LABORATORY 1

–

OPERATING SYSTEMS

RUFINO GARCIA SANCHEZ

EXERCISE 1

Write 4 simple programs in C, these programs should have significant number of computation, memory allocation/deallocation, or I/O device access (e.g. reading or writing data in a file) operations, such as the time of their execution is significant enough allowing to measure it.

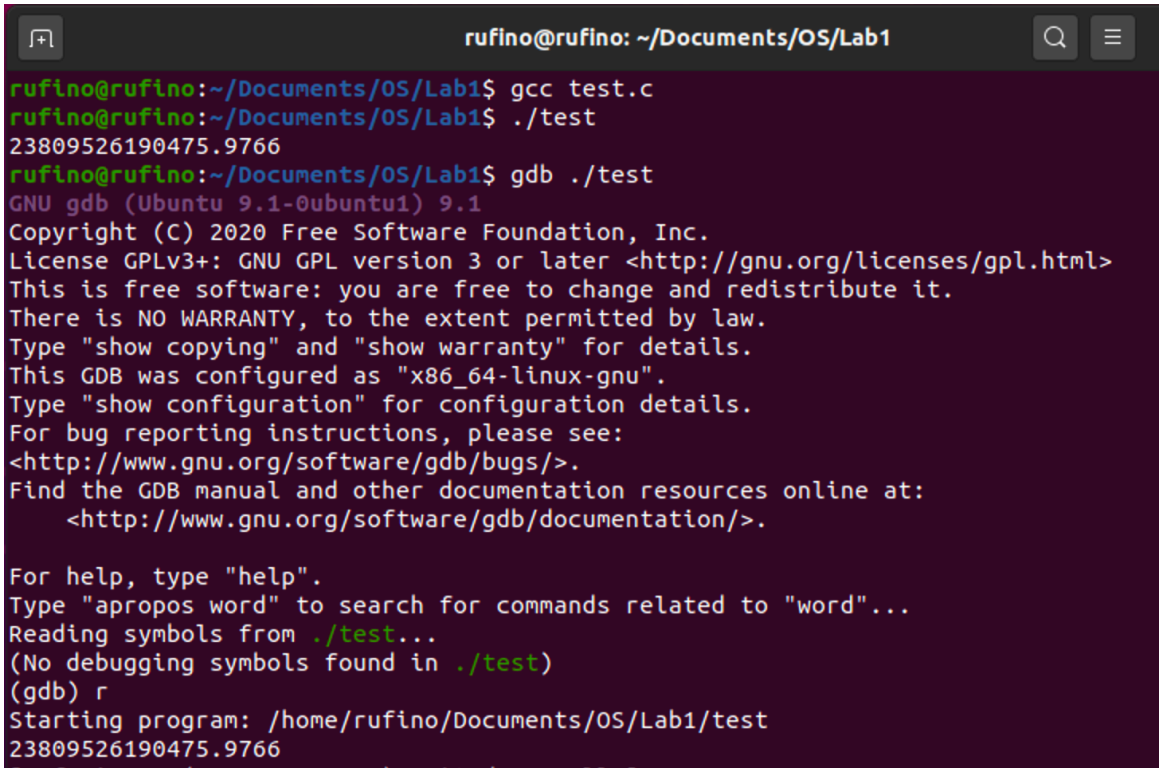
Compile these programs with (at least) f4 different optimization flags, e.g.:

- **O1**: This flag tries to reduce code size and execution time.
- **O2**: This flag optimize even more. Reduce more than -O.
- **O3**: This flag reduces more than -O2.
- **Os**: This flag is used only to optimize the size.

