's advanced text editor `emous` has a number of purpose-built commands to assist elite hackers such as herself in accomplishing a task of this kind. A particularly efficient command (`M-x rot3`) allows Mouse Binna to select three distinct lines $i$, $j$ and $k$. The command then simultaneously updates all three lines such that their contents are *rotated*: Line $j$ will contain the previous contents of line $i$, line $k$ will contain the previous contents of line $j$, and line $i$ will contain the previous contents of line $k$.

As a vehement proponent of DRY (Don't Repeat Yourself) principles, Mouse Binna of course did not repeat any line of code. Therefore, for each line of code in her shuffled text file, she knows exactly where it will need to end up.

Mouse Binna very quickly unshuffles her code using excusively `rot3` commands, as described above. Of course, she does so in a maximally efficient way, using the smallest possible number of `rot3` operations.

Can you solve this kind of task generally?

`emous` numbers lines starting from 0. Given is an array of $N$ distinct integers between 0 and $N-1$. The $i$-th integer $p_i$ indicates that line $i$ of the shuffled source code contains line $p_i$ of the original source code. Find a shortest possible sequence of `rot3` operations that will restore the original source code or determine that it is not possible.

For example, consider the following input with $N = 6$:

- $p_0 = 3$
- $p_1 = 0$
- $p_2 = 4$
- $p_3 = 1$
- $p_4 = 5$
- $p_5 = 2$

In this case, we can unshuffle the source code using two `rot3` operations:

1. `rot3`$(0, 3, 1)$. This operation moves line 0 to line 3, line 3 to line 1 and line 1 to line 0. The following input describes the new situation:
   - $p_0 = 0$
   - $p_1 = 1$
   - $p_2 = 4$
   - $p_3 = 3$
   - $p_4 = 5$
   - $p_5 = 2$
2. `rot3`$(2, 4, 5)$. This operation moves line 2 to line 4, line 4 to line 5 and line 5 to line 2. The following input describes the final situation:
   - $p_0 = 0$
   - $p_1 = 1$
   - $p_2 = 2$
   - $p_3 = 3$
   - $p_4 = 4$
   - $p_5 = 5$

   Now, every line of the shuffled file contains the corresponding line of the original file, therefore, we have successfully unshuffled. Furthermore, there is no way to achieve this result with fewer operations.

[1] Namely, she accidentally ran the bash script `./shuffle.sh e.py` instead of `./submit.sh e.py`.

## Subtask 1: Small example (10 points)

Explicitly provide any shortest sequence of `rot3` operations that will successfully unshuffle a file described by the following input with $N = 11$:

- $p_0 = 1$
- $p_1 = 0$
- $p_2 = 3$
- $p_3 = 4$
- $p_4 = 5$
- $p_5 = 2$
- $p_6 = 7$
- $p_7 = 8$
- $p_8 = 9$
- $p_9 = 10$
- $p_{10} = 6$

In contrast to other subtasks, you do not need to justify the correctness of your solution in case it is correct. (But an explanation of intermediate steps may be helpful to score partial points in case the final result is incorrect.)

## Subtask 2: Cyclic Shift of 2023 Lines (10 points)

Optimally solve the following test case with $N = 2023$, or prove that it is not possible:

- $p_0 = 1$
- $p_1 = 2$
- $p_2 = 3$
- $p_3 = 4$
- $p_4 = 5$
- ...
- ...
- $p_{2020} = 2021$
- $p_{2021} = 2022$
- $p_{2022} = 0$

I.e., for $0 \leq i < 2022$, we have $p_i = i + 1$ and we have $p_{2022} = 0$.

You do not have to provide an explicit list of all operations, but you should make clear what they are. Note that now you need to argue why your solution is correct.

## Subtask 3: Cyclic Shift of 2024 Lines (10 points)

Optimally solve the following test case with $N = 2024$, or prove that it is not possible:

- $p_0 = 1$
- $p_1 = 2$
- $p_2 = 3$
- $p_3 = 4$
- $p_4 = 5$
- ...
- ...
- $p_{2020} = 2021$
- $p_{2021} = 2022$
- $p_{2022} = 2023$
- $p_{2023} = 0$

I.e., for $0 \leq i < 2023$, we have $p_i = i + 1$ and we have $p_{2023} = 0$.

You do not have to provide an explicit list of all operations, but you should make clear what they are. Note that you need to argue why your solution is correct.

## Subtask 4: Reversal (30 points)

In this subtask, $N \geq 1$ is arbitrary and the order of lines has been exactly reversed. I.e., we have $p_i = N - 1 - i$.

Design an algorithm that always computes a correct answer in this restricted case. (It should either provide a shortest possible list of `rot3` operations or print that there is no solution.)

