Algo Rush

HotRace

42 staff staff@42.fr

Summary:
A fast-paced competition on effective programming algorithms and techniques

Contents

Chapter I

Foreword

Throughout history, the human being has gradually emerged from their burdens which hampered the full development their fundamental character, and the development of their fundamental cultural identity.

Thus, in the year 1524 of our era, the robust and courageous sign, itself fixed on a robust and courageous support, bearing these words:
"Ne fermez pas la porte, le blount s'en chargera."

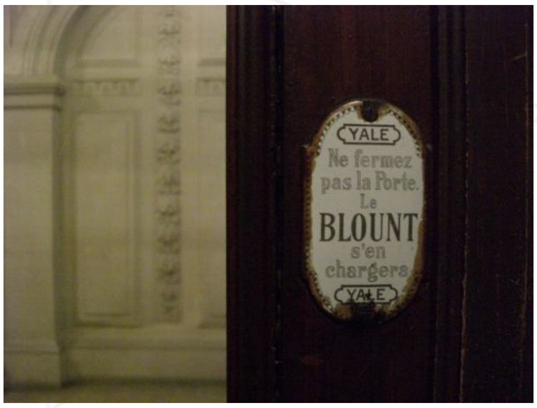

Figure I.1: Robust and Courageous writing

Chapter II

Subject

II.1 Mandatory Requirements

Finally, it isn't very complicated to make a search engine like *Google*. All you have to do is get a lot of into, then just search for key words inside. No, the difficulty is make it quickly. Their was a fierce struggle between *google*, *yahoo*, *bing*, and a whole bunch of others that hardly existed. It's a race where everyone has a fire on their ass.

The **HotRace** Everyone should take part in the **HotRace**.

To do this, you just have to create a program that will do the following:
Step 1
Store a large amount of information in the "keyword = value" form.
Step 2
Create a search query of this information, by entering a keyword

- Your program is called **hotrace**.
- Your program does not take parameters.
- Your program reads from the standard input.
- Your program reads the element to search from the standard input.
- Your program displays the value for each search follow by a '\n'.

The format for the standard input information is as follows :

```
tag-1
value1
tag-2
value2
[....]
keyword-x
valuex

keyword-search1
keyword-search2
[...]
```

Note the line break that send the search information.

The search may fail, then the following message will appear :

```
keyword-search Not found.
```

Replace "keyword-search" with the keyword that has been searched.

Among the programs that work correctly, the fastest will be rewarded 125 points. The next 120, and so on.

That validation threshold is a 1.
Your performance determines the points.
The speed, and thus the optimization of your program is crucial.
For this situation, 5 points between each team is an example.
The range is set to vary to as to distribute the points in a homogeneous way.
This will ensure real stakes during the competition.

P.S. : If you think it's complicated, wait until you have to write in as a shell script

II.2 Submitting the project

- You must have, at the root of your repo, an `author` file contaning the login of each group member. followed by a '\n':

```
$>cat -e auteur
xlogin$
ylogin$
$>
```

- You must have a `Makefile` with the usual rules. if in doubt, put in all the rules you know.
- Only the content that is present on your repo will be evaluated.

II.3 Permitted Functions

- read
- write
- malloc
- free
- strerror
- compiler directive `__asm__`

Chapter III

Instructions

- A moulinette is available. Use it. It will help you rate your program.
- The results of the moulinette will be used to grade you.
- Your project must be in C and to the standard Norminette.
- Your program should not quit unexpectedly (Segmentation fault, bus error, double free, etc).
- Any memory allocated on the job must be cleaned properly.
- You cannot use your library `libft`.

Good luck to everyone for this rush!