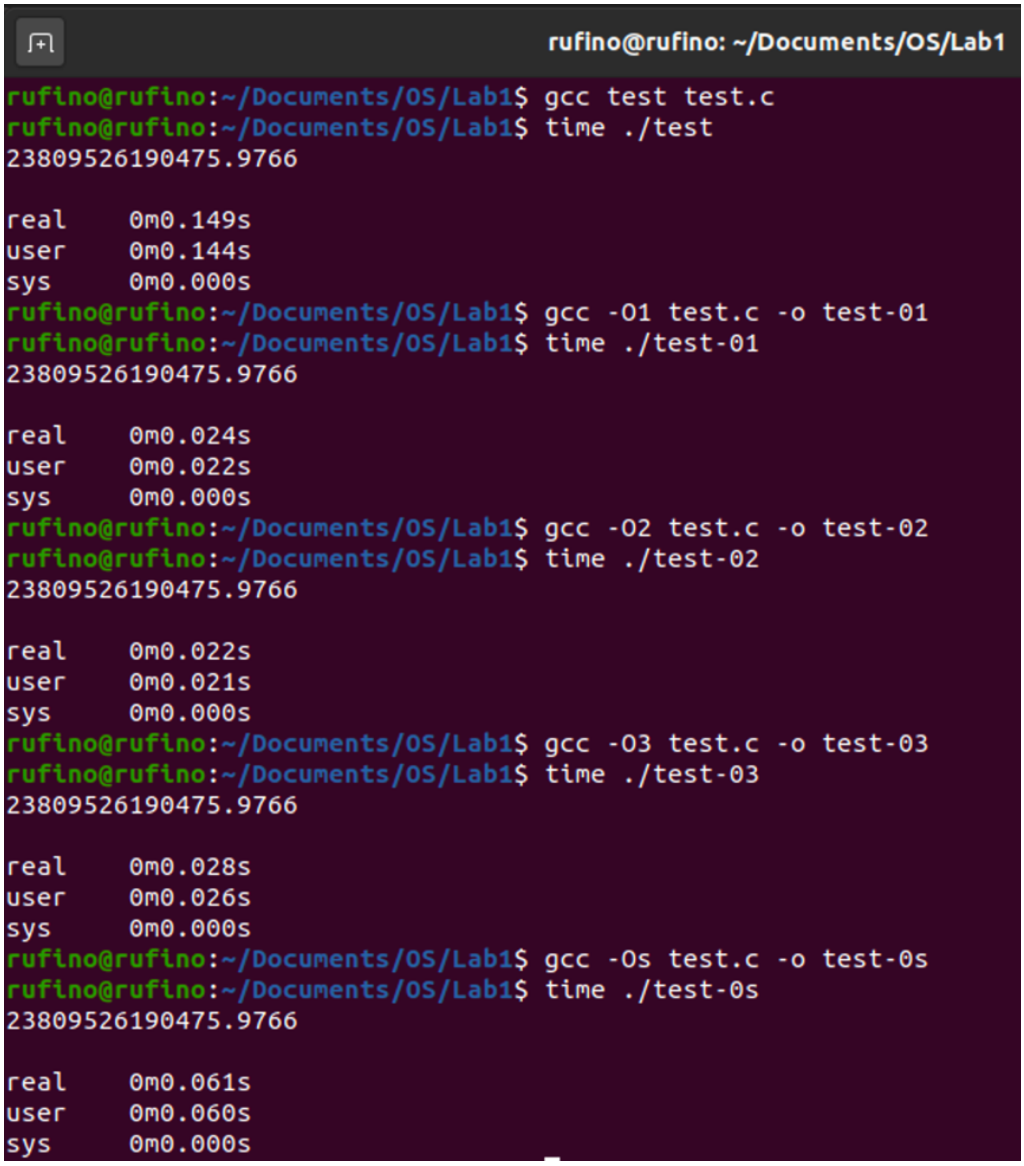
Calculate real/user/system times of these programs without optimization and with all given above optimization flags using *time* program.

My first program is called **test**. First what I have done in all is compile the program with **gcc** and then execute with **gdb** with the command **r or run**. As we can see in the next image.

This first program called test return the sum of two floats that are in an loop for and when the variable of the loop its less than that sum the loop breaks and then the value is returned.



