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 08

You should evaluate I student in this team -Q

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 O (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 O. 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 nag.

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 -\Nerror
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 nags and/or with something other than clang++

then once the file is fully read, it executes the Instructions

Any of these means that you must nag the proiect as Forbidden Function:

- Use of a 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

Attachments

subiect.pdf

exOO

exOO

As usual, there has to be a main function that contains enough tests to prove the program works as required. If there isn't, do not grade this exercise. If any non-interface class is not in Coplien's form, do not grade this exercise.

There is a templated function easyFind(T, int) that does what the subiect requires.
It HAS to use STL algorithms.
If it does not (Manual search using iterators for example), count it as wrong.

✓ Yes	X No

ex01

ex01

As usual, there has to be a main function that contains enough tests to prove the program works as required. If there isn't, do not grade this exercise. If any non-interface class is not in Coplien's form, do not grade this exercise.

There is a class that respects the constraints of the subiect.

Its member functions use STL algorithms to find their result, as much as possible.

U Yes	✕ No

Better addNumber

There's a way to add numbers thaYs more practical than calling addNumber repeatedly.

U Yes	✕ No

ex02

ex02

As usual, there has to be a main function that contains enough tests to prove the program works as required. If there isn't, do not grade this exercise. If any non-interface class is not in Coplien's form, do not grade this exercise.

There is a MutantStack class that inherits from (Or is composed of one, dealer's choice), and offers all of its member functions. It has an iterator, and it is possible to do at least the operations in the subiecys example with it.

✓ Yes	X No

Better tests

There is a test main() function that has more tests than the one in the subject.

U Yes	✕ No

ex03

ex03

As usual, there has to be a main function that contains enough tests to prove the program works as required. If there isn't, do not grade this exercise. If any non-interface class is not in Coplien's form, do not grade this exercise.

There is a program that can interpret a Brainfuck-like language.
It functions correctly (The student has to provide test files to prove it) It works in the way specified by the subiect, that is, it has a set of

Leave a commenton thisevaluation

Finish evaluation

Instruction classes that inherit from a common interface or class, it reads from the file, creating one such Instruction object and storing it in an appropriate container, It has to use STL containers and algorithms. If the way it works deviates too much from what the subject requires, count it as wrong.

U Yes   ✕ No

ex04

ex04

As usual, there has to be a main function that contains enough tests to prove the program works as required. If there isn't, do not grade this exercise. If any non-interface class is not in Coplien's form, do not grade this exercise.

The program works as the subject requires:
- First, it converts the expression to a set of Token-derived objects -It converts the expression to postfix (aka Reverse Polish) notation - It evaluates the expression while outputting every single step, as in the subiecfs example.
Errors are handled appropriately.
STL algorithms are used in a reasonable enough amount.

U Yes   ✕ No

Ratings

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

0k   Outstanding project

Empty work   No author file   Invalid compilation   Norme   Cheat

Crash   🩸 Leaks   @ Forbidden function

Conclusion