SCALE FOR PROJECT CUB3D (/PROJECTS/CUB3D)

You should evaluate 2 students in this team

★

Git repository

```
git@vogsphere.42vienna.com:vogsphere/intra-uuid-b98f6214-a4dc-456f-9
```

Introduction

Please adhere to the following rules:

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

- Identify the possible dysfunctions in the project of the student or group whose work is being evaluated. Take the time to discuss and debate the problems that may have been identified.

- 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. Pedagogy is useful only if peer evaluation is done seriously.

Guidelines

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

- Double-check that the Git repository belongs to the student or students. Ensure that the project is the expected one. Also, check that 'git clone' is used in an empty directory.

- Check carefully that no malicious aliases were used to deceive 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 grading, such as testing or automation scripts.

- If you have not completed the assignment you are going to evaluate, you must read the entire subject before starting the evaluation process.

- Use the available flags to report an empty repository, a non-functioning program, a Norm error, cheating, etc.
In these cases, the evaluation process ends, and the final grade is 0, or -42 in the case of cheating. However, except in cases of cheating, students are strongly encouraged to review the submitted work together to identify any mistakes that should not be repeated in the future.

- You must also verify that there are no memory leaks. Any memory allocated on

the heap must be properly freed before the program's execution ends.
You are allowed to use any of the tools available on the computer,
such as leaks, valgrind, or e_fence. In case of memory leaks, tick the
appropriate flag.

Attachments

- subject.pdf (https://cdn.intra.42.fr/pdf/pdf/179707/en.subject.pdf)
- minilibx_macos_metal.tgz (https://cdn.intra.42.fr/document/document/40544/minilibx_macos_metal.tgz)
- minilibx-linux.tgz (https://cdn.intra.42.fr/document/document/40545/minilibx-linux.tgz)
- minilibx_macos_opengl.tgz (https://cdn.intra.42.fr/document/document/40546/minilibx_macos_opengl.tgz)

Mandatory part

Executable name

Check that the project compiles well (without re-link) when you execute the `make` command and that the executable name is `cub3D`.
If not, use the invalid compilation flag at the end of the scale.

☑ Yes ☒ No

Configuration file

Ensure that you can configure ALL the following elements in the configuration file.
The formatting must be as described in the subject.

- north texture path - NO
- east texture path - EA
- south texture path - SO
- west texture path - WE
- floor color - F
- ceiling color - C
- the map (see subject for the map configuration details)

Additionally, verify that the program returns an error and exits properly when the configuration file is misconfigured (e.g., an unknown key, double keys, an invalid path) or if the filename does not end with the `.cub` extension.

If not, the defense is over, and use the appropriate flag incomplete work or crash.

☑ Yes ☒ No

Technical elements of the display

We will evaluate the technical elements of the display. Execute the program and perform the following tests. If any test fails, no points will be awarded for this section. Proceed to the next section.

- A window must open at the launch of the program.
  It must stay open during the whole execution.
- An image representing the inside of a maze must be displayed inside the window.
- Hide all or part of the window by using another window

or by using the screen's borders, then minimize the windows and maximize it again. In all cases, the window's content must remain consistent.

☑ Yes ☒ No

User basic events

In this section, we wil evaluate the program's user-generated events. Perform the following three tests. If any test fails, no points will be awarded for this section; proceed to the next section.

- Click the red cross at the top left of the window. The window must close, and the program must exit cleanly.
- Press the ESC key. The window must close and the program must exit cleanly. For this test, we will accept that another key, such as Q, exits the program.
- Press the four movement keys (we'll accept WASD or ZQSD keys) in the order of your liking. Each key press must produce a visible result in the window, such as a player's movement or rotation.

☑ Yes ☒ No

Movements

In this section, we will evaluate the implementation of the player's movement and orientation within the maze.
Perform the following five tests. If any test fails, no points will be awarded for this section.

- The player's initial orientation in the first image must align with the configuration file; test for each cardinal orientation (N, S, E, W).
- Press the left arrow, then the right arrow.
  The player's view must rotate to the left, then to the right, as if the player’s head were moving.
- Press W (or Z), then S.
  The player's view must move forward and then backward in a straight line.
- Press A (or Q), then D.
  The player's view must move to the left and then to the right in a straight line.
- During these four movements, was the display smooth? By 'smooth', we mean whether the game is 'playable' or slow.

☑ Yes ☒ No

Walls

In this section, we will evaluate the walls in the maze.
Perform the following four tests. If any test fails, no points should be awarded for this section.

- The wall's texture varies depending on which compass point the wall faces (north, south, east, west).
  Ensure that the textures on the walls and perspective are

visible and correct.

- Verify that modifying the path of a wall texture in the configuration file changes the rendered texture when the program is re-executed.
- Additionally, verify that setting a non-existent path raises an error.
- Ensure that the floor and ceiling colors are correctly handled when modified in the configuration file.

Yes No

Error management

In this section, we will evaluate the program's error management and reliability. Perform the following four tests. If any test fails, no points will be awarded for this section; proceed to the next section.

- Execute the program using various arguments and random values. Even if the program does not require any arguments, it is critical that those arguments do not cause unexpected or unhandled errors.
- Verify that there are no memory leaks. You may use the `top` or `leaks` command in another shell to ensure that memory usage is stable. Memory usage must not increase with each action performed.
- Roll either your arm or face across the keyboard. The program must not exhibit any strange behaviors and must remain functional.
- Alter the map. The program must not exhibit any strange behaviors and must stay functional if the map is correctly configured; otherwise, it must raise an error.

Yes No

Bonus

We will consider your bonuses if and only if your mandatory part is excellent. This means you must complete the mandatory part from beginning to end, and your error management must be flawless, even in cases of improper or incorrect usage. Therefore, if the mandatory part did not score all the points during this defense, bonuses will be completely ignored.

When I'll be older I'll be John Carmack

Refer to the subject's bonus section and add one point for each implemented and fully functional bonus.

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

Ratings

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

Ok Outstanding project

Empty work Incomplete work Invalid compilation Norme Cheat Crash

Incomplete group Concerning situation Leaks Forbidden function

Can't support / explain code

Conclusion

Leave a comment on this evaluation (2048 chars max)

very good work, project is nice, readable and clean.

Finish evaluation