☰   🔍 (https://profile.intra.42.fr/searches)   **rmartins**

(https://profile.intra.42.fr)

Remember that the quality of the defenses, hence the quality of the of the school on the labor market depends on you. The remote defences during the Covid crisis allows more flexibility so you can progress into your curriculum, but also brings more risks of cheat, injustice, laziness, that will harm everyone's skills development. We do count on your maturity and wisdom during these remote defenses for the benefits of the entire community.

SCALE FOR PROJECT CPP MODULE 02 (/PROJECTS/CPP-MODULE-02)

You should evaluate 1 student in this team

★

Git repository

`git@vogsphere.42lisboa.com:vogsphere/intra-uuid-af7ec962-86`   📋

Introduction

- Only grade the work that is in the student or group's
GiT repository.

- Double-check that the GiT repository belongs to the student
or the group. Ensure that the work is for the relevant project
and also check that "git clone" is used in an empty folder.

- Check carefully that no malicious aliases were used to fool you
and make you evaluate something other than the content of the
official repository.

- To avoid any surprises, carefully check that both the evaluating
and the evaluated students have reviewed the possible scripts used
to facilitate the grading.

- If the evaluating student has not completed that particular
project yet, it is mandatory for this student to read the
entire subject prior to starting the defence.

- Use the flags available on this scale to signal an empty repository,
non-functioning program, a norm error, cheating etc. In these cases,
the grading is over and the final grade is 0 (or -42 in case of
cheating). However, with the exception of cheating, you are

encouraged to continue to discuss your work (even if you have not
finished it) in order to identify any issues that may have caused
this failure and avoid repeating the same mistake in the future.

- Remember that for the duration of the defence, no segfault,
no other unexpected, premature, uncontrolled or unexpected
termination of the program, else the final grade is 0. Use the
appropriate flag.
You should never have to edit any file except the configuration file if it exists.
If you want to edit a file, take the time to explicit the reasons with the
evaluated student and make sure both of you are okay with this.

- You must also verify the absence of memory leaks. Any memory allocated on the heap must
be properly freed before the end of execution.
You are allowed to use any of the different tools available on the computer, such as
leaks, valgrind, or e_fence. In case of memory leaks, tick the appropriate flag.

Disclaimer

Please respect the following rules:

- Remain polite, courteous, respectful and constructive
throughout the evaluation process. The well-being of the community
depends on it.

- Identify with the person (or the group) evaluated the eventual
dysfunctions of the work. Take the time to discuss
and debate the problems you have identified.

- You must consider that there might be some difference in how your
peers might have understood the project's instructions and the
scope of its functionalities. Always keep an open mind and grade
him/her as honestly as possible. The pedagogy is valid only and
only if peer-evaluation is conducted seriously.

Guidelines

You must compile with clang++, with -Wall -Wextra -Werror
As a reminder, this project is in C++98 and C++20 members functions or containers are NOT expected.

Any of these means you must not grade the exercise in question:
- A function is implemented in a header (except in a template)
- A Makefile compiles without flags and/or with something other than clang++

Any of these means that you must flag the project as Forbidden Function:
- Use of a "C" function (*alloc, *printf, free)
- Use of a function not allowed in the subject

- Use of "using namespace" or "friend"
- Use of an external library, or C++20 features
- Use of an already existing container, or any existing function, to implement another container

Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/25810/en.subject.pdf)

Exercise 00: My First Canonical

This exercice introduces the notion of canonical class with a simple arithmetic exemple: the fixed point numbers.

Makefile

There is a Makefile that compiles using the appropriate flags.

           ✓ Yes   ✕ No

Accessors

The Fixed class (or whatever its name) must provide accessors to
the raw value:
- int getRawBits(void) const;
- void setRawBits(
int const raw);
Are these member functions present and functional?

           ✓ Yes   ✕ No

Canonical

A canonical class must provide at least:
- A default constructor
- A destructor
- A copy constructor
- An assignation operator
Are these elements present and functional?

           ✓ Yes   ✕ No

Exercise 01: Towards a more useful fixed point class

Ex00 was a good start, but our class is still pretty useless being only able to represent the fixed point value 0.0.

Makefile

There is a Makefile that compiles using the appropriate flags.

    ☑ Yes   ✕ No

Floating point constructor

Is it possible to construct an instance from a floating point value?

    ☑ Yes   ✕ No

<< operator

Is there a << operator overload and is it functional?

    ☑ Yes   ✕ No

Fixed point value to integer value

A member function "int toInt(void) const;" that converts the
fixed point value to an integer value must be present. Is it functional?

    ☑ Yes   ✕ No

Fixed point value to floating point value

A member function \"float toFloat(void) const;\" that converts
the fixed point value to a float value must be present. Is it functional?

    ☑ Yes   ✕ No

Integer constructor

Is it possible to construct an instance from an integer value?

    ☑ Yes   ✕ No

Exercise 02: Now we're talking

This exercice add comparison and arithmetic features to the class.

Makefile

There is a Makefile that compiles using the appropriate flags.

⊘ Yes ✕ No

Division

The class must provide a division operator. Is it present and functional?

⊘ Yes ✕ No

Exercise 03: BSP

This exercice should have make you realize how easy it is to implement complex algorithms once the basics work as intended.

Makefile

There is a Makefile that compiles using the appropriate flags.

⊘ Yes ✕ No

Class Point

There is a class Point which has two attributes (x and y) of type Fixed const.
It also has a constructor that takes two floats and initialize x and y with those values.

⊘ Yes ✕ No

Function bsp

There is a function bsp which prototype is
"bool bsp(Point const a, Point const b, Point const c, Point const point)".
The function returns True if the point is inside of the triangle described by the vertices a, b, and c.
It returns False otherwise.

⊘ Yes ✕ No

Main and tests

There is a main to test that the function bsp works as metined above.
Run several test to make sure that the return value is correct.

⊘ Yes ✕ No

Ratings

Don't forget to check the flag corresponding to the defense

✔ Ok

🖿 Empty work   🖿⚠ Incomplete work   ☠ Invalid compilation   🗐 Cheat   ♔ Crash

⊘ Forbidden function

Conclusion

Leave a comment on this evaluation

```

```

<div style="border:1px solid #00B0C8; text-align:center; padding:1em">**Finish evaluation**</div>

Declaration on the use of cookies (https://signin.intra.42.fr/legal/terms/2)
Rules of procedure (https://signin.intra.42.fr/legal/terms/4)
Privacy policy (https://signin.intra.42.fr/legal/terms/5)
General term of use of the site (https://signin.intra.42.fr/legal/terms/6)
Terms of use for video surveillance (https://signin.intra.42.fr/legal/terms/1)
Legal notices (https://signin.intra.42.fr/legal/terms/3)