



EXERCISES — Micro Shell

version #7be580532266ed398481e31366afcc24b1950c2a



**The way is lit. The path is clear.
We require only the strength to follow it.**

Copyright

This document is for internal use at EPITA ([website](#)) only.

Copyright © 2022-2023 Assistants [<assistants@tickets.assistants.epita.fr>](mailto:assistants@tickets.assistants.epita.fr)

The use of this document must abide by the following rules:

- ▷ You downloaded it from the assistants' intranet.*
- ▷ This document is strictly personal and must **not** be passed onto someone else.
- ▷ Non-compliance with these rules can lead to severe sanctions.

Contents

1	Goal	4
2	Introduction	4
3	libruncom.a	4
3.1	add_command	4
3.2	del_command	4
3.3	run_command	5
4	micro_shell	5
5	Standard commands	5
5.1	cat	6
5.2	echo	6
5.3	head	6
5.4	quit	7

*<https://intra.assistants.epita.fr>

File Tree

```
micro_shell/
├── Makefile    (to submit)
├── micro_shell.c  (to submit)
└── run_command.c (to submit)
```

Makefile

- all: Produce the **libruncom.a** library and the **micro_shell** binary. This must be the first target.
- exec: Produce the **micro_shell** binary.
- library: Produce the **libruncom.a** library
- clean: Delete everything produced by make

Authorized functions : You are only allowed to use the following functions

- isatty(3)
- malloc(3)
- calloc(3)
- realloc(3)
- free(3)
- strerror(3)
- strcmp(3)
- printf(3)
- fprintf(3)
- puts(3)
- getline(3)
- fflush(3)
- fopen(3)
- feof(3)
- fclose(3)
- strtok_r(3)

Authorized headers : You are only allowed to use the functions defined in the following headers

- err.h
- errno.h
- assert.h
- stddef.h

Compilation : Your code must compile with the following flags

- -std=c99 -pedantic -Werror -Wall -Wextra -Wvla

Main function : Required

1 Goal

2 Introduction

In this exercise, you will have to implement a small command executer, a few standard commands, and an interactive prompt to use these commands.

Your `Makefile` must defines a least the following targets:

- *all*: Produce the *libruncom.a* library and the *micro_shell* binary. This must be the first target.
- *library*: Produce the *libruncom.a* library.
- *exec*: Produce the *micro_shell* binary.
- *clean*: Delete everything produced by *make*.

3 libruncom.a

Your library have to export three functions to manipulate commands:

3.1 add_command

The `add_command` function adds the function pointer `command` into an internal structure for a later use. If there is an error (such as a command with the same name already in the structure), the function returns `-1` and you must not change the internal structure. Otherwise, it returns `0`.

```
int add_command(const char *name, int (*command)(int, char **));
```

3.2 del_command

Removes the command `name` if it had been added before in the internal structure.

This function returns an `int` indicating whether the element has been suppressed or not. As an example, if `name` does not exist in the internal structure, the function returns `-1`. The function returns `0` if the function `name` has been successfully deleted.

```
int del_command(const char *name);
```

3.3 run_command

Execute the given command (first element of `argv`) if present in the internal structure. Parameters `argc` and `argv` are given as is to the command to execute. The function `run_command` returns 127 if name does not exist in the internal structure, the return value of the executed command otherwise.

```
int run_command(int argc, char **argv);
```

You are allowed to use **one** global variable for this library.

4 micro_shell

You will write a program named `micro_shell` which will read a line from the standard input, split it on the spaces, then execute the corresponding command using the library `libruncom.a`. The return code of your shell will be the one of the last executed command, 0 if no command was executed. Note that a command can have 10 arguments maximum. If there are more than 10 arguments, you may ignore the exceed.

There is an example of a session of `micro_shell` that uses a few of the commands to implement in input.

```
42sh$ cat test.txt
cat wut.txt
echo asdf
42sh$ ./micro_shell < test.txt
wut
wut
wut
asdf
42sh$
```

5 Standard commands

You will have to implement some commands for your `micro_shell`. If an error occurs while a command is executed, you must display the relevant error message (see `strerror(3)`) and return 1. If everything goes well, the command has to return 0. For every command, refer to its man page. You do not have to handle any option of these functions. You can assume that we will not call your commands with more than 10 arguments. If the command is unknown, you must print nothing and return 127 if it was the last command.

5.1 cat

cat prints on the standard output the content of each file passed as an argument, without any separator, in the given order. If there is an error, stop the processing of the arguments.

```
42sh$ ./micro_shell
21sh$ cat wut.txt
wut
wut
wut
21sh$ cat nop.txt
cat: nop.txt: No such file or directory
21sh$ cat
cat: Argument is missing
```

5.2 echo

echo prints on the standard output given arguments as is, followed by a newline character.

```
42sh$ ./micro_shell
21sh$ echo

21sh$ echo foo
foo
21sh$ echo foo bar
foo bar
21sh$ echo *
*
```

5.3 head

head prints on the standard output the first 10 lines of the given files. In the case where there is more than one file to read, head will display the name of the read file before its content. If there is an error, stop the processing of the arguments.

```
42sh$ ./micro_shell
21sh$ head wup
wup
wup
wup
21sh$ head wup wup
==> wup <==
wup
wup
wup

==> wup <==
wup
wup
wup
```

(continues on next page)

(continued from previous page)

```
21sh$ head nop.txt
head: nop.txt: No such file or directory
21sh$ head
head: Argument is missing
```

5.4 quit

The `quit` command will stop the execution of your `micro_shell` correctly. Its return value has to be the same than the one of the last executed command. If there was not any command executed before, it returns 0.

The way is lit. The path is clear. We require only the strength to follow it.