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

You should evaluate 1 student in this team

★

Git repository

git@vogsphere.kzn.21-school.ru:vogsphere/intra-uuid-cf91

Introduction

Please comply with 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 student or group whose work is evaluated the possible dysfunctions in their project. Take the time to discuss and debate the problems that may have been identified.

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

Guidelines

- Only grade the work that was turned in the Git repository of the evaluated student or group.

- Double-check that the Git repository belongs to the student(s). Ensure that the project is the one expected. Also, check that 'git clone' is used in an empty folder.

- Check carefully that no malicious aliases was used to fool you and make you evaluate something that is not the content of the official repository.

- To avoid any surprises and if applicable, review together any scripts used to facilitate the grading (scripts for testing or automation).

- If you have not completed the assignment you are going to evaluate, you have

to read the entire subject prior to starting the evaluation process.

- Use the available flags to report an empty repository, a non-functioning program, a Norm error, cheating, and so forth.
In these cases, the evaluation process ends and the final grade is 0, or -42 in case of cheating. However, except for cheating, student are strongly encouraged to review together the work that was turned in, in order to identify any mistakes that shouldn't be repeated in the future.

- 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.

Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/52150/en.subject.pdf)

Preliminary tests

If cheating is suspected, the evaluation stops here. Use the "Cheat" flag to report it. Take this decision calmly, wisely, and please, use this button with caution.

Prerequisites

The code must compile with c++ and the flags -Wall -Wextra -Werror
Don't forget this project has to follow the C++98 standard. Thus, C++11 (and later) 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 file (except for template functions).
- A Makefile compiles without the required flags and/or another compiler than c++.

Any of these means that you must flag the project with "Forbidden Function":
- Use of a "C" function (*alloc, *printf, free).
- Use of a function not allowed in the exercise guidelines.
- Use of "using namespace " or the "friend" keyword.
- Use of an external library, or features from versions other than C++98.

Yes No

Exercise 00: My First Class in Orthodox Canonical Form

This exercise introduces the notion of canonical class with a simple arithmetic example: 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 copy assignment operator
Are these elements present and functional?

☑ Yes ☒ No

Exercise 01: Towards a more useful fixed-point number class

The previous exercise was a good start, but the class was still pretty useless since it was 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 are talking

This exercise adds comparison and arithmetic features to the class.

Makefile

There is a Makefile that compiles using the appropriate flags.

Yes No

Comparison operators

Are the 6 comparison operators (>, <, >=, <=, == and !=) implemented and working properly?

Yes No

Arithmetic operators

Are the 4 arithmetic operators (+, -, * and /) implemented and working properly?

☑ Yes ☒ No

Other operators

Are the pre-increment, post-increment, pre-decrement and post-decrement operators implemented and working properly?

☑ Yes ☒ No

Static member functions overloads

Last but not least, test the the min() and max() static member functions are implemented and working properly.

☑ Yes ☒ No

Exercise 03: BSP

This exercise 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 the triangle described by the vertices a, b, and c.
It returns False otherwise.

☑ Yes ☒ No

Main and tests

There is at least a main to test that the function bsp() works as required.
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 ★ Outstanding project

Empty work Incomplete work Invalid compilation Cheat Crash

Concerning situation Leaks Forbidden function

Conclusion

Leave a comment on this evaluation

Finish evaluation