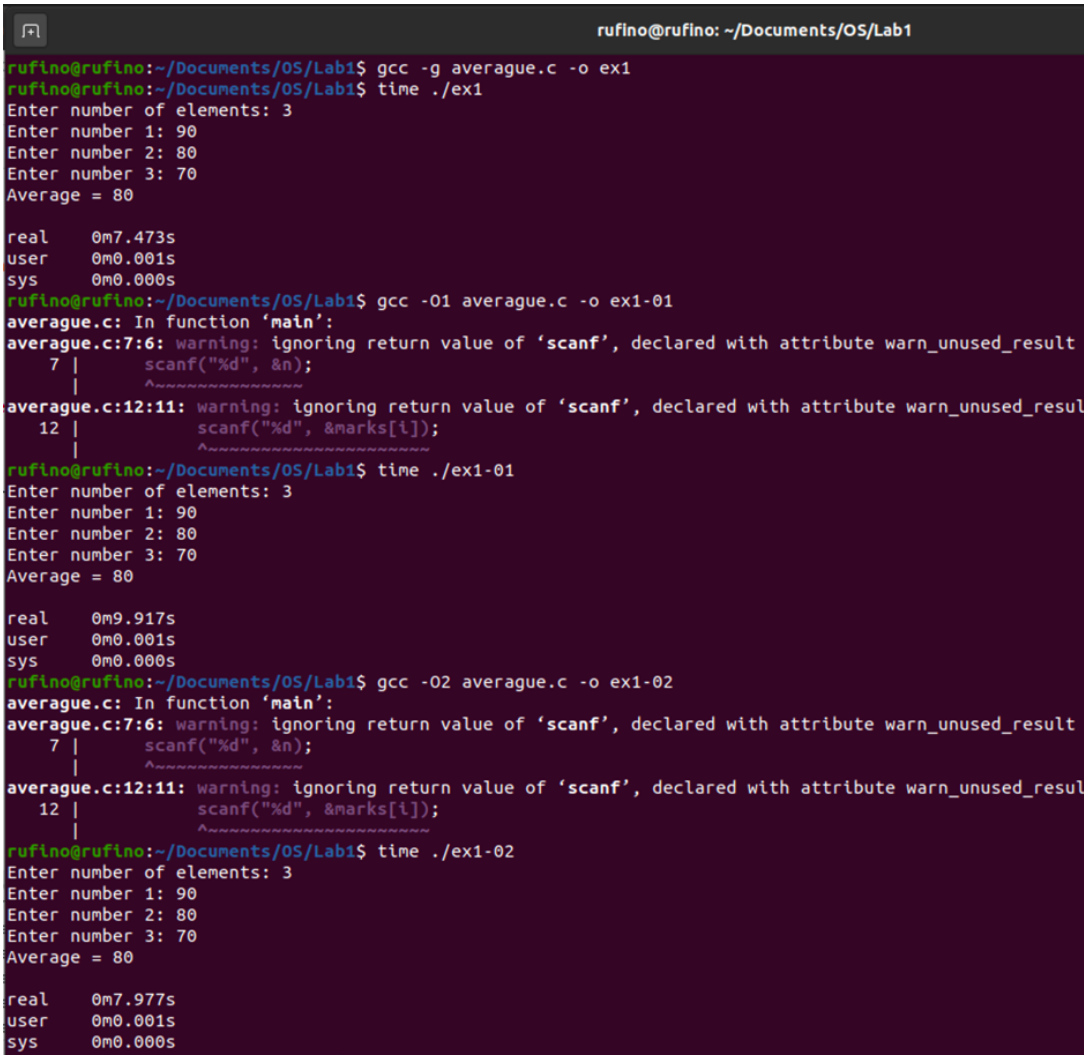
In this next photo we can observe the **real/user/system** times of this program without optimization and with all given above optimization flags

