|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Ing. García Morales Karina. |
| *Asignatura:* | Fundamentos de programación. |
| *Grupo:* | 22 |
| *No de Práctica(s):* | 7 |
| *Integrante(s):* | Dueñas Jarvio Pablo Alam |
|  |  |
| *Semestre:* | 2018-2 |
| *Fecha de entrega:* | 10/04/2018 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_

Objetivo:

Elaborar programas en lenguaje C utilizando las instrucciones de control de tipo secuencia,

para realizar la declaración de variables de diferentes tipos de datos, así como efectuar

llamadas a funciones externas de entrada y salida para asignar y mostrar valores de

variables y expresiones.

Desarrollo.

*Etapas de codificación.*

Para crear un programa en C se siguen tres etapas principales: edición, compilación y

ejecución.

* Edición: Se escribe el código fuente en lenguaje C desde algún editor de textos.
* Compilación: A partir del código fuente (lenguaje C) se genera el archivo en lenguaje máquina (se crea el programa objeto o ejecutable).
* Ejecución: El archivo en lenguaje máquina se puede ejecutar en la arquitectura

correspondiente.

*Ciclo de vida del software.*

El término ciclo de vida del software describe el desarrollo de software, desde la fase inicial hasta la fase final. El propósito de este programa es definir las distintas fases intermedias que se requieren para validar el desarrollo de la aplicación, es decir, para garantizar que el software cumpla los requisitos para la aplicación y verificación de los procedimientos de desarrollo: se asegura de que los métodos utilizados son apropiados.

*Estructuras.*

Las estructuras son colecciones de variables relacionadas bajo un nombre.

Las estructuras pueden contener variables de muchos tipos diferentes de datos

a diferencia de los arreglos que contienen únicamente elementos de un mismo

tipo de datos.

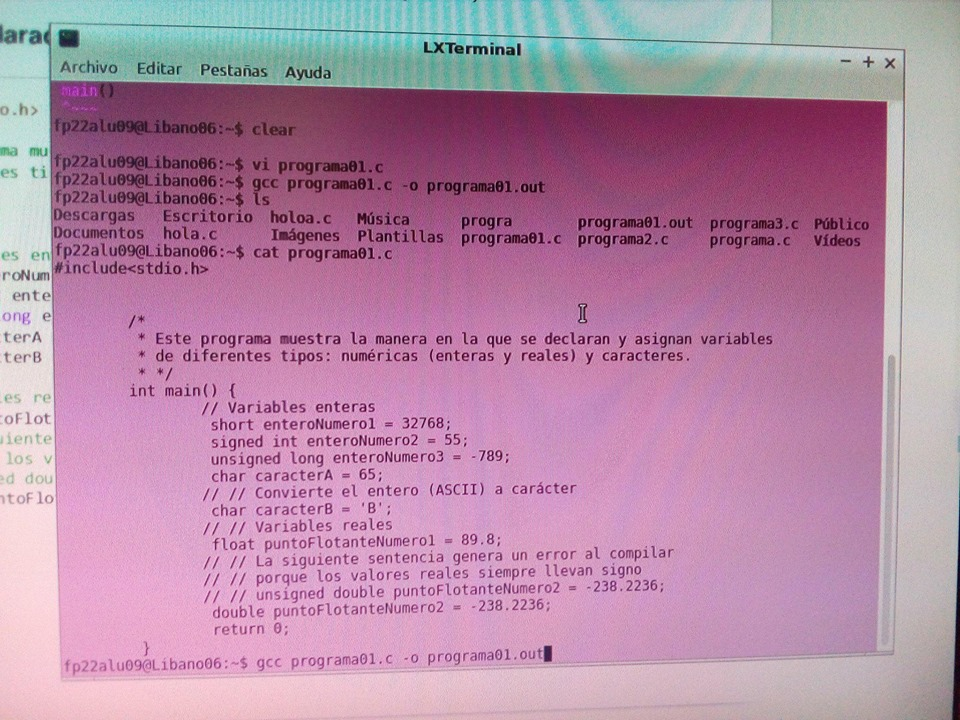
*Tipos de datos.*

|  |  |
| --- | --- |
| Tipo de dato. | Especificador de formato. |
| ENTERO | %ld, %li, %i, %d |
| FLOTANTE (REAL) | %lf, %c, %g, %f, %2f |
| CARACTER | %d, %i, %o, %x, %c, %s |

