42 Evals

Back to all evaluation sheets

Points earned

0

push_swap

You should evaluate 1 student in this team

Introduction

Please follow the rules below:

Remain polite, courteous, respectful and constructive throughout the evaluation process. The well-being of the community depends on it. Identify with the student or group whose work is evaluated the possible dysfunctions in their project. 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. The pedagogy is useful only and only if the peer-evaluation is done seriously.

Guidelines

Please follow the guidelines below:

Only grade the work that was turned in the Git repository of the evaluated student or group.

Double-check that the Git repository belongs to the student(s). Ensure that the

project is the one expected. Also, check that 'git clone' is used in an empty folder.s

Check carefully that no malicious aliases was used to evaluate something that is not the content of the offic

To avoid any surprises and if applicable, review toge facilitate the grading (scripts for testing or automatio

Points earned

0

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

Use the available flags to report an empty repository, a non-functioning program, a Norm error, cheating, and so forth.

In these cases, the evaluation process ends and the final grade is 0, or -42 in case of cheating. However, except for cheating, student are strongly encouraged to review together the work that was turned in, in order to identify any mistakes that shouldn't be repeated in the future.

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.

Attachments

Please download the attachments below:

🔗 subject.pdf

🔗 checker_Mac

🔗 checker_linux

Mandatory Part

Memory leaks

Reminder : Remember that for the duration of the ⬚
other unexpected, premature, uncontrolled or une⬚
program, else the final grade is 0. Use the appropr⬚
active thoughout the whole defence.

Points earned

0

Throughout the defence, pay attention to the amount of memory

used by push_swap (using the command top for example) in order

to detect any anomalies and ensure that allocated memory is

properly freed. If there is one memory leak (or more), the final

grade is 0.

Yes   No

Error management

In this section, we'll evaluate the push_swap's error management. If at least
one fails, no points will be awarded for this section. Move to the next one.

Run push_swap with non numeric parameters. The program must display
"Error".

Run push_swap with a duplicate numeric parameter. The program must
display "Error".

Run push_swap with only numeric parameters including one greater than
MAXINT. The program must display "Error".

Run push_swap without any parameters. The program must not display
anything and give the prompt back.

Yes   No

Push_swap - Identity test

In this section, we'll evaluate push_swap's behavi⟨...⟩
has already been sorted. Execute the following 3 t⟨...⟩
no points will be awarded for this section. Move to the next one.

Points earned

0

Run the following command "$>./push_swap 42". The program should
display nothing (0 instruction).

Run the following command "$>./push_swap 2 3". The program should
display nothing (0 instruction).

Run the following command "$>./push_swap 0 1 2 3". The program should
display nothing (0 instruction).

Run the following command "$>./push_swap 0 1 2 3 4 5 6 7 8 9". The
program should display nothing (0 instruction).

Run the following command "$>./push_swap 'Between 0 and 9 randomly
sorted values chosen>'. The program should display nothing (0 instruction).

Yes   No

Push_swap - Simple version

If the following tests fails, no points will be awarded for this section. Move to
the next one. Use the checker binary given on the attachments.

Run "$>ARG="2 1 0"; ./push_swap $ARG | ./checker_OS $ARG".

Check that the checker program displays "OK" and that the size of the list of
instructions from push_swap is 2 OR 3. Otherwise the test fails.

Run "$>ARG="'Between 0 and 3 randomly values chosen'"; ./push_swap

$ARG | ./checker_OS $ARG". Check that the checker program displays "OK" and that the size of the list of instructions from push_swap is between 0 AND 3. Otherwise the test fails.

Yes   No

Points earned

0

Another simple version

Execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one. Use the checker binary given on the attachments.

Run "$>ARG="1 5 2 4 3"; ./push_swap $ARG | ./checker_OS $ARG".

Check that the checker program displays "OK" and that the size of the list of instructions from push_swap isn't more than 12. Kudos if the size of the list of instructions is 8.

Run "$>ARG="<5 random values>"; ./push_swap $ARG | ./checker_OS $ARG" and replace the placeholder by 5 random valid values.

Check that the checker program displays "OK" and that the size of the list of instructions from push_swap isn't more than 12. Otherwise this test fails. You'll have to specifically check that the program wasn't developed to only answer correctly on the test included in this scale. You should repeat this test couple of times with several permutations before you validate it.

Yes   No

Push_swap - Middle version

If the following test fails, no points will be awarded for this section. Move to the next one. Move to the next one. Use the checker binary given on the attachments.

