

Back to all evaluation sheets

CPP Module 02

You should evaluate 1 student in this team

Introduction

Please follow th	e rules	below:
------------------	---------	--------

\bigcirc	demain polite, courteous, respectful and constructive throughout the evaluation process.
The	vell-being of the community depends on it.

✓ Ider	ntify with the student	or group whose wo	ork is evaluated the	e possible dysfund	ctions in their
project.	Take the time to disc	uss and debate the	problems that ma	y have been ident	ified.

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

Please follow the guidelines below:

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

Please download the attachments below:



Mandatory Part

Comparison Operators

Comparison Operators

Are the six comparison operators (>, <, >=, <=, ==, and !=) present and functional?

Yes No

Fixed-Point to Float

Fixed-Point to Float

The class must include a member function float toFloat(void) const;

that converts a fixed-point number to a float.

Is it present and functional?

Yes No

Overloading of Public Static Member Functions

Overloading of Public Static Member Functions

Finally, check that the static member functions min() and max() are implemented and functional.

Yes No

Exercise 03: BSP

This exercise should demonstrate how easy it is to implement complex algorithms once the basics are functioning as expected.

Makefile

There is a Makefile that compiles using the appropriate flags.

Yes No

Canonical Class

Canonical Class

A canonical class must have at least:

- A default constructor
- A destructor
- A copy constructor
- An assignment operator

Are these elements present and functional?

Yes No

Preliminary tests

If a case of cheating is suspected, the grading and evaluation will immediately be terminated. To report it, select the 'Cheat' flag. Be careful to use it calmly, cautiously, and

with discernment. *Prerequisites* The code must compile with C++ and the flags -Wall -Wextra -Werror. As a reminder, this project must follow the C++98 standard. Therefore, C++11 (or other standards) functions and containers are NOT expected. Do not grade the exercise if you find: A function implemented in a header file (except for template functions). A Makefile that compiles without the required flags and/or with anything other than C++. Select the 'Forbidden function' flag if you encounter: The use of a "C" function (*alloc, *printf, free). The use of a function forbidden in the project. The use of "using namespace <ns_name>" or the keyword "friend". The use of an external library or features specific to versions later than C++98." Yes No **Fixed-Point to Integer**

Fixed-Point to Integer

The class must include a member function int toInt(void) const;

that converts a fixed-point number to an integer.

Is it present and functional?

Yes No

Arithmetic Operators

Arithmetic Operators

Are the four arithmetic operators (+ , - , * , and /) present and functional?

(If a division by zero occurs, it is acceptable for the program to crash.)

`<<` Operator		
`<<` Operator		
Is there a << operator, a	and is it functional?	
Yes	No	
Exercise 01: First S	teps Toward a Useful Class	
The previous exercise we could only represent the	as a good first step. However, the class was of little use since it value 0.0.	
Makefile		
There is a Makefile that	compiles using the appropriate flags.	
Yes	No	
Exercise 02: Now \	Ve Can Talk	
This exercise adds comp	parison and arithmetic operators to the class.	
Makefile		
There is a Makefile that	compiles using the appropriate flags.	
Yes	No	
`bsp` Function		
`bsp` Function		
There is a function bsp	() with the following prototype:	
```cpp		
bool bsp(Point const a, Point const b, Point const c, Point const point);		

No

Yes

The function returns True if the point is inside the triangle defined by the vertices a, b, and c. Otherwise, it returns False. No Yes **Other Operators** Other Operators Are the four increment and decrement operators (pre-increment, post-increment, predecrement, and post-decrement) present and functional? Yes No Accessors Accessors The Fixed class (or another class) must have accessors for the raw value: ```cpp int getRawBits(void) const; void setRawBits( int const raw ); Are these members present and functional?" Yes No

### **Constructor via Floating Point**

Constructor via Floating Point

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

Yes No

# Construction with an 'int' Is it possible to instantiate the class using the constructor that takes an int? Yes No **Point Class Point Class** There is a Point class that has two attributes (x and y) of type Fixed const. It also has a constructor that takes two floats and initializes x and y with these values. Yes No **Exercise 00: My First Canonical Class** This exercise introduces the concept of a canonical class with a simple arithmetic exercise: fixed-point numbers. *Makefile* There is a Makefile that compiles using the appropriate flags. Yes No **Bonus Part**

## Main and Tests

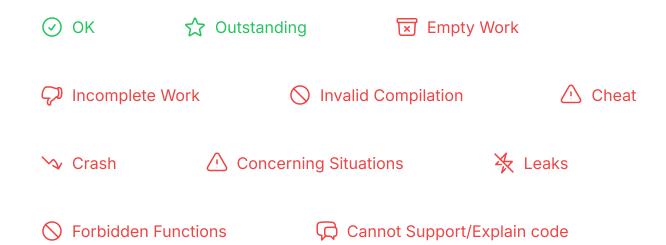
Construction with an 'int'

Main and Tests

There is a main function to test that the bsp() function works as described above. Run several tests to ensure that the return value is correct.

Yes No

## **Ratings**



© 2024 42evals. All rights reserved.