42 Evals

Back to all evaluation sheets

Points earned
0

get_next_line

You should evaluate 1 student in this team

Introduction

Please follow the rules below:

This tool is designed to help you self-evaluate your project. By following the questions and grading criteria, you can identify potential issues in your work and improve your project accordingly.

Keep in mind that this is not an official evaluation but a way to assess your own progress. Use it as a guide to understand where you might need to make adjustments.

Guidelines

Please follow the guidelines below:

- Follow the structure provided to assess your project.
- Any **segmentation fault** , **invalid compilation** , or **memory leak** will result in a final grade of **0** for the project. Make sure your program compiles and runs without these issues.
- No **Norm errors** are tolerated. If your code does not comply with the Norm,

this also results in a **0** .

If you are using this tool to simulate an evaluation, remember that it only reflects common evaluation practices and not the off

This tool is an example to guide your self-assess considered a replacement for official project evaluati

Points earned
0

Attachments

Please download the attachments below:

subject.pdf

Mandatory Part

get_next_line

Does the `get_next_line` function correctly return each line from the file descriptor?

Does the `get_next_line` function correctly return each line from the file descriptor?

Yes No

Error management

Does the function return NULL when there is nothing more to read or when an error occurs?

Does the function return NULL when there is nothing more to read or when an error occurs?

Yes No

Points earned
0

Management of `\n`

Does the returned line include the newline character (`\n`) unless at the end of the file?

Does the returned line include the newline character (`\n`) unless at the end of the file?

Yes No

-D BUFFER_SIZE

Does the code compile without errors when the `-D BUFFER_SIZE` flag is used?

Does the code compile without errors when the `-D BUFFER_SIZE` flag is used?

Yes No

`get_next_line` Efficiency

How efficient is the function when handling various buffer sizes (e.g., BUFFER_SIZE=1, 42, 9999)?

1 point: Inefficient, large delays or errors with smaller or larger buffer sizes.

2 points: Struggles with edge cases for buffer

3 points: Handles typical buffer sizes but issu

4 points: Efficient for most cases.

5 points: Efficient in all tested buffer sizes, no noticeable delay or issues.

Points earned
0

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

0 1 2 3 4 5

Error Handling

How robust is the error handling in the function (e.g., invalid file descriptors, memory allocation failures)?

1 point: Fails to handle errors, crashes.

2 points: Handles only some errors but still crashes in other cases.

3 points: Handles common errors but fails in edge cases.

4 points: Handles most errors well.

5 points: Handles all errors gracefully without crashing.

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

0 1 2 3 4 5

Bonus Part

Points earned
0

Multiple File Descriptors

Is the `get_next_line` function able to handle multi
same time?

Is the `get_next_line` function able to handle multiple file descriptors at the same time?

Yes No

Single Static Variable

Does the bonus part use only one static variable?

Does the bonus part use only one static variable?

Yes No

Switching between different file descriptors

Does the function work correctly when switching between different file descriptors?

Does the function work correctly when switching between different file descriptors?

Yes No

Ratings

OK

Outstanding

Points earned
0

Incomplete Work

Invalid Co

Norme

Cheat

Crash

Concerning Situations

Leaks

Forbidden Functions

Cannot Support/Explain code