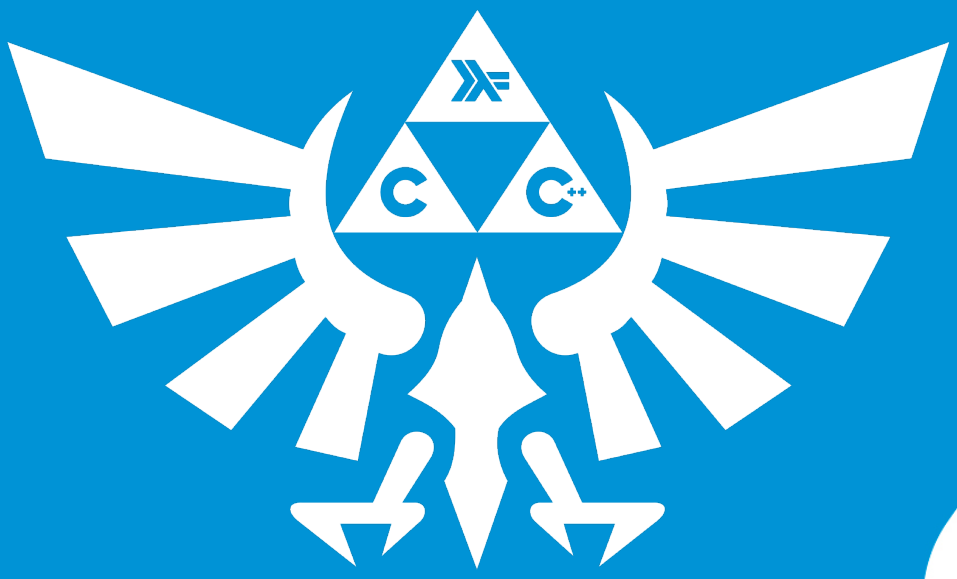
{EPITECH}

DAY 06

IOSTREAM, STRING AND OBJECTS



DAY 06

All your exercises will be compiled with `g++` **and the** `-std=c++20 -Wall -Wextra -Werror` **flags**, unless specified otherwise.

All output goes to the standard output, and must be ended by a newline, unless specified otherwise.

None of your files must contain a `main` function, unless specified otherwise. We will use our own `main` functions to compile and test your code. It will include your header files.

For each exercise, **the files must be turned-in at the root of your repository unless specified otherwise**.

Read the examples CAREFULLY. They might require things that weren't mentioned in the subject…

The `*alloc`, `free`, `*printf`, `open` and `fopen` functions, as well as the `using namespace` keyword, are forbidden in C++. By the way, `friend` is forbidden too, as well as any library except the standard one.

Unit Tests

It is highly recommended to test your functions as you implement them. It is common practice to create and use what are called **unit tests**.

From now on, we expect you to write unit tests for your functions (when possible). To do so, please follow the instructions in the **“How to write Unit Tests”** document on the intranet, available here.

For them to be executed and evaluated, put a `Makefile` at the root of your directory with the `tests_run` rule as mentionned in the documentation linked above.

Here is a sample set of unit tests for the **string** class :

```
#include <criterion/criterion.h>

Test(string, default_value)
{
    std::string s;
    cr_assert_eq(s, "");
}

Test(string, assign)
{
    std::string s;

    s = "test";
    cr_assert_eq(s, "test");
}

Test(string, append)
{
    std::string s("test");

    s += "ing";
    cr_assert_eq(s, "testing");
}
```

Exercise 0 - Z is (still) for Zorglub

Turn in : `Makefile`, `Z.cpp` in `ex00/`
Makefile rules : `all`, `clean`, `fclean`, `re`
Executable name : `Z` (uppercase 'Z')

Our Lord and Genuis Zorglub, master of Zorgland, requires your services. Your brillant zorgmanizer character selection software must be converted to a much more evoluted programming language. Rewrite your program in C++.

As we use the CamelCase naming convention in C++, be careful to correctly name your source files and binary.

Terminal

```
~/B-PDG-300> ~/B-PDG-300> ./Z ''0x42242112'' | cat -e
z$
~/B-PDG-300> ./Z ''invalid_ID'' ; echo $?
z
0
```