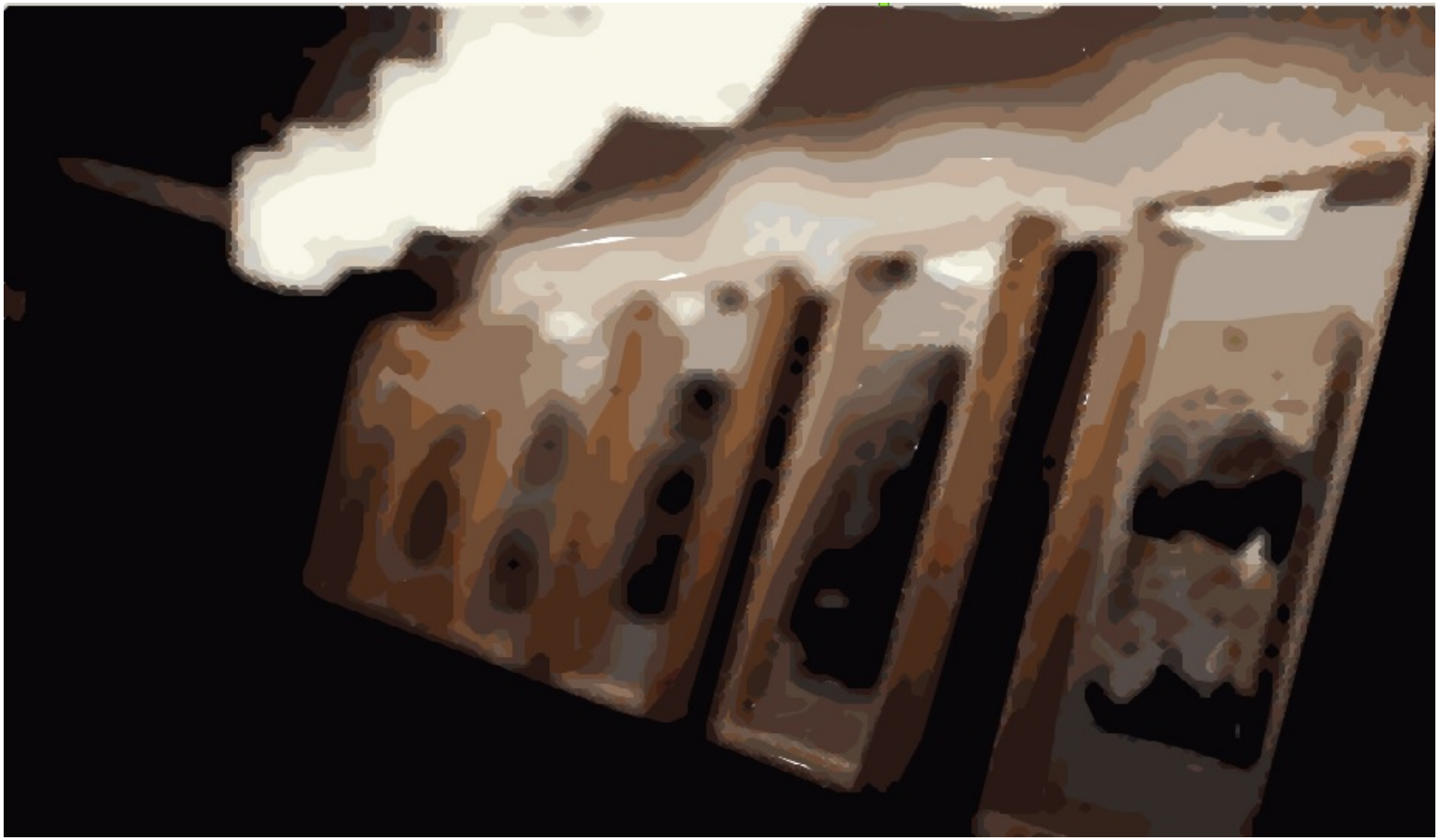
Kaggle Challenge Week (KCW)
The Optimized Exhibition Opening
— EPITA — MSc

R. DEHAK,
L. AVANTHEY and L. BEAUDOIN

M. ANGOUSTURES (server)

Thanks to R. ERRA and A. LETOIS

May 2025



Your Goal :

1. You are working in an **art gallery** and tonight, you have an **exhibition** opening.
2. The exhibition is made up of **4 showrooms**, each presenting a collection of **paintings** arranged sequentially in a long corridor
3. This exhibition is very special, the attendees are **robots** and have very specific tastes.
4. You are in charge of ***ordering the paintings*** in the 4 showroom where the robots will look at them.
5. To **satisfy** them, you must follow some **basic rules** specified in the following slides which will allow you to provide them with a **pleasant experience** balancing **continuity** with **surprises**.

