42 Evals

Back to all evaluation sheets

Points earned
0

CPP Module 06

You should evaluate 1 student in this team

Introduction

Please follow the rules below:

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

Work with the student or group being evaluated to identify potential issues in their project. Take time to discuss and debate the problems identified.

Understand that there may be differences in how peers interpret the project instructions and scope. Always keep an open mind and grade as honestly as possible. Pedagogy is effective only when peer evaluations are taken seriously.

Guidelines

Please follow the guidelines below:

Only grade the work submitted to the Git repository of the evaluated student or group.

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

Carefully verify that no malicious aliases are used to d
grading non-official content.

If applicable, review any scripts used for testing or aut
student.

If you haven't completed the assignment you're evaluating, read the entire subject before starting the evaluation.

Use the available flags to report an empty repository, a non-functioning program, a Norm error, or cheating. The evaluation process ends with a final grade of 0 (or -42 for cheating). However, except in cases of cheating, students are encouraged to review the work together to identify mistakes to avoid in the future.

Remember that no segfaults or other unexpected program terminations will be tolerated during the evaluation. If this occurs, the final grade is 0. Use the appropriate flag.

You should not need to edit any files except the configuration file, if it exists. If editing a file is necessary, explain the reasons to the evaluated student and ensure mutual agreement.

Verify the absence of memory leaks. All memory allocated on the heap must be properly freed before the program ends.

You may use tools like leaks, valgrind, or e_fence to check for memory leaks. If memory leaks are found, tick the appropriate flag.

Points earned

0

Attachments

Please download the attachments below:

subject.pdf

Mandatory Part

Prerequisites

Points earned
0

If cheating is suspected, the evaluation stops here. Use
it. Take this decision calmly, wisely, and please, use this

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 <ns_name>" or the "friend" keyword.
- Use of an external library, or features from versions other than C++98.

Yes No

Exercise 00: Conversion of scalar types

This exercise is about using the static_cast.

Scalar conversion

Did the student create a class with a private constructor and static methods ?

Did the student use the static_cast to convert values?

Accept the use of implicit casts for promotion casts only

Does the program work as required?

Points earned
0

Anyway, please don't be too uncompromising towards the exercise's outputs if the spirit of the exercise is respected.

Even if this exercise is wrong, continue the evaluation process.

Yes No

Exercise 01: Serialization

This exercise is about using the reinterpret_cast.

Retyping of raw data

Does the program work as required?

Did the student create a class with a private constructor, and static

methods ?

The reinterpret_cast<> should be used twice:

- First from data* to uintptr_t.
- Then, from uintptr_t to data*.

And the resulting data struct should be usable.

Yes No

Exercise 02: Identify real type

Points earned
0

This exercise is about using the dynamic_cast.

Real type identification

Does the program work as required?

Check the code. Did the student use the dynamic_cast to identify the real type?

void identify(Base* p) should check if the cast return is NULL.

void identify(Base& p) should use a try and catch block to check if the cast failed.

(In case you're wondering, the header <typeinfo> must not appear anywhere.)

Yes No

Bonus Part

no bonus

no bonus

no bonus

Yes No

Points earned
0

Ratings

OK Outstanding Empty Work

Incomplete Work Invalid Compilation

Cheat Crash Concerning Situations

Leaks Forbidden Functions