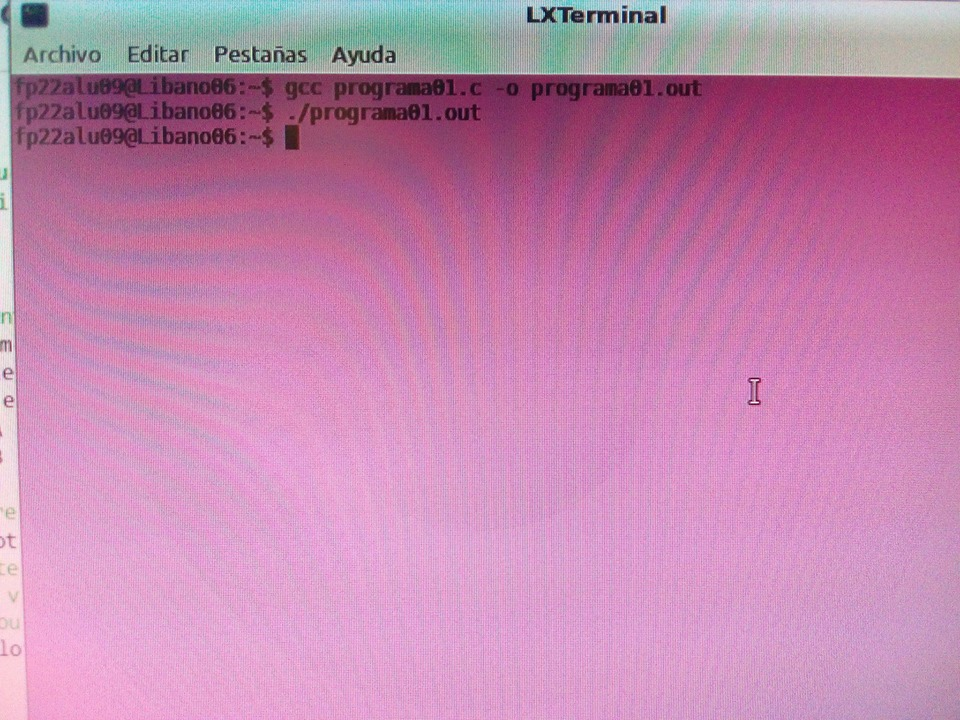
*Lenguajes de programación estructurado.*

Algunos de los lenguajes utilizados inicialmente para programación estructurada incluyen ALGOL, Pascal, PL/I y Ada, pero la mayoría de los nuevos lenguajes de programación procedimentales desde entonces han incluido características para fomentar la programación estructurada y a veces, deliberadamente, omiten características en un esfuerzo para hacer más difícil la programación no estructurada.

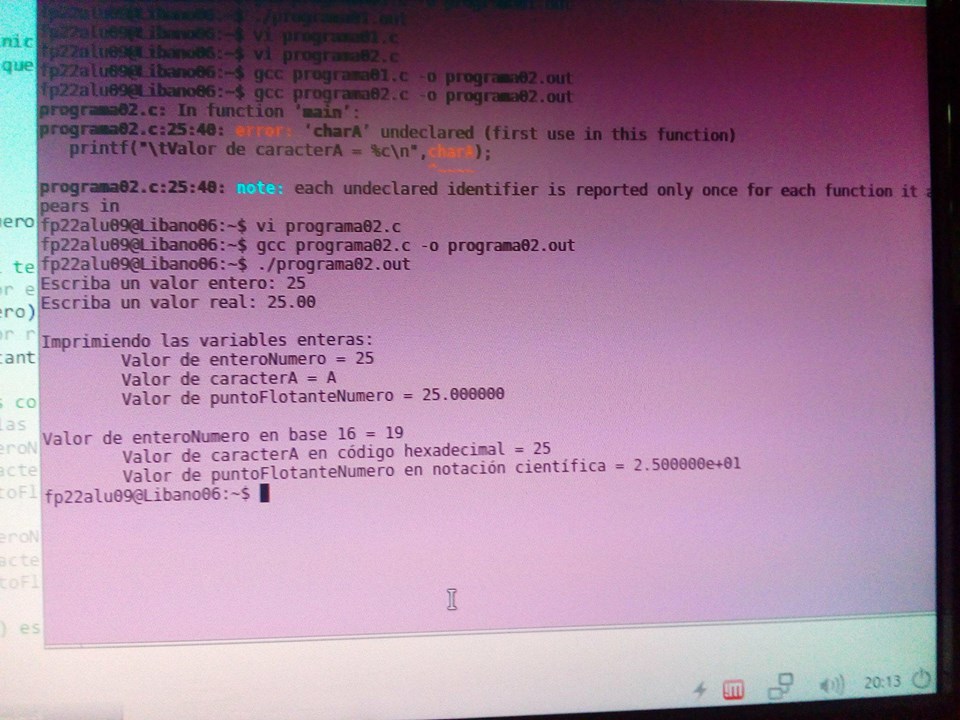
Creamos un archivo llamado programa01.c en vi, y en este mismo, hemos colocado un código para la declaración de variables.