Good Luck !

2. Problem description
   - Robotic visualization
   - Frames

Robots ?

Robots are integrating each painting with a *simple* ***list of tags***. Each tag represents an element of the painting. For example, if a robot sees the painting *Mona Lisa* it will integrate it as a list of tags : [woman, smile].

Robots also separate two types of paintings :

- **Landscape** for an horizontal painting
- **Portrait** for a vertical painting

Landscape

Portrait

2. Problem description
   - Robotic visualization
   - Frames

Frameglasses

To better suit to robots preferences, paintings are placed behind a frameglass. Each frameglass contains **one** Landscape painting or **two** Portrait paintings.

Just like paintings, robots are integrating each frameglass with a list of tags.

- **In a case of a Landscape :** the list of tags is the same than the list of the painting tags.
- **In a case of two Portraits :** the list of tags of the frameglass is the list of all the tags present in any or both of the two paintings it contains.

Each tag in a list is only counted **once** !

3. Input data set
   - Input description
   - Input example

The list of paintings (ID, tags and orientation) to be ordered in each showroom is described in a plain text file (only ASCII characters) :

- The first line contains one integer N ($1 \leq N \leq 10^5$) : the number of paintings for the showroom.
- It is followed by N lines : line i contains a description of the painting with an ID i ($0 \leq i < N$). The description of a painting i contains the following data, separated by a single space :
  1. A single character 'L' if the painting is horizontal (Landscape), or 'P' if it is vertical (Portrait).
  2. An integer M_i ($1 \leq M_i \leq 100$) : the number of tags for this painting.
  3. M_i text strings : the tags for the painting i.
- All lines are terminated by a single '\n' character (UNIX-style line endings).
- Each tag consists only of lowercase ASCII letters and digits, between 1 and 10 characters (max). The order of those tags is not important.

Example

animals, fear, war

smile, woman

woman, pearl

fear, raft, survivors

Input file	Description
4	The exhibition has 4 paintings
L 3 animals fear war	Painting 0 is a Landscape and has tags [animals, fear, war]
P 2 smile woman	Painting 1 is a Portrait and has tags [smile, woman]
P 2 woman pearl	Painting 2 is a Portrait and has tags [woman, pearl]
L 3 fear raft survivors	Painting 3 is a Landscape and has tags [fear, raft, survivors]

0_example.txt

```
4
L 3 animals fear war
P 2 smile woman
P 2 woman Pearl
L 3 fear raft survivors
```

List of frameglasses

1. We have **only 2 Portrait Painting**. So we put them in 1 frameglass {1, 2} with tags : `smile woman Pearl`
2. We have 2 Landscape Paintings. We built 2 frameglasses :
   - First frameglass contains the first painting {0} with tags : `animals fear war`
   - Second frameglass contains the fourth painting {3} with tags : `fear raft survivors`

4. Painting ordering
   - Frameglasses composition
   - Example of ordering for Showroom "Example"

0_example.txt

```
4
L 3 animals fear war
P 2 smile woman
P 2 woman Pearl
L 3 fear raft survivors
```

List of frameglasses : 3 frameglasses

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

Satisfaction

The good part with having robots attendees is that their **satisfaction** is very easy to determine : it is simply a *numerical value*, an **integer** ! We will call this scoring value the **Global Robotic Satisfaction** for a showroom.

The score is based on how interesting the **transitions** between each pair of subsequent (neighboring) frameglass are :

- Robots like when the transitions have something in common to preserve continuity (two consecutive frameglass should not be totally different),
- but they also want them to be different enough to stay interested.

The similarity of two Portrait paintings on a single frameglass is not taken into account for the score. For example, if a frameglass contains two Portraits with [smile, woman] and [woman, pearl], then the tags of this frameglass will be [smile, woman, pearl].