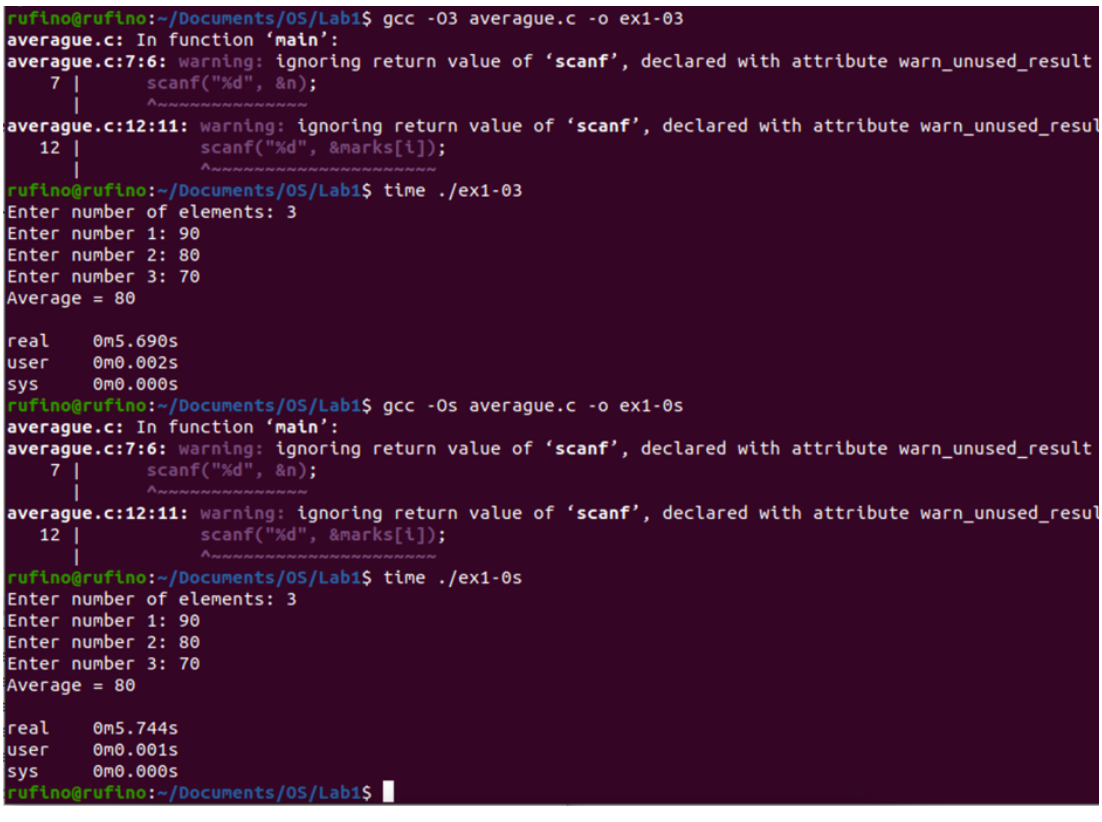


As we can see when we are using the **Optimizations flags** we obtain that the real and the user times **decrease** a lot if we compare to the time we have obtained in the firs execution. That's why we are optimizing the code using that **Optimizations flags.**
We can observe too that the time in the **Os** execution increase, why? Because this flag only want to reduce the size no the time like O1,O2 and O3.

We are going to watch some images for the **second program.**

This second program is called **averague.c.** The main function of this program is that he calculates the average of N (it depends on how many numbers you choose) numbers you introduce in the terminal and it depend of the quantity you give to that numbers.

