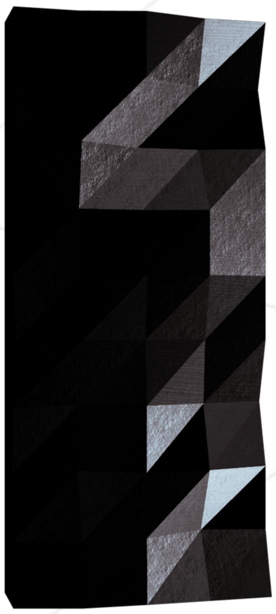

C Bootcamp

Day 12

Staff WeThinkCode_ info@wethinkcode.co.za

Summary: This document is the subject for Day12 of the C Bootcamp @ WeThinkCode_.

Contents

Chapter I

Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change up to an hour before submission.
- Make sure you have the appropriate permissions on your files and directories.
- You have to follow the submission procedures for every exercise.
- Your exercises will be checked and graded by your fellow classmates.
- On top of that, your exercises will be checked and graded by a program called Moulinette.
- Moulinette is very meticulous and strict in its evaluation of your work. It is entirely automated and there is no way to negotiate with it. So if you want to avoid bad surprises, be as thorough as possible.
- Moulinette is not very open-minded. It won't try and understand your code if it doesn't respect the Norm. Moulinette relies on a program called `Norminator` to check if your files respect the norm. TL;DR: it would be idiotic to submit a piece of work that doesn't pass `Norminator`'s check.
- These exercises are carefully laid out by order of difficulty - from easiest to hardest. We `will not` take into account a successfully completed harder exercise if an easier one is not perfectly functional.
- Using a forbidden function is considered cheating. Cheaters get `-42`, and this grade is non-negotiable.
- If ft_putchar() is an authorized function, we will compile your code with our `ft_putchar.c`.
- You'll only have to submit a main() function if we ask for a program.

- Moulinette compiles with these flags: -Wall -Wextra -Werror, and uses `gcc`.
- If your program doesn't compile, you'll get `0`.
- You cannot leave any additional file in your directory than those specified in the subject.
- Got a question? Ask your peer on your right. Otherwise, try your peer on your left.
- Your reference guide is called `Google / man / the Internet / ....`
- Check out the "C Bootcamp" part of the forum on the intranet.
- Examine the examples thoroughly. They could very well call for details that are not explicitly mentioned in the subject...
- By Odin, by Thor ! Use your brain !!!

Chapter II

Foreword

An interesting review about the movie `Face/Off`.

```
"I'd like to take his face...off." Yes, the pinnacle of American action cinema
is available for your viewing pleasure in the form of John Woo's iconic
Face/Off, with awesomely bad mainstays Nicolas Cage and John Travolta in
the leads. Sean Archer, FBI agent and family man, undergoes advanced plastic
surgery that gives him the face of his arch nemesis, the comatose terrorist
Caster Troy. However, Troy soon awakens and forces scientists to grant him
the face of Archer. What follows is an unbearably entertaining display of
action sequences and one-liners that promises to be worth your time.

-Brian Tremml
```

As you can imagine even if we all love Nicolas Cage, today's subjest isn't related at all.

Chapter III

Exercise 00 : display_file

	Exercise 00
display_file	
Turn-in directory : *ex*00/	
Files to turn in : `Makefile, and files needed for your program`	
Allowed functions : `close, open, read, write`	
Notes : `n/a`	

- Create a program called `ft_display_file` that displays, on the standard output, only the content of the file given as argument.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`. The binary will be called `ft_display_file`.
- The `malloc` function is forbidden. You can only do this exercise by declaring a fixed-sized array.
- All files given as arguments will be valid.
- Error messages have to be displayed on their reserved output.

```
$> ./ft_display_file
File name missing.
$> ./ft_display_file Makefile
*contenu du Makefile*
$> ./ft_display_file Makefile display_file.c
Too many arguments.
$>
```

Chapter IV

Exercise 01 : cat

	Exercise 01
cat	
Turn-in directory : *ex*01/	
Files to turn in : `Makefile, and files needed for your program`	
Allowed functions : `close, open, read, write`	
Notes : `n/a`	

- Create a program called `ft_cat` which does the same thing as the system's `cat` command-line.
- You don't have to handle options.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno` (check the `man` for `Errno`).
- You can only do this exercise by declaring a fixed-sized array. This array will have a size limited to a little less than `30 ko`. In order to test that size-limit, use the `limit` command-line in your Shell.

```
$> limit stacksize 32
$> limit stacksize
stacksize 32 kbytes
$>
```

Chapter V

Exercise 02 : tail

	Exercise 02
tail	
Turn-in directory : *ex*02/	
Files to turn in : `Makefile, and files needed for your program`	
Allowed functions : `close, open, read, write, malloc, free`	
Notes : `n/a`	

- Create a program called `ft_tail` which does the same thing as the system command `tail`, but which takes at least one file as argument.
- The only option you have to handle is `-c`. This option will be present in all tests.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno`.

Chapter VI

Exercise 03 : hexdump

	Exercise 03
hexdump	
Turn-in directory : *ex*03/	
Files to turn in : `Makefile, and files needed for your program`	
Allowed functions : `close, open, read, write, malloc, free`	
Notes : `n/a`	

- Create a program called `ft_hexdump` which does the same thing as the system's `hexdump` command-line.
- The only option you have to handle is `-C`.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno`.

Chapter VII

Exercise 04 : last

	Exercise 04
last	
Turn-in directory : *ex*04/	
Files to turn in : `Makefile, and files needed for your program`	
Allowed functions : `close, open, read, write, malloc, free`	
Notes : `n/a`	

- Create a program called `ft_last` and that does the same thing as the system's command-line : `last`, without options.

- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.

- You're only allowed to `include <utmp.h>`, `<time.h>`, `<errno.h>` and `<sys/types.h>`

- You cannot use functions such as `ctime()`, `asctime()`, `stat()`, etc.