5 Robotic satisfaction
- Robotic's taste
- Calculating the satisfaction score
- Score calculating for Showroom "Example"

The Local **Robotic Satisfaction**

For two subsequent frameglass F_i and F_{i+1}, the **Local Robotic Satisfaction** is the **minimum** (the smallest number of the three) of the following three (3) integers :

1. The number of common tags between F_i and F_{i+1}
2. The number of tags in F_i but not in F_{i+1}
3. The number of tags in F_{i+1} but not in F_i.

The **Global Robotic Satisfaction**

For a full set of ordered paintings the *Final Score* is the **Global Robotic Satisfaction**, defined as the sum of all **Local Robotic Satisfactions**. Your goal is to maximize it ! *Remark : the code of the computation of the* **Global Robotic Satisfaction** *will be given on Wednesday morning : score_checker.py*

5 Robotic satisfaction
- Robotic's taste
- Calculating the satisfaction score
- Score calculating for Showroom "Example"

```
4
L 3 animals fear war
P 2 smile woman
P 2 woman Pearl
L 3 fear raft survivors
```

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

Six different possible orders for the 3 frameglasses : 6 possible solutions

1. {1, 2} ; {0} ; {3}
2. {1, 2} ; {3} ; {0}
3. {0} ; {1, 2} ; {3}
4. {0} ; {3} ; {1, 2}
5. {3} ; {1, 2} ; {0}
6. {3} ; {0} ; {1, 2}

Which is the best ?

Compute the Robotic satisfaction for each order

First Order : {1, 2} ; {0} ; {3}

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

Frameglass {1,2}

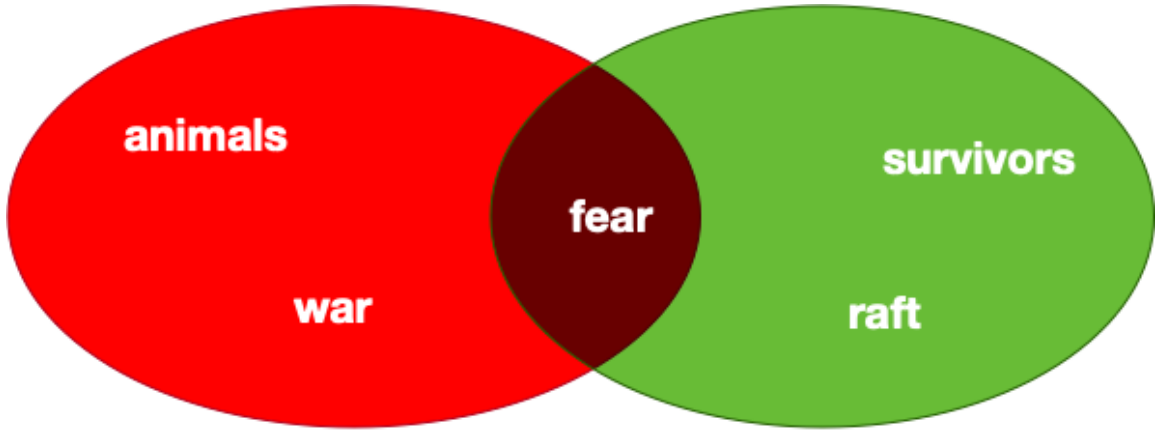
Frameglass {0}

- The number of common tags is 0
- The number of tags in `{1, 2]` but not in `{0}` is 3 : `smile woman Pearl`
- The number of tags in `{0}` but not in `{1, 2]` is 3 : `animals fear war`