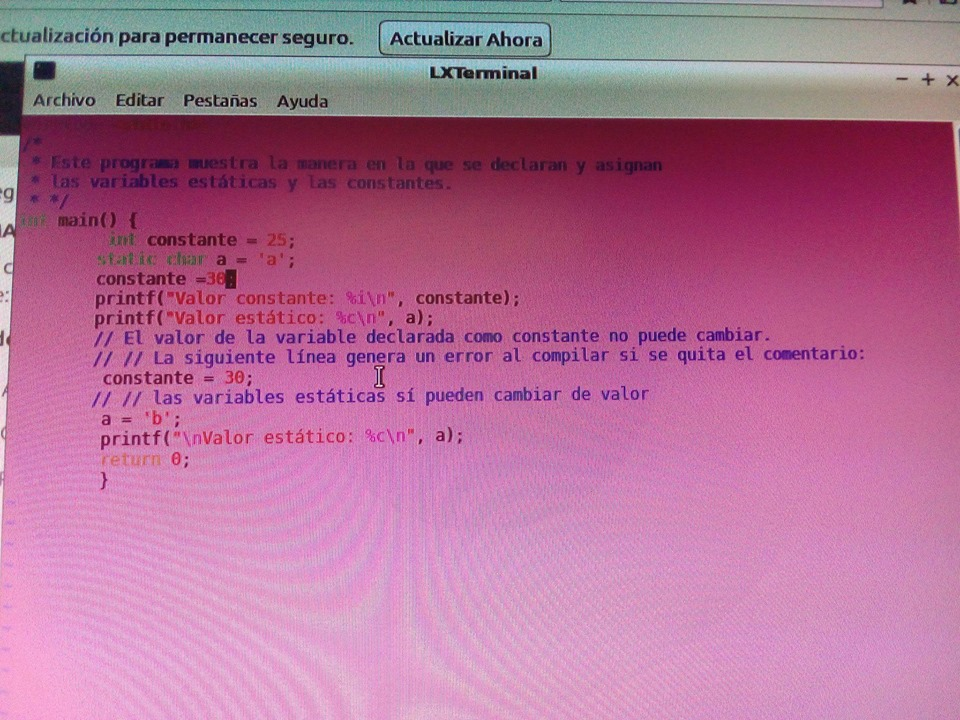
Y posteriormente compilamos y corremos el programa, al no aparecer “nada” en pantalla, quiere decir que todo el programa es correcto.



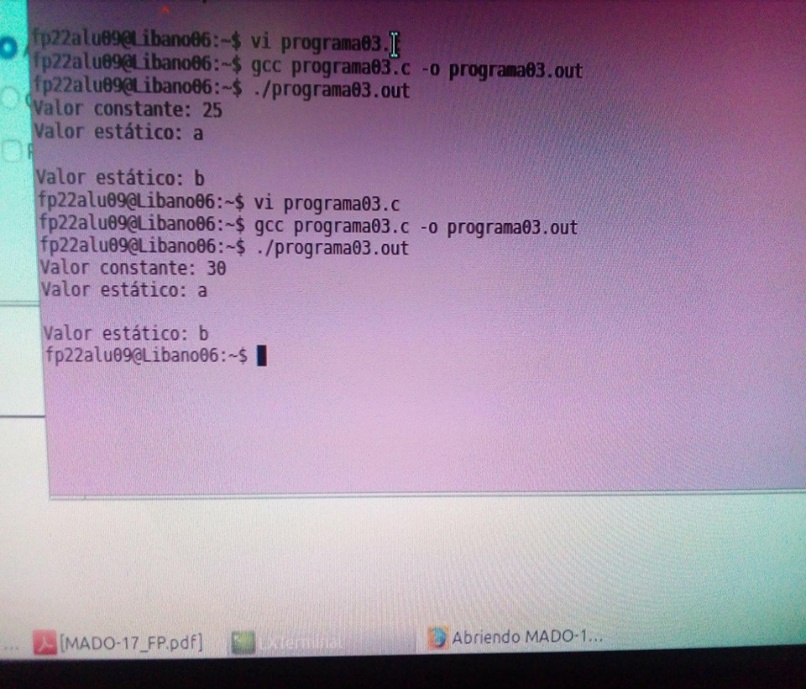