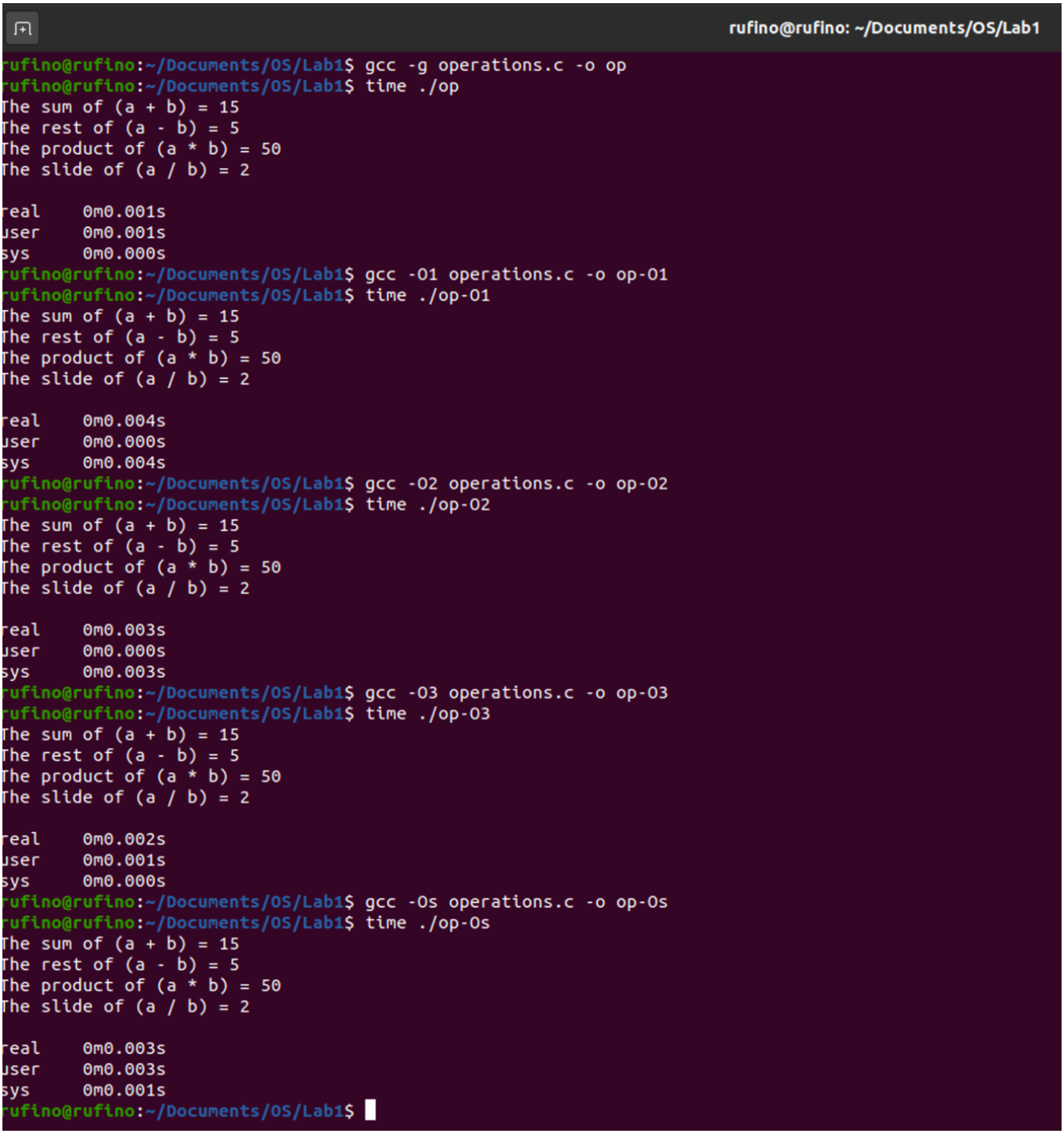

rufino@rufino: ~/Documents/OS/Lab1
rufino@rufino:~/Documents/OS/Lab1$ gcc -g averague.c -o ex1
rufino@rufino:~/Documents/OS/Lab1$ time ./ex1
Enter number of elements: 3
Enter number 1: 90
Enter number 2: 80
Enter number 3: 70
Average = 80
real 0m7.473s
user 0m0.001s
sys 0m0.000s
rufino@rufino:~/Documents/OS/Lab1$ gcc -O1 averague.c -o ex1-01
averague.c: In function 'main':
averague.c:7:6: warning: ignoring return value of 'scanf', declared with attribute warn_unused_result
7 | scanf("%d", &n);
averague.c:12:11: warning: ignoring return value of 'scanf', declared with attribute warn_unused_resul
12 | scanf("%d", &marks[i]);
rufino@rufino:~/Documents/OS/Lab1$ time ./ex1-01
Enter number of elements: 3
Enter number 1: 90
Enter number 2: 80
Enter number 3: 70
Average = 80
real 0m9.917s
user 0m0.001s
sys 0m0.000s
rufino@rufino:~/Documents/OS/Lab1$ gcc -O2 averague.c -o ex1-02
averague.c: In function 'main':
averague.c:7:6: warning: ignoring return value of 'scanf', declared with attribute warn_unused_result
7 | scanf("%d", &n);
averague.c:12:11: warning: ignoring return value of 'scanf', declared with attribute warn_unused_resul
12 | scanf("%d", &marks[i]);
rufino@rufino:~/Documents/OS/Lab1$ time ./ex1-02
Enter number of elements: 3
Enter number 1: 90
Enter number 2: 80
Enter number 3: 70
Average = 80
real 0m7.977s
user 0m0.001s
sys 0m0.000s