The Robotic satisfaction for the transition `{1, 2}` ; `{0}` is

$$\min(0; 3; 3) = 0$$

First Order : {1, 2} ; {0} ; {3}




```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

Frameglass {0} **Frameglass {3}**

- The number of common tags is 1 : `fear`
- The number of tags in `{0]` but not in `{3}` is 2 : `animals war`
- The number of tags in `{3}` but not in `{0]` is 2 : `raft survivors`

The Robotic satisfaction for the transition `{0}` ; `{3}` is

$$\min(1; 2; 2) = 1$$

First Order : {1, 2} ; {0} ; {3}

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

- The Robotic satisfaction for the transition {1, 2} ; {0} is $\min(0; 3; 3) = 0$
- The Robotic satisfaction for the transition {0} ; {3} is $\min(1; 2; 2) = 1$

The Robotic satisfaction for the order {1, 2} ; {0} ; {3} is

$$0 + 1 = 1$$

Second Order : {1, 2} ; {3} ; {0}

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

Frameglass

Frameglass {3}

- The number of common tags is 0
- The number of tags in `{1, 2]` but not in `{3}` is 3 : `smile woman Pearl`
- The number of tags in `{3}` but not in `{1, 2]` is 3 : `fear raft survivors`

The Robotic satisfaction for the transition `{1, 2}` ; `{3}` is

$$\min(0; 3; 3) = 0$$

Second Order : {1, 2} ; {3} ; {0}

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

- The Robotic satisfaction for the transition {1, 2} ; {3} is $\min(0; 3; 3) = 0$
- The Robotic satisfaction for the transition {3} ; {0} is $\min(1; 2; 2) = 1$

The Robotic satisfaction for the order {1, 2} ; {3} ; {0} is

$$0 + 1 = 1$$

```
4
L 3 animals fear war
P 2 smile woman
P 2 woman Pearl
L 3 fear raft survivors
```

```
{1, 2}: smile woman Pearl
{0}: animals fear war
{3}: fear raft survivors
```

Six different possible orders for the 3 frameglasses : 6 possible solutions

1. {1, 2} ; {0} ; {3} : **1**
2. {1, 2} ; {3} ; {0} : **1**
3. {0} ; {1, 2} ; {3} : **0**
4. {0} ; {3} ; {1, 2} : **1**
5. {3} ; {1, 2} ; {0} : **0**
6. {3} ; {0} ; {1, 2} : **1**

Which is the best ?

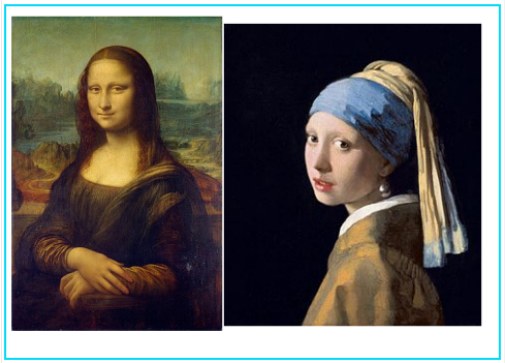
Select an order with a score of **1**

The output file must start with a **single integer** F $(1 \leq F \leq N)$: the number of frameglass in the exhibition. This must be followed by **F lines** describing each frameglass. Each line should contain either :

- A **single integer** : ID of the single Landscape painting in the frameglass.
- **Two integers** separated by a single space : IDs of the two Portrait paintings in the frameglass (in any order).

Occurrence

Each painting can be used **only one time or not at all**.



Frameglass {1, 2}

Frameglass {0}

Frameglass {3}

Submission file :

