Menu

My projects

Holy Graph

List projects

Available Cursus

Your projects

minishell

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 03

You should evaluate 1 student in this team

Git repository

```
git@vogsphere-v2-bg.1
```

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 0 (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 0. 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 flag.

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 peadagogy is valid only and
only if peer-evaluation is conducted seriously.

Guidelines

You must compile with clang++, with -Wall -Wextra -Werror
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 flags and/or with something other than clang++

Any of these means that you must flag the project as Forbidden Function:
- Use of a "C" 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
- Use of an already existing container, or any existing function, to implement another container

Attachments

subject.pdf

ex00

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.

Class and attributes

There is a FragTrap class present.
It has all the required attributes.
Its attributes are initialized to the required values.

Yes | No

Special attack

There is a vaulthunter_dot_exe function that works as specified by
the subject.

Yes | No

It's called style, look it up.

How elegant do you think the method used to determine the attack
in the vaulthunder_dot_exe function is?

Rate it from 0 (failed) through 5 (excellent)

I AM FUNNYBOT. AWKWAAAARD! AWKWAAAAARD!

How funny are the output messages?

Rate it from 0 (failed) through 5 (excellent)

Member functions

The following member functions are present and work as specified:

- rangedAttack
- meleeAttack
- takeDamage
- beRepaired

Also, the constraints about the HP limits and the armor reduction must be
taken into account.

Yes | No

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.

Class and attributes

There is a ScavTrap class present.
It has all the required attributes.
Its attributes are initialized to the required values.

Yes | No

Member functions

The following member functions are present and work as specified:

- rangedAttack
- meleeAttack
- takeDamage
- beRepaired

Also, the constraints about the HP limits and the armor reduction must be taken into account.
The outputs of the constructor, destructor, rangedAttack and meleeAttack must be different from the ones in the previous exercise.

Yes | No

Wow, what a t-t-terrific audience!

How funny are the output messages ?

Rate it from 0 (failed) through 5 (excellent)

Special features

There is a challengeNewcomer function that works as specified by the subject.

Yes | No

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.

Parent class

There is a ClapTrap class present, and both ScavTrap and FragTrap inherit publicly from it.
All the functions and attributes that were shared between both ScavTrap and FragTrap are now in ClapTrap, namely:
- Hit points
- Max hit points
- Energy points
- Max energy points

- Level
- Name
- Melee damage
- Ranged damage
- Armor
- damage reduction
- takeDamage
- beRepaired

rangedAttack and meleeAttack can either be in the ClapTrap class and use an attribute to have a different output depending on the child class, or remain in the child classes.
Whatever.
The attributes specific to each class (vaulthunter_dot_exe,

challengeNewcomer, and whatever the student created to help with these)
must of course remain where they are.

Yes | No

Construction and destruction

There must be a constructor and a destructor for the ClapTrap with
its own specific messages, and it must be implemented so that it is called in the correct order
when used, namely, if you create a FragTrap it must first display the ClapTrap's
message then the FragTrap's, and if you delete it, it must display the FragTrap's
message first, then the ClapTrap's

Yes | No

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.

But once, he break out of his cage, and he "get this"! Very nice.

The ninjaShoebox function has to do something funny! Even better
if it's different for each ClapTrap type

Rate it from 0 (failed) through 5 (excellent)

Special attack

There is a ninjaShoebox function that is present multiple times in
the NinjaTrap, one for each ClapTrap concrete type that can be taken as parameter
(So, ClapTrap, ScavTrap, FragTrap and NinjaTrap).

Yes | No

Subclass

There is a NinjaTrap class present.
It inherits from ClapTrap,
and sets the attributes to their appropriate values.

Yes | No

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.

Ultimate shoebox

There is a SuperTrap class present.
It inherits from both the
FragTrap and the NinjaTrap.
It sets the attributes to the appropriate values.
It uses virtual inheritance to avoid the pitfalls of diamond inheritance.

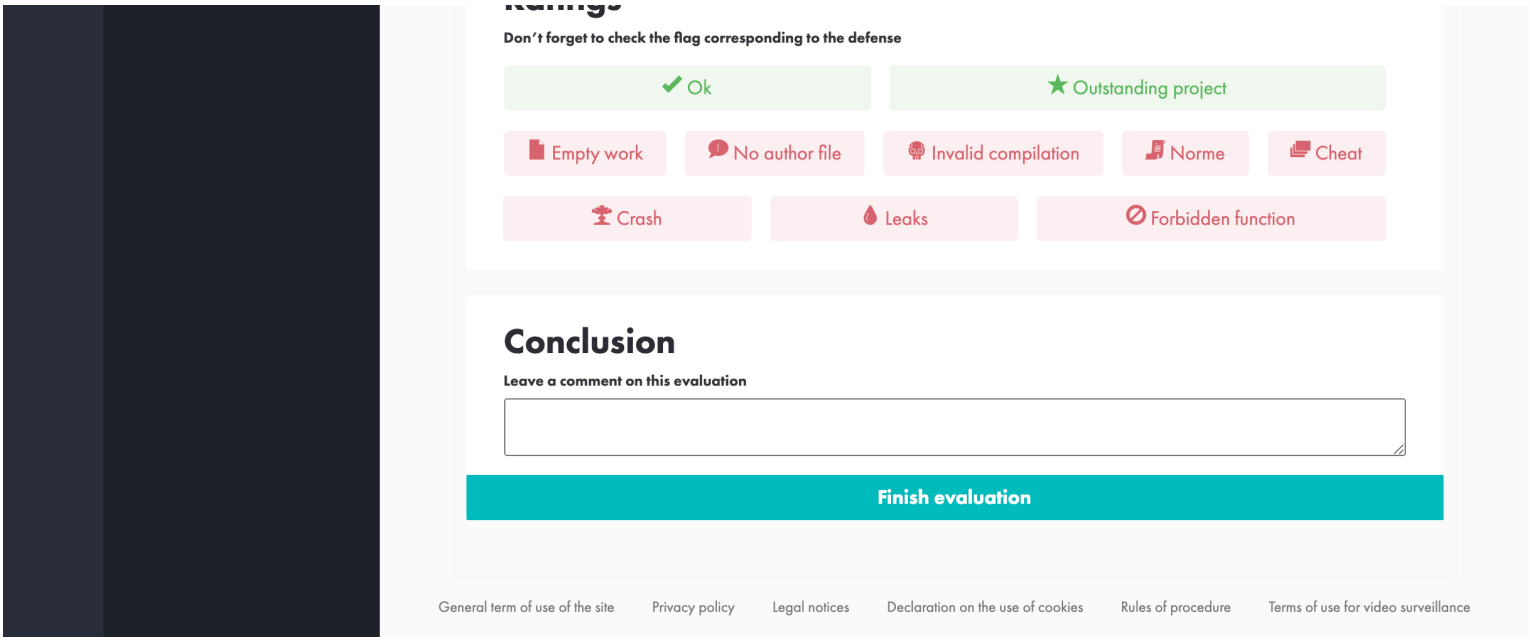
Yes | No

Choose wisely...

The SuperTrap uses the rangedAttack of the FragTrap and the meleeAttack
of
the NinjaTrap.
It has the special attacks of both its parents.

Yes | No

Ratings



Conclusion

Leave a comment on this evaluation

Finish evaluation