## Subtask 5: General Case (40 points)

In this subtask, there are no further restrictions. I.e., the lines can have been shuffled arbitrarily. Design an algorithm that always computes a correct answer. (It should either provide a shortest possible list of `rot3` operations or print that there is no solution.)

You can obtain up to 20 points if you find a solution that is suboptimal by at most a constant factor (you still have to prove that this is the case). Algorithms that use at least three times the optimal number of operations will be awarded at most 10 points.

The description of a good algorithm and its correctness proof are allocated an equal amount of points.

# Odd Canals

A research team led by Mouse Binna has discovered the ruins of the former capital of the Soian people, an ancient tribe known for their technological advancements and their cryptic puzzle-solving skills. The ruins are crisscrossed by a system of canals that probably once served to transport goods. It consists of $2 \cdot N$ sites connected by $M$ canals. Mouse Binna had long suspected that the Soians ascribed certain meanings to certain numbers. Especially, numbers that have a remainder of 1 after dividing the number by 4 are considered lucky. In fact, the number of canals connected to a site is always a multiple of 4 plus 1, as if the Soians had taken great care to incorporate their lucky numbers into their architecture.

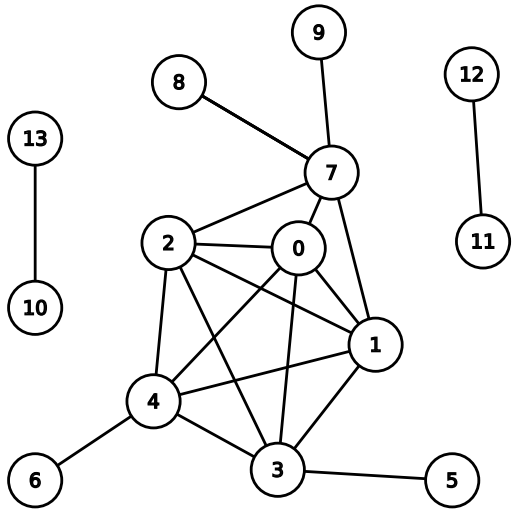
Unfortunately, the Mouseity Council has decided to make the ruins accessible as a tourist attraction, despite fierce opposition from scientists. Mouse Binna is now working on a concept for tourist visits that will ensure respectful treatment of the Soian remains. First and foremost, she wants tourists to travel only on the canals and not walk through the ruins. For this to work, each canal must be given a direction for the boats to pass through to avoid collisions. In addition, it is important to her to maintain respect for the Soian culture and to incorporate at least some of their beliefs into the concept. From a Soian legend, we learn that a journey doesn't always have to be lucky, but rather balanced in all kinds of ways. That is why Mouse Binna wants the number of canals leading away to be odd for $N$ sites and even for the other $N$ sites.

For the concept to have a chance of being approved by the city council, she needs your help to determine the directions of the canals.

## Subtask 1: Solve an example (10 points)

Mouse Binna found some construction drawings hinting at alternative layouts for the former capital. In these drawings, sites are represented by circles and canals by lines.

She took one of a smaller size and tries to find a solution to this one first. Can you direct the canals (by transforming the lines into arrows), such that exactly half of the sites have an odd number of canals leading away?




As Binna gave you a copy, you are allowed to directly draw the directions in this picture. Remember to hand it in at the end though.

## Subtask 2: Odd Number of sites? (5 points)

While you were solving the example, Mouse Binna looked at some more layouts, which have $S$ sites and $M$ canals. She noticed that if the number of canals leaving a site is a multiple of 4 plus 1, the number of sites is always even. Can you help Mouse Binna explain why this is always the case?

## Subtask 3: Just Odd is Impossible (10 points)

In a Soian folk tale, multiples of 4 plus 3 are also considered lucky, but mixing them with multiple of 4 plus 1 brings bad luck. Mouse Binna wants to show the value of this tale.

She wants to construct a layout of a canal system fulfilling the following two properties:

a) the number of canals on each site is a multiple of 4 plus 1 or multiples of 4 plus 3, in other words just odd.

b) it is impossible to direct the canals such that from exactly $N$ sites an odd number of canals lead away (like in the Soian legend).

Can you help her find such an example?

## Subtask 4: Find an algorithm (75 points)

Help Binna find an algorithm to direct the canals, given a layout of the ruins.

In this subtask, the main focus is the correctness of your algorithm.

Your solution can score up to 55 points, if it runs in polynomial time for any polynomial. By polynomial we mean something like $O(n^{100})$ as opposed to $O(2^n)$.

Additionally, you can score up to 35 points by solving a special type of layouts: Layouts with $2 \cdot N - 1$ canals and where every site can be reached from every other site via canals before directing them.

Note: If you attempt them both, the polynomial and the special type one, then you only get the one with more points, and not the sum of both.