42 Evals

Back to all evaluation sheets

Points earned
0

Libft

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

LibC Functions

Does the `ft_strlen` function return the correct string length?

Does the `ft_strlen` function return the correct string length?

Yes No

LibC Functions

Does the `ft_isalpha` function correctly identify alphabetic characters?

Does the `ft_isalpha` function correctly identify alphabetic characters?

Yes No

Points earned
0

LibC Functions

Does the `ft_isdigit` function properly detect digit c

Does the `ft_isdigit` function properly detect digit characters?

Yes No

LibC Functions

Does the `ft_strdup` function correctly allocate memory and duplicate strings?

Does the `ft_strdup` function correctly allocate memory and duplicate strings?

Yes No

ft_memcpy

How accurately does the `ft_memcpy` function copy memory blocks?

- 1 point: Basic implementation but inefficient.
- 2 points: Handles simple cases but lacks robustness.
- 3 points: Correct for most cases, struggles with edge cases (e.g., overlapping memory).

- 4 points: Handles almost all cases with minor issues.
- 5 points: Fully correct and efficient, handles all edge cases.

Points earned
0

Rate it from 0 (failed) through 5 (ex

0 1 2 3

ft_atoi

How effectively does the `ft_atoi` function convert a string to an integer?

- 1 point: Basic conversion, fails on non-trivial inputs.
- 2 points: Correct for simple strings, but lacks support for special cases (e.g., spaces, signs).
- 3 points: Correct but fails on boundary cases like overflow.
- 4 points: Works well but with minor inefficiencies.
- 5 points: Handles all cases, including edge and special cases.

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

0 1 2 3 4 5

ft_substr

How well does the `ft_substr` function extract substrings?

- 1 point: Basic extraction but with incorrect edge handling.
- 2 points: Works for simple cases, but fails for out-of-bounds indices.

- 3 points: Correct but struggles with null or empty strings.
- 4 points: Handles most cases efficiently.
- 5 points: Perfectly extracts substrings, even fo

Points earned
0

Rate it from 0 (failed) through 5 (ex

0 1 2 3 4 5

ft_strjoin

How accurately does the `ft_strjoin` function concatenate two strings?

- 1 point: Basic concatenation but with errors (e.g., missing null terminators).
- 2 points: Joins correctly but with minor issues in specific cases.
- 3 points: Correct for most cases, issues with handling null or empty strings.
- 4 points: Mostly correct with small inefficiencies.
- 5 points: Perfectly joins strings, handling all edge cases efficiently.

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

0 1 2 3 4 5

Bonus Part

ft_lstnew

Is the `ft_lstnew` function correctly implemented, c

Is the `ft_lstnew` function correctly implemented

Yes No

Points earned
0

ft_lstadd_front

Does the `ft_lstadd_front` function correctly add nodes to the beginning of the list?

Does the `ft_lstadd_front` function correctly add nodes to the beginning of the list?

Yes No

ft_lstsize

Does the `ft_lstsize` function correctly return the size of the list?

Does the `ft_lstsize` function correctly return the size of the list?

Yes No

ft_lstlast

Does the `ft_lstlast` function correctly return the last node of the list?

Does the `ft_lstlast` function correctly return tl

Yes No

Points earned
0

ft_lstadd_back

Does the `ft_lstadd_back` function correctly add nodes to the end of the list?

Does the `ft_lstadd_back` function correctly add nodes to the end of the list?

Yes No

Ratings

OK Outstanding Empty Work

Incomplete Work Invalid Compilation

Norme Cheat Crash

Concerning Situations Leaks

Forbidden Functions

Points earned

0