Go to homepage 🏠 (../../)

SCALE FOR PROJECT CPP MODULE 00 (HTTPS://PROJECTS.INTRA.42.FR/PROJECTS/CPP-MODULE-00)

You should evaluate 1 student in this team

Introduction

Please respect the following rules:

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

- Identify with the evaluated person or group any malfunctions in their work. Take the time to discuss and debate the identified problems.

- Keep in mind that there may be slight differences in interpretation between the project instructions, its scope, and functionalities. Keep an open mind and rate as honestly as possible. Pedagogy is only valid if peer evaluation is taken seriously.

Guidelines

- Only rate what is contained in the cloned Git repository of the student or group.

- Verify that the Git repository belongs to the student or group, that the project is the expected one, and that "git clone" is used in an empty folder.

- Thoroughly check that no alias has been used to deceive you and ensure that you are evaluating the official submission.

- To avoid any surprises, check with the student or group any potential scripts used to facilitate evaluation (e.g., test or automation scripts).

- If you have not done the project you are evaluating, you must read the subject entirely before starting the evaluation.

- Use the available flags to report an empty submission, a non-functioning program, a Norm error, cheating, etc.

Attachments

subject.pdf (https://github.com/rphlr/42-Subjects/)

Go to homepage 🏠 (../../)

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

Yes No

Exercise 00: Annnnnnd... ACTION!

As usual, there should be enough tests to prove that the program works as requested. If there are none, do not grade this exercise.

Class and Attributes

There is a ClapTrap class. It has all the following private attributes:

- name
- hit points
- energy points
- attack damage

These attributes are initialized to the requested values.

Yes No

Go to homepage 🏠 (../../)

Member Functions

The following member functions are present and function as specified:

- attack()
- takeDamage()
- beRepaired()

Yes No

Exercise 01: Serena, My Love!

As usual, there should be enough tests to prove that the program works as requested. If there are none, do not grade this exercise.

Class and Attributes

There is a ScavTrap class. ScavTrap publicly inherits from the Claptrap class. It does not redeclare the attributes. The attributes of the ClapTrap class are now protected instead of private. The attributes are initialized to the requested values.

Yes No

Member Functions

The following member functions are present and functional:

- attack()
- takeDamage() (inherited)
- beRepaired() (inherited)

The messages from the constructor, destructor, and attack() function must be different from those of the ClapTrap.

Yes No

Construction and Destruction

ScavTrap must have a constructor and destructor with specific messages. Their proper implementation must be demonstrated by a sequence of calls in the expected order: if you create a ScavTrap, the message from the ClapTrap constructor should be displayed first, followed by that of the ScavTrap. Conversely, if you delete a ScavTrap, the message from the ScavTrap destructor should be displayed

first, followed by that of the ClapTrap.

Yes No

Go to homepage 🏠 (../../)

Special Feature

ScavTrap has a guardGate() function that displays a message on the standard output. ScavTrap also has an attack() function that displays a message different from that of the ClapTrap on the standard output.

Yes No

Exercise 02: Assembly Line Work

As usual, there should be enough tests to prove that the program works as requested. If there are none, do not grade this exercise.

Class and Attributes

There is a FragTrap class that publicly inherits from ClapTrap. Attributes should not be redeclared without reason.

Yes No

Construction and Destruction

FragTrap must have a constructor and destructor with specific messages. Their proper implementation must be demonstrated by a sequence of calls in the expected order: if you create a FragTrap, the message from the ClapTrap constructor should be displayed first, followed by that of the FragTrap. Conversely, if you delete a FragTrap, the message from the FragTrap destructor should be displayed first, followed by that of the ClapTrap.

Yes No

Special Feature

FragTrap has a highFivesGuys() function that displays a message on the standard output.

Yes No

Exercise 03: Ok, This Is Getting Weird

As usual, there should be enough tests to prove that the program works as requested. If there are none, do not grade this exercise.

Go to homepage 🏠 (/)

The Ultimate Weirdness of C++

There is a DiamondTrap class. It inherits from both FragTrap and ScavTrap. It defines attributes with the requested values. It uses virtual inheritance to avoid the pitfalls of diamond inheritance.

Yes No

Choose Wisely...

The DiamondTrap class uses the attack() function of Scavtrap. It has the special functions of both its parents. DiamondTrap has a private member std::string name. The whoAmI() function has access to both name and ClapTrap::name.

Yes No

Ratings

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

Ok Outstanding project

Empty work Incomplete work W Invalid compilation Cheat Crash

Concerning situation Leaks l Forbidden function Can’t support / explain code

Conclusion

Give this repository a star. ⭐

(https://github.com/rphlr/42-Evals)