INTRODUCTION TO C PROGRAMMING IN RASPBIAN

Objective

Students will be capable of writing, debugging, and running a C code program on the Raspbian environment

Procedure

Activity 1

1. First of all, we need to make sure that your Ubuntu version is fully updated with the essential drivers and libraries. Open a Terminal and enter the following command:
```
$ sudo apt-get update
[Enter the admin password if needed]
```
2. Let's see if our Linux version has the GCC files needed to run C code. Enter the next commands:
```
$ whereis gcc
$ which gcc
$ gcc –version
```
3. **IF you got an error saying that there is no GCC program then you need to install it.** To do so, type in the following commands:
```
$ sudo apt-get install build-essential manpages-dev
```
4. Check again with the command in the step 2.
5. Now on the Terminal, move to the "*Documents*" directory (using "*cd*" or "../.." to move around) and create a new folder with the command "*mkdir*" (name it "*Lab02*") and go inside it. **NOTE:** you can check your location using the command "*pwd*"
6. Create a file with the name "*first.c*" using the commands "*gedit*" or "*vi*".
7. Add the following code:
```
#include<stdio.h>
int main(void)
{
     printf("Hello World!\n");
     return 0;
}
```
8. Save the code and exit to return to the Terminal.
   Now, is time to test the code. From Terminal, insert the next command:

```
$ gcc -o First first.c
```

9. And to run it, you do it like this:

```
$ ./First
```

Activity 2

10. Write a code with the name "*rectangle.c*", in which you enter parameters in order to calculate the area or perimeter according to the values that you chose. For example:
    The next command:

```
$ ./Rectangle -a -l 10 -w 5
```

    Sends the result:

```
$ area = 50 units
```

    The next command:

```
$ ./Rectangle -p -l 10 -w 5
```

    Sends the result:

```
$ perimeter = 30 units
```

11. You can use the following code for context:

```
#include<stdio.h>
int main(int argc, char *argv[])
{
   int i = 0;
   for (i = 0; i < argc; i++) {
   printf("argv[%d] = %s\n", i, argv[i]);
}
   return 0;
}
```

12. To convert to integer use the "atoi" function.

Activity 3

13. Write a code with the name "LittleBlackboard.c", in which you ask the user the names and IDs of a list of students, and then it saves them on a .txt file with the name "Datalog.txt". You should specify a size limit and the program must end itself if you reach the limit. For example:
    The next command:

```
$ ./LittleBlackboard -s 40
```

    Sends the result:

```
$ Enter Name:
[give name][ENTER]
$ Enter ID:
[give ID][ENTER]
$ Do you wish to add more [Y/n]:
[give n][ENTER]
```

```
$ Students information stored in Datalog.txt
```

You can use the following code for context:

```
[…Missing code…]
FILE *f = fopen("file.txt", "w");
if (f == NULL)
{
    fprintf(stderr,"Error opening file!\n");
    exit(1);
}
const char *text = "Write this to the file";
fprintf(f, "Some text: %s\n", text);
fclose(f);
[…Missing code…]
```

You can also use the function "*scanf*".

Report

Turn in a technical report including the following:

- Links to GitHub of Activities 2 and 3
- Links to YouTube with demo videos of your working code
- Conclusions