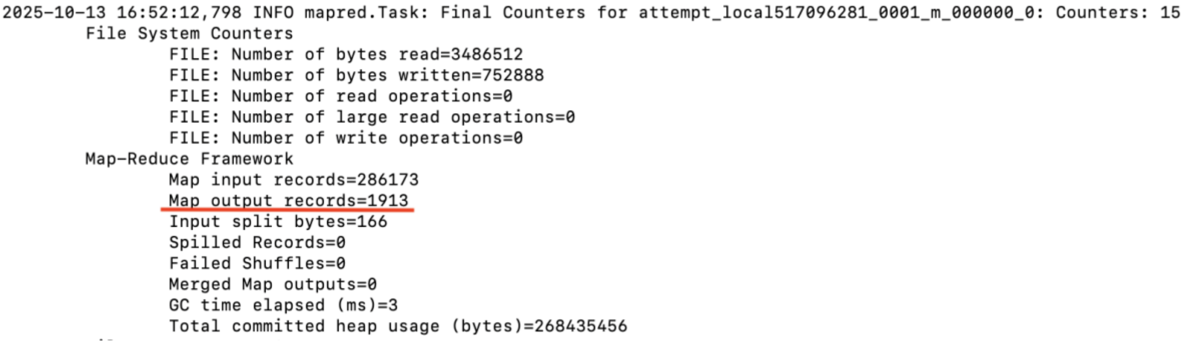
Lab 2

In this lab, you will write by your own a complete Hadoop application. Start by importing the template project available in Lab2_Skeleton.zip. Once you have imported the template, modify the content of the classes to implement the application described in the following. The template contains the skeleton of a standard MapReduce application based on three classes: Driver, Mapper, and Reducer. Analyze the problem specification and **decide if you really need all classes** to solve the assigned problem.

From now on, keep in mind the complexity of your program (even if we do not explicitly ask for it). Specifically, try to understand the effort the cluster makes in terms of network and I/O.

- How many pairs and bytes are emitted by the mappers, and hence how much data is sent on the network?

You can find this information in your application output by checking the “Map output records” counter (pay attention that, based on the number of input blocks and files, there is one “Map output records” value for each instance of the mapper class. You must sum all the “Map output records” values to evaluate to total number of pairs emitted by the map phase):



```
2025-10-13 16:52:12,798 INFO mapred.Task: Final Counters for attempt_local517096281_0001_m_000000_0: Counters: 15
        File System Counters
                FILE: Number of bytes read=3486512
                FILE: Number of bytes written=752888
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
        Map-Reduce Framework
                Map input records=286173
                Map output records=1913
                Input split bytes=166
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=3
                Total committed heap usage (bytes)=268435456
```

1. Filter an input dataset

If you completed Lab 1, you should now have (at least one) large files with the word frequencies in the Amazon food reviews, in the format `word\tnumber`, where number is an integer (a copy of the output of Lab 1 is available in the shared folder /share/students/bigdata/Dati/Lab2/OutputFolderLab1). You should also have realized that inspecting these results manually is not feasible. Your task is to write a Hadoop application to filter the content of the output of Lab 1 and analyze the filtered data.
The filter you should implement is the following:

- Keep only the lines containing words that start with “ho”

Store the selected lines (word\tnumber) in the output folder.

How large is the result of this filter? Do you need to filter more?

Modify the application to accept the beginning string (i.e. the prefix) as a command-line parameter.

Execute the new version of the program to select the words starting with the prefix that you prefer.

2. Filter and count

Extend the previous application.

The new version of your application must:

1. As in the previous application, select only the lines that start with the provided prefix as a parameter and store them in the output folder.
2. (New part of the problem specification) **Print on the standard output of the Driver** the number of selected words and the number of discarded words.

Bonus task

If you completed the bonus task of lab 1, try your filter on the 2-grams you have generated. If you did not complete the bonus task of Lab 1, you can use the files available in the shared folder /share/students/bigdata/Dati/Lab2/OutputFolderLab1BonusTrack

What is the size of this new input dataset, compared to the simple word counts (1-grams) we used in the previous step? Did you really need the cluster to filter 1-grams? What about 2-grams?

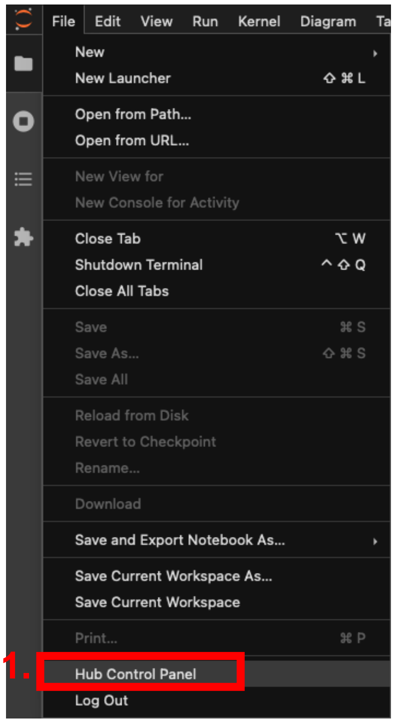
Implement a new application that selects all the 2-grams that contain, at any position, the word “like” (i.e., “like” can be either the first or the second word of the selected 2-grams).

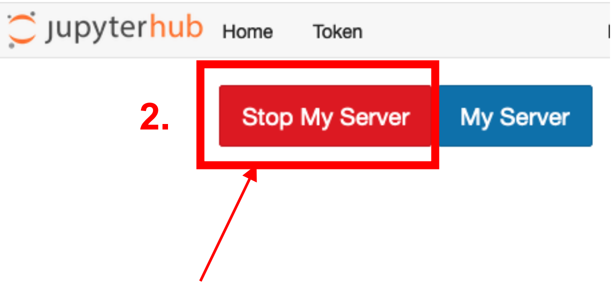
What do you think will be, most likely, the other word?

⚠⚠⚠⚠⚠⚠ Shut down JupyterHub container

As soon as you complete all the tasks and activities on JupyterHub environment, please remember to shut down the container to let all your colleagues in all the sessions connect on JupyterHub and do all the lab activities.

1. Go into File -> Hub Control Panel menu
2. A new browser tab opens with the “Stop My Server” button. Click on it and wait till it disappears.





Click the “Stop My Server” button