```
3
1 2
0
3
```

3	The exhibition has 3 frameglass
1 2	The first frameglass contains the portrait paintings 1 and 2
0	The second frameglass contains the landscape painting 0
3	The third frameglass contains the landscape painting 3

The score of this exhibition is 1

8 To know before starting
- Time and space constraints
- Optimization
- Terms and conditions

Time and space constraints

Let us imagine you have a "classical" laptop : icore I7 with 8GB of RAM, then :

1. For an input file of one (1) MB the execution time has to be less than 1mn
2. For an input file of one (2) MB the execution time has to be less than 2mn
3. ...
4. For an input file of $n > 5$ MB the execution time has to be less than 10mn
5. All executions can be done with a 8GB RAM computer and **must be finished at max in an 30 minutes for all input file**.

8 To know before starting
- Time and space constraints
- Optimization
- Terms and conditions

Some hints

1. The exhibition is tonight ! We need your help !
2. We don't have much time to think for the best ordering, so the computation need to be fast.
3. Begin with a simple solution, once it works . . .
4. . . .try to optimize it so it won't take too much time to execute, or it will be too late.
5. Use functions in your code !
6. Write comments if you want but be concise ! (It is a challenge).
7. Use the **score checker file** or submission website, it will help you to evaluate quickly your strategies & tactics.
8. **FORGET THE BRUTE FORCE ATTACK** !

Team work organisation

- Try to solve this problem with the members of your team.
- Python libraries (numpy, panda, ...) make a lot of stuff, try to find the best library and the best function before starting coding.
- Distribute the whole tasks among your team members to make fast progress (using functions in your code).

8 To know before starting
- Time and space constraints
- Optimization
- Terms and conditions

What we expect from you at the end

- 1 Python script, called **KCW_Team_<your_team_number>.py** that takes one input file and gives as output the adequate submission file.
- We will suppose that all input files will be in a directory called "Data".
- 5 submission files, one for each input file.
- Try to Optimize the output files to get the biggest score.
- *Remark : to get the real "optimal" (maximal) score is probably an untractable problem, so think "approximation".*

Your Final Score for the "ranking"

1. We will compute the score for each of the 5 files
2. We will compute the sum of these 5 scores
3. This will be your **final score** for the "ranking"
4. Don't forget : think simple, begin with the "simplest" solution
5. Don't try to find first a too complex algorithm
6. Try to begin with a "very" simple solution
7. And only after try to find better strategies/tactics
8. Think "step by step", test different strategies/tactics

Top score

1. You can access a site where you can upload your files and compare your scores with the other teams
2. The site address is `http://challenge.dehak.org`
3. This website shows the different scores of each team for each uploaded file and the final score
4. The score is updated live
5. The maximum size of a file is 10Mo
6. You can upload one or multiple files at the same time
7. You can do only one upload each 5min

Presentation of your results

1. You will do a resume (at most one page A4, 11pt) in pdf, doc, docx, odt or txt that will describe your strategies and tactics, you can also use a jupyter notebook.
2. Your (main) python file will be named **KCW_Team_<your team number>.py**
3. You will write a python file, named **KCW_final_score_<your team number>.py** that will :
   - Run your main python file on each of the 5 given input files
   - Save all the outputs (on different files)
   - Print for each file the global score computed with the help of the scoring python file you have been given
   - Compute the sum of these 5 scores and print it, clearly.
4. The teams with the five highest scores (on Wednesday, May 21th, 12h00) will give a talk on Friday, May 23th.

Where to sent your work

An assignment will be created on Teams, you need to send your work there. You will create one zip file named Team_XX where XX is your team number. This zip file must contain :

1. Your **two** python codes
2. Your report file
3. **Mandatory** : add **all** your Family and First names as a commentary at the beginning of both your python code and your report file
4. **Mandatory** : add YOUR TEAM NUMBER in your python file name and in your report file name
5. **Only one** person in each group send the work for all the group

Get the best score you can !

. . .**Congratulations and** *bon courage* **to all of you.**

9 Week Program
- Planning

Planning

1. Monday 12/05 10h-13h30 : Challenge presentation – **Kick-Off**
2. Monday 12/05 14h30-17h30 : Session 1 – **Input parsing & First tests**
3. Tuesday 13/05 9h-13h : Session 2 – **Optimize data representation**.
4. Wednesday 14/05 9h-12h : Team work / 13h-17h : Session 3 – **Problem analysis**.
5. Thursday 15/05 9h-13h : Team work
6. Friday 16/05 9h-12h : Team Work / 13h-17h : Session 4 – **Order optimization**.
7. Monday 19/05 10h-12h : Team Work / 13h-17h : Session 5 – **Advanced representation and Graph algorithms**
8. Wednesday 21/05 13h : deadline for the program's submission.
9. Wednesday 21/05 afternoon : List of oral presentations.
10. Friday 23/05 13h : deadline to upload the report or the slides.
11. Friday 23/05 14h-18h : Kaggle Week **Defenses**

First session : Input parsing and first tests

Write functions for each following item :

1. Read and parse input file.
2. Create frameglasses from the input data
3. Order the frameglasses
    1. Using same order.
    2. Using reverse order.
    3. Using random order.
    4. Ordered according to the number of tags of the frameglasses.
4. Write output file.
5. Add the execution time evaluation function of these different strategies to your main program.
6. Write the scoring function.
7. Submit first results for each showroom

First session : Input parsing and first tests

Oral defense (5 minutes) :

- Present the different strategies to parse and represent the input file (Pros and cons).
- Present the structure of your program (reading input, generate output order and generate output file).
- Analyse the four stategies to generate the output order (satisfaction score and execution time)