rufino@rufino:~/Documents/OS/Lab1$ gcc -O3 averague.c -o ex1-03
averague.c: In function 'main':
averague.c:7:6: warning: ignoring return value of 'scanf', declared with attribute warn_unused_result
7 | scanf("%d", &n);
averague.c:12:11: warning: ignoring return value of 'scanf', declared with attribute warn_unused_resul
12 | scanf("%d", &marks[i]);
rufino@rufino:~/Documents/OS/Lab1$ time ./ex1-03
Enter number of elements: 3
Enter number 1: 90
Enter number 2: 80
Enter number 3: 70
Average = 80
real 0m5.690s
user 0m0.002s
sys 0m0.000s
rufino@rufino:~/Documents/OS/Lab1$ gcc -Os averague.c -o ex1-0s
averague.c: In function 'main':
averague.c:7:6: warning: ignoring return value of 'scanf', declared with attribute warn_unused_result
7 | scanf("%d", &n);
averague.c:12:11: warning: ignoring return value of 'scanf', declared with attribute warn_unused_resul
12 | scanf("%d", &marks[i]);
rufino@rufino:~/Documents/OS/Lab1$ time ./ex1-0s
Enter number of elements: 3
Enter number 1: 90
Enter number 2: 80
Enter number 3: 70
Average = 80
real 0m5.744s
user 0m0.001s
sys 0m0.000s
rufino@rufino:~/Documents/OS/Lab1$

We can observe that the flags that reduces the time are the flags that optimizes the size and not the code.

For the **Third program** we are going to watch the next images. This third program is called **operations.c.**

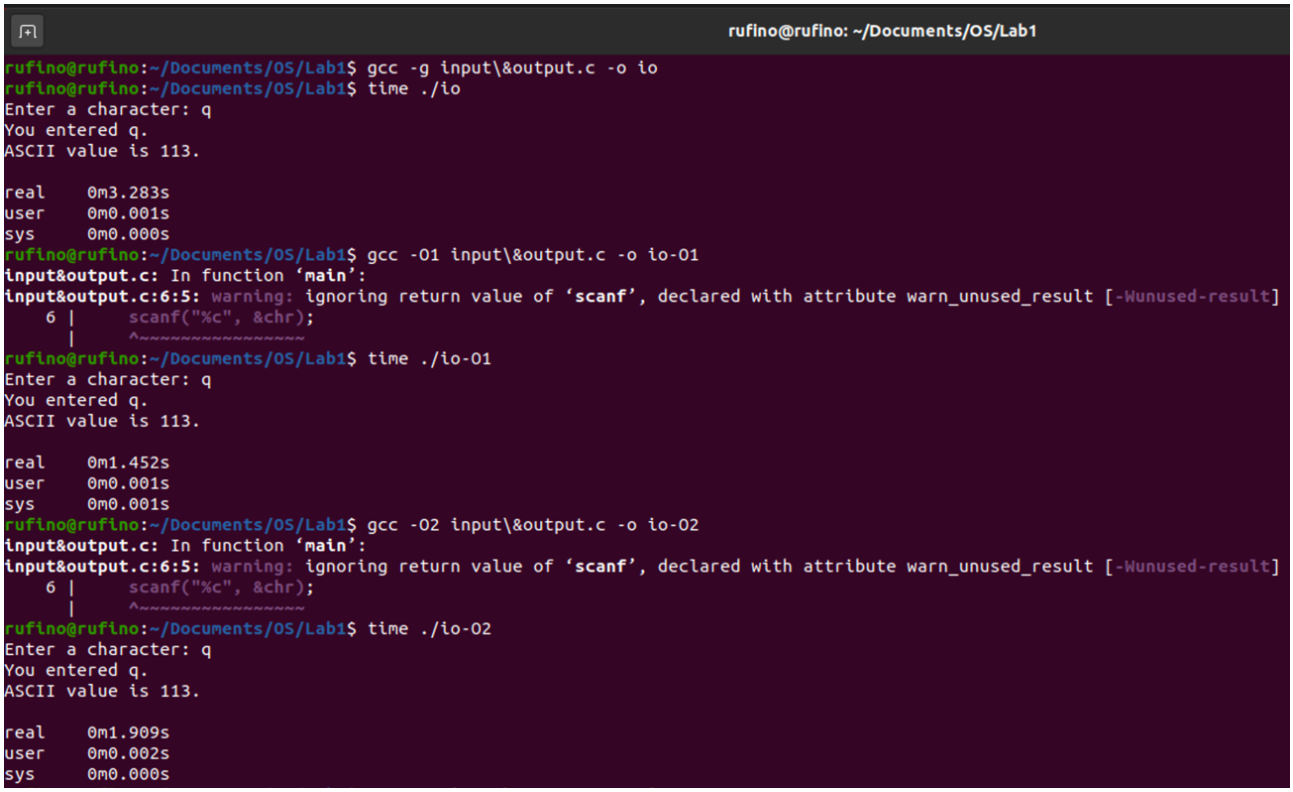
This program consists in different operations with different variables that have an integer value.

