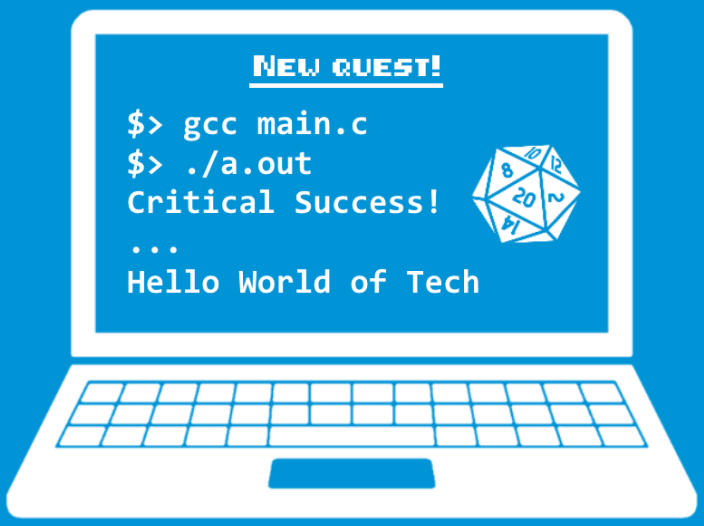
{EPITECH}

RUSH2

WHAT LANGUAGE IS THIS?



RUSH2

Preliminaries

Language: C
Binary name: rush2

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Segfault, bus error, floating exception are all grounds for disqualification!

All .c files from your delivery folder will be collected and compiled with your `libmy`, which must be found in `lib/my/`. For those of you using .h files, they must be located in `include/` (like the `my.h` file).

Allowed system function(s): `write`

Your library will be built using the `lib/my/build.sh` script you previously made (see Day07).

Terminal

```
~/B-CPE-100> cd lib/my && ./build.sh && cd ../..
~/B-CPE-100> cc -o rush2 *.c -I./include -L./lib/my -lmy
```

Don't forget to write unit tests for all your functions!

Team work

Here are a couple of important rules to follow:

- ✓ The **team leader** must register his/her group for the review on the intranet.
- ✓ You must complete your review on Sunday, at the right time, and **with all of your partners**.
- ✓ Every member of the group should be fully aware of the work you will have completed. Each member will be questioned, and your group's grade will be based on the worst explanations.
- ✓ You have to do everything within your power to contact your partners; look at their intranet profile, find them on Facebook or by any other mean.
  Excuses regarding group problems will not be accepted.
  If, after you have tried **everything**, and your partner is still unreachable, send an email to the local staff **ASAP**.

What language is this?

The objective of this rush is to code a program that automatically guesses the language in which a text is written.
It is divided into 4 progressive steps.

Don't start a step if you are not 100% sure to fully complete the previous steps!
Use descent and exhaustive unit tests to guarantee it…

For this whole projet, only the characters from 'a' to 'z' and from 'A' to 'Z' are to be taken into account (no accent, no punctuation whatsoever,…).

Step 1 - Counting the number of occurences of a letter

For this first step, your program must display the number of occurences of a letter (Upper or lower case).
The program must handle 2 arguments and work as follows:

```
Terminal
~/B-CPE-100> ./rush2 “Just because I don’t care doesn’t mean I don’t understand!” a
a:4
~/B-CPE-100> ./rush2 “Just because I don’t care doesn’t mean I don’t understand!” A
A:4
~/B-CPE-100> ./rush2 “Just because I don’t care doesn’t mean I don’t understand!” j
j:1
```

Respect **precisely** the formatting of this output!

Step 2 - Counting the number of occurences of several letters

Now your program must display the number of occurences of several letters (one time, upper or lower case).
The program must be displayed in the following way:

```
Terminal
~/B-CPE-100> ./rush2 “I hope I didn’t brain my damage.” a i
a:3
i:4
~/B-CPE-100> ./rush2 “I hope I didn’t brain my damage.” A i I i
A:3
i:4
I:4
i:4
```

Step 3 - Frequencies

Add in the output of your program the frequency of each letter.
The program must be displayed in the following way:

```
Terminal
~/B-CPE-100> ./rush2 "Good things do not end with 'ium'. They end with 'mania' or
'teria'." a q e i | cat -e
a:3 (6.25%)$
q:0 (0.00%)$
e:4 (8.33%)$
i:6 (12.50%)$
```

Display percentages with 2 figures after the decimal point.

Step 4 - Language

Have a look at this array of letters frequencies for several languages.
Using this array and an algorithm of your own, display at the end of your output the language in which you estimate the text was written:

```
Terminal
~/B-CPE-100> ./rush2 "Good things do not end with 'ium'. They end with 'mania' or
'teria'." a q e i | cat -e
a:3 (6.25%)$
q:0 (0.00%)$
e:4 (8.33%)$
i:6 (12.50%)$
=> English
```

Only consider "English", "French", "German" and "Spanish".

This is a complex problem, and many cases are ambiguous. Get on with it!

Bonus

Why not customize your program and make it even more attractive? It could earn you some bonus points!

Every bonus you make must be placed in a subdirectory called `bonus`.

v 2.2