Run "$>ARG="<100 random values>"; ./push_swap $ARG | ./checker_OS $ARG" and replace the placeholder by 100 random valid values. Check that the checker program displays "OK" and that the si⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯ instructions. Give points in accordance:

less than 700: 5

less than 900: 4

less than 1100: 3

less than 1300: 2

less than 1500: 1

You'll have to specifically check that the program wasn't developed to only answer correctly on the test included in this scale. You should repeat this test couple of times with several permutations before you validate it.

Points earned

0

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

◯

0   1   2   3   4   5

Push_swap - Advanced version

If the following test fails, no points will be awarded for this section. Move to the next one. Move to the next one. Use the checker binary given on the attachments.

Run "$>ARG="<500 random values>"; ./push_swap $ARG | ./checker_OS $ARG" and replace the placeholder by 500 random valid values (One is not called John/Jane Script for nothing). Check that the checker program displays "OK" and that the size of the list of instructions

less than 5500: 5

less than 7000: 4

less than 8500: 3

less than 10000: 2

less than 11500: 1

You'll have to specifically check that the program
answer correctly on the test included in this scale.
test couple of times with several permutations bef

Points earned

0

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

◯

0   1   2   3   4   5

Bonus Part

Checker program - Error management

In this section, we'll evaluate the checker's error management. If at least
one fails, no points will be awarded for this section. Move to the next one.

Run checker with non numeric parameters. The program must display
"Error".

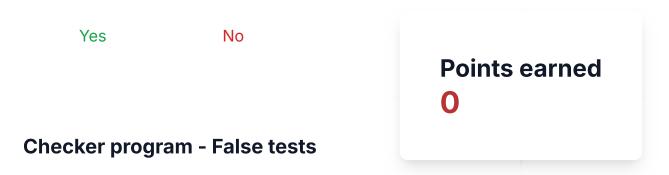Run checker with a duplicate numeric parameter. The program must display
"Error".

Run checker with only numeric parameters including one greater than
MAXINT. The program must display "Error".

Run checker without any parameters. The program must not display
anything and give the prompt back.

Run checker with valid parameters, and write an action that doesn't exist
during the instruction phase. The program must display "Error".

Run checker with valid parameters, and write an action with one or several
spaces before and/or after the action during the instruction phase. The

program must display "Error".

Yes   No

Checker program - False tests

In this section, we'll evaluate the checker's ability to manage a list of instructions that doesn't sort the list. Execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one. Don't forget to press CTRL+D to stop reading during the intruction phase.

Run checker with the following command "$>./checker 0 9 1 8 2 7 3 6 4 5" then write the following valid action list "[sa, pb, rrr]". Checker should display "KO".

Run checker with a valid list as parameter of your choice then write a valid instruction list that doesn't order the integers. Checker should display "KO". You'll have to specifically check that the program wasn't developed to only answer correctly on the test included in this scale. You should repeat this test couple of times with several permutations before you validate it.

Yes   No

Checker program - Right tests

In this section, we'll evaluate the checker's ability to manage a list of instructions that sort the list. Execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one. Don't forget to press CTRL+D to stop reading during the instruction phase.

Run checker with the following command "$>./checker 0 1 2" then press CTRL+D without writing any instruction. The program should display "OK".

Run checker with the following command "$>./checker 0 9 1 8 2" then write the following valid action list "[pb, ra, pb, ra, sa, ra, pa, pa]". The program
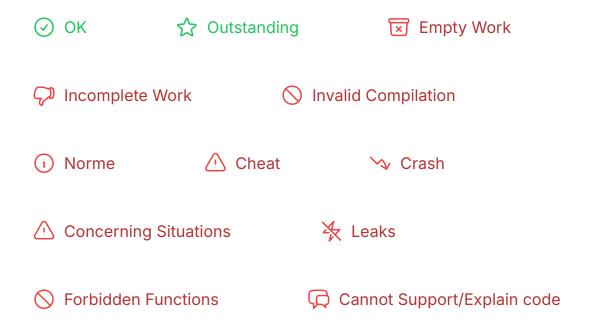
should display "OK".

Run checker with a valid list as parameter of your choice then write a valid instruction list that order the integers. Checker mu[...]
have to specifically check that the program wasn't [...]
answer correctly on the test included in this scale. [...]
test couple of times with several permutations bef[...]

Points earned

0

Yes   No

Ratings

✓ OK   ☆ Outstanding   Empty Work

👎 Incomplete Work   🚫 Invalid Compilation

ⓘ Norme   ⚠ Cheat   Crash

⚠ Concerning Situations   Leaks

🚫 Forbidden Functions   Cannot Support/Explain code