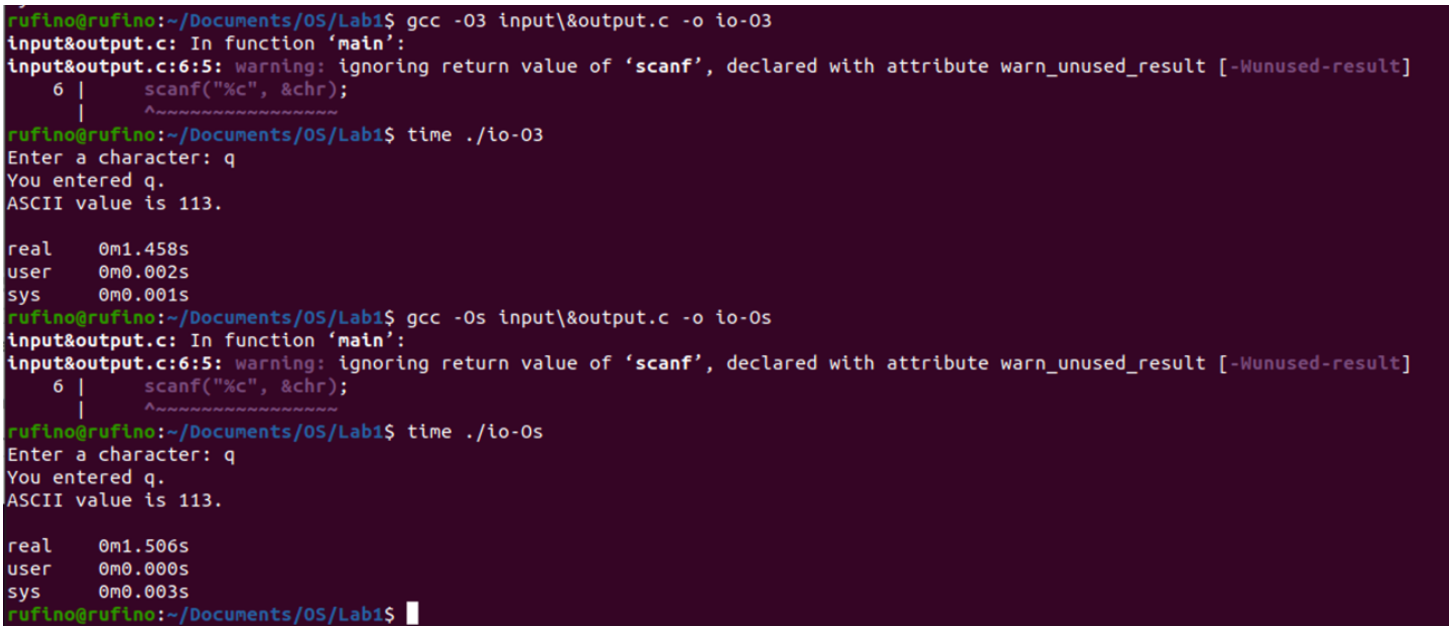


As we can observe the times are very similar the change it is almost imperceptible.

For the **fourth program** we are going to observe the next screenshots. This fourth program is called **inputoutput.c**.

This program consists in introduce something in the terminal and they will give to us the value of the characters we have introduce in the ASCII table.





We can observe that the Optimizations flags in this case works correctly and they decrease the time.

We can observe that in a first moment we had a real time superior of 3 second and when we introduction these fags the time is reduced at 1 second so we can observe that they optimize the code. They work correctly optimizing the code and the size of this.