Introduction to Computer Programming Using the C Programming Language
LABORATORY EXERCISE 2

OBJECTIVES
At the end of this lesson, you should be able to:

1. identify the different commands used in LINUX environment,
2. use the appropriate command/s for a given problem; and,
3. implement a simple program.

LINUX PROGRAMMING ENVIRONMENT
Under *Linux* there are two interfaces:

- GUI – Graphical User Interface
  - point, click and drag
- CLI – Command Line Interface
  - type commands
  - command prompt(in Windows OS)/Terminal
  - **CASE SENSITIVE**

CLI File and Directory Commands

1. `cd`
   - The **cd** command changes directories.
   - To navigate into the root directory, type:
     - `cd /`
   - `To navigate to your home directory, type:`
     - `cd or cd ~`
   - `To navigate up one directory level, type:`
     - `cd ..`
   - `To navigate to the previous directory (or back), type:`
     - `cd -`

   `To navigate through multiple levels of directory at once, specify the full directory path. For example, type:`
     - `cd /var/www`
2. `pwd`
   - The **pwd** command will show which directory you're located in.
3. `ls`
   - The **ls** command shows you the files in your current directory.
4. `cp`
   - The **cp** command makes a copy of a file for you.
5. `mv`
   - The **mv** command moves a file to a different location or will rename a file.
6. `rm`
   - The **rm** command to remove or delete a file in your directory
7. `mkdir`
   - The **mkdir** command will allow you to create directories.

`Other Useful Things: Save on Typing`
`Here are some useful ways on pasting commands.`

Up Arrow	Scrolls through the command you've entered previously.
Down Arrow	Takes you back to a more recent command.
Tab	It *autocompletes* any commands or filenames, if there's only one option, or else gives you a list of options.

EXERCISE:

`1. In the home directory create a folder named CMSC11_<Section> (e.g. CMSC11_D1L)`

`2. Create a folder named <SURNAME>_EXER (e.g. BENNET_EXER) inside the CMSC11<Section> folder.`

`3. Create a folder named EXTRA inside the CMSC11_<Section> folder.`

`4. Create a text file named <initials>.txt (e.g. abc.txt) inside the CMSC11_<Section> folder.`

`5. Move <initials>.txt in the folder <SURNAME>_EXER.`

`6. Delete the EXTRA folder.`

`A FIRST PROGRAM`

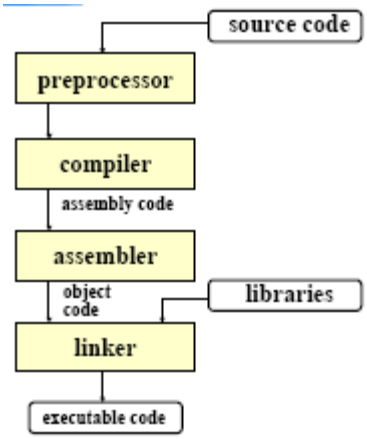
```
/* Author: Maverick C. Crisostomo
   Date created: November 17, 2008
*/
#include <stdio.h>

main( )
{
       printf("\nHello World!\n");
}
```

- Use any text editor to type your first program. You may use `kate, kwrite` or `gedit` text editors.
- After typing the code, save the file in your own folder using the filename:
  `hello.c`
- In the *`Terminal`*, make sure that you are in the directory where you have saved the file. Then, compile it. To compile, type:
  `gcc -o hello hello.c`
- Finally, execute the program simply by typing
  `./hello`
- You should see the words "Hello World" printed out on the *`Terminal.`*

`THE C COMPILATION MODEL`

- The Preprocessor accepts source code as input and
  - removes comments
  - extends the code according to the preprocessor directives included in the source code (lines starting with #)

- The Compiler takes the output of the preprocessor and produces assembly code
- The Assembler takes the assembly code and produces machine code (or object code)
- The Linker takes the object code, joins it with other pieces of object code and libraries and produces code that can be executed



STRUCTURE of a C PROGRAM

- C program contains the following elements:
  - Preprocessor Commands
  - Type Definitions
  - Function Prototypes
  - Variables
  - Functions
- All programs must contain a single *main( )* function. All functions, including *main*, have the following format:

```
type function_name ( parameters ) {
       local variables
       statements
}
```