Attachments

subject.pdf

Mandatory Part

Compile

- USE make -n to see if compilation use -Wall -Wextra -Werror if not use invalid compilation flags
- minishell Compile without errors if not use flags
- makefile must not re-link

Yes | No

Simple Command & global

- Execute a simple command with an absolute path like /bin/ls or any other command without options
- How many global variables? why? Give a concrete example of why it feels mandatory or logical.
- Test an empty command.
- Test only spaces or tabs.
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes | No

Arguments & history

- Execute a simple command with an absolute path like /bin/ls or any other command with arguments but without quotes and double quotes
- Repeat multiple times with different commands and arguments
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes | No

echo

- Execute the echo command with or without arguments or -n
- Repeat multiple times with different arguments
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes | No

exit

- Execute exit command with or without arguments
- Repeat multiple times with different arguments
- Don't forget to relaunch the minishell
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes No

Return value of a process

- Execute a simple command with an absolute path like /bin/ls or any other command with arguments but without quotes and double quotes then execute echo $?
- Check the printed value. You can repeat the same in bash and compare it.
- Repeat multiple times with different commands and arguments, use some failing commands like '/bin/ls filethatdoesntexist'
- anything like expr $? + $?
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes No

Signals

- Try ctrl-C in an empty prompt should show a new line with a new prompt
- Try ctrl-\ in an empty prompt should not do anything
- Try ctrl-D in an empty prompt should quit minishell –> RELAUNCH!
- Try ctrl-C in a prompt after you wrote some stuff should show a new line with a new prompt
- The buffer should be clean too, press "enter" to make sure nothing from the old line is executed.
- Try ctrl-D in a prompt after you wrote some stuff should not do anything
- Try ctrl-\ in a prompt after you wrote some stuff should not do anything!
- Try ctrl-C after running a blocking command like cat without arguments or grep “something”
- Try ctrl-\ after running a blocking command like cat without arguments or grep “something”
- Try ctrl-D after running a blocking command like cat without arguments or grep “something”
- Repeat multiple times with different commands
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes No

Double Quotes

- Execute a simple command with arguments but this time double quotes (you should include whitespaces)
- a command like : echo "cat lol.c | cat > lol.c"
- anything except $.
- if something crashes use the crash flag.
- if something is not working use the incomplete work flag.

Yes No

Single Quotes

- Execute commands with single quotes as an argument
- Try empty arguments
- Try environment variables, whitespaces, pipes, redirection in the single quotes
- echo '$USER' must print $USER
- Nothing should be interpreted

Yes No

env

- Check if env shows you the current environment variables

Yes | No

export

- Export environment variables, create new ones and replace old ones
- Check them with env

Yes | No

unset

- Export environment variables, create new ones and replace old ones
- Use unset to remove some of them
- Check the result with env

Yes | No

cd

- Use the command cd to move the working directory and check if you are in the right directory with /bin/ls
- Repeat multiple times with working and not working cd
- try '.' '..' as arguments too

Yes | No

pwd

- Use the command pwd
- Repeat multiple times in multiple directories

Yes | No

Relative Path

- Execute commands but this time use a relative path
- Repeat multiple times in multiple directories with a complex relative path (lots of ..)

Yes | No

Environment Path

- Execute commands but this time without any path. (ls, wc, awk etc...)
- Unset the $PATH and check if it is not working anymore
- Set the $PATH to a multiple directory value (directory1:directory2) and check that directories are checked in order from left to right

Yes | No

Redirection

- Execute commands with redirections < and/or >
- Repeat multiple times with different commands and arguments and sometimes change > with >>
- Check if multiple of the same redirections fail
- Test << redirection (it doesn't need to update history).

Yes | No

Pipes

- Execute commands with pipes like 'cat file | grep bla | more'
- Repeat multiple times with different commands and arguments
- Try some failing commands like 'ls filethatdoesntexist | grep bla | more'
- Try to mix pipes and redirections.

Yes | No

Go Crazy and history

- type a command line then use ctrl-C then press enter the buffer should be clean and nothing try to execute.
- Can we navigate through history with up and down and retry some command
- Execute commands that should not work like 'dsbksdgbksdghsd' and check if the shell doesn't crash and prints an error
- cat | cat | ls behave "normally"
- Try to execute a long command with a ton of arguments
- Have fun with that beautiful minishell and enjoy it

Yes | No

Environment Variables

- Execute echo with some $ variables as arguments
- Check that $ is interpreted as an environment variable
- Check that double quotes interpolate $
- Check that $USER exists or set it.
- echo "$USER" should print the value of $USER

Yes | No

Bonus

We will look at your bonuses if and only if your mandatory part is excellent. This means that you must complete the mandatory part, beginning to end, and your error management must be flawless, even in cases of twisted or bad usage. So if you didn't score all the points on the mandatory part during this defense bonuses will be totally ignored.

And, Or

- Use &&, || and parenthesis with commands and check if it works as bash

Yes | No

WildCard

- Use wildcards in arguments for the local directory.

Yes | No

Surprise (or not...)

- set USER environment variable.
- Test echo "'$USER'" this should print 'USER_VALUE'
- Test echo "'$USER'" this should print "$USER"

Yes | No

Ratings

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

Ok | Outstanding project

Empty work | No author file | Invalid compilation | Norme | Cheat | Crash | Leaks

Forbidden function

Conclusion

Leave a comment on this evaluation