Rush

HotRace

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

Version: 2

Contents

Chapter I

Preamble

Throughout history, the human being gradually overcame the limitations which hampered the full development their fundamental character, and the development of their fundamental cultural identity.

Thus, in 1524, a robust and courageous sign, fixed on a robust and courageous plaque, was hung bearing these words:
"Don't shut the door, the Blount will take care of it."

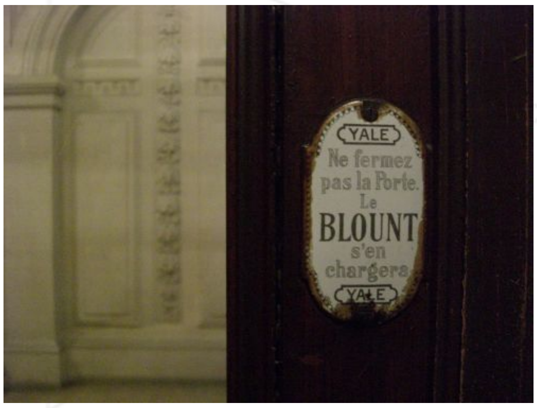

Figure I.1: The robust and courageous plaque

Chapter II

General rules

You can organize and name your files as you wish but your assignment has to comply with the following rules.

- Your assignment must be written in `C`.
- Your assignment must comply with the `Norm`.
- Place all your files at the root of your repository.
- You have to compile your program using: `cc`
- You have to compile your program with the following flags: `-Wall -Wextra -Werror`
- You have to turn in a `Makefile` which will compile your source files. It must not relink.
- Your `Makefile` must at least contain the rules: `NAME`, `all`, `clean`, `fclean` and `re`.
- Your functions should not quit unexpectedly (segmentation fault, bus error, double free, and so forth) except for undefined behaviors. If it happens, your project will be considered non-functional and your grade will be `0`.
- You cannot use your `libft`.
- You are allowed to use the following functions:
    - `read`
    - `write`
    - `malloc`
    - `free`
    - `strerror`
    - The compiler directive: `__asm__`
- Your grade will be based on the speed, and thus the optimization of your program.

Chapter III

Project instructions

Making a search engine like *Google* or *Bing*, is not so complicated in the end. The steps are simple: get a lof of data, then just perform search using keywords.

All the difficulty lies in returning the responses quickly. There is fierce competition in the search industry between *Google*, *Yahoo*, *Bing* and a whole bunch of other search engines that hardly existed. It's a race where everyone is on the hot seat.

- Name your program: **hotrace**.
- It takes no parameters.
- The first step is to store data as values associated to keys. The next one is to search them using keywords.
  - First, the program reads on the standard input the keyword to store and, on the next line, its associated value.
  - Then, it reads on the standard input the keyword to search and displays the result followed by a `\n` character.

The information are displayed as follows:

```
$>cat -e example.htr
keyword-1$
value-1$
keyword-2$
value-2$
$
keyword-1$
keyword-2$
$>./hotrace < example.htr
value-1
value-2
$>./hotrace
keyword-1
value-1
keyword-2
value-2

keyword-3
keyword-3: Not found.
keyword-1
value-1
```

```
...
$>
```

Note the line break that separate the data from the searches.

If a search fails, the following message is displayed:

```
keyword-searched: Not found.
```

Replace “keyword-searched” with the keyword that was searched.

Good luck to everyone for this Rush!

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

Chapter IV

Submission

IV.1 Guidelines

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated. Don't hesitate to double check the names of your files to ensure they are correct.

You are encouraged to create test programs for your project, but you must not **submit them**.