{EPITECH}

MINISHELL1

BOOTSTRAP

MINISHELL1

Introduction

binary name: my_exec
language: C
groupe size: 1
compilation: via Makefile, including re, clean and fclean rules
authorized functions: malloc, free, exit, opendir, readdir, closedir, getcwd, chdir, fork, stat, lstat, fstat, open, close, getline, strtok, strtok_r, read, write, execve, access, isatty, wait, waitpid, wait3, wait4, signal, kill, getpid, strerror, perror, strsignal

- ✓ The totality of your source files, except all useless files (binary, temp files, objfiles,...), must be included in your delivery.
- ✓ All the bonus files (including a potential specific Makefile) should be in a directory named bonus.
- ✓ Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Step 1: we lied to you

Print the content of env

```
int    main(int argc, char **argv, char **env)
{
    ...
}
```

Step 2: a simple execution

With the step 1 done, add a program that execute /bin/ls using execve.

man execve

Step 3: segmentation (not fault)

Now we are going to prepare your program for the next step.
Your program has to take one parameter (a program name with its path and arguments) and transform into a char **

```
void    print_arg(char **arg)
{
    while (*arg)
        {
            my_putstr(*arg);
            my_putchar('\n');
            arg++;
        }
}
```

```
Terminal
~/B-PSU-200> ./my_exec "/bin/ls -l /dev"
/bin/ls
-l
/dev
```

Step 4: execute

Now rewrite your program whose has to take one string as parameter, that contains a program name with its path and arguments.
Your program must execute the program with these arguments and display as following:

```
Terminal
~/B-PSU-200> ./my_exec "/bin/ls -l /dev"
Program name: /bin/ls
Nb args: 2
PID: 1346
Child PID: 1348
... . .
... . .
... . . /* execution of ls -l /dev */
... . .
... . .
Program terminated.
Status: OK
```

You have to use fork (Read the man)

```
Terminal
~/B-PSU-200> ./my_exec "./my_segfault"
Program name: ./my_segfault
Nb args: 0
PID: 1513
Child PID: 1514
... . .
... . .
... . . /* execution of ./my_segfault */
... . .
... . .
Program terminated.
Status: Segmentation fault
```

How do you get the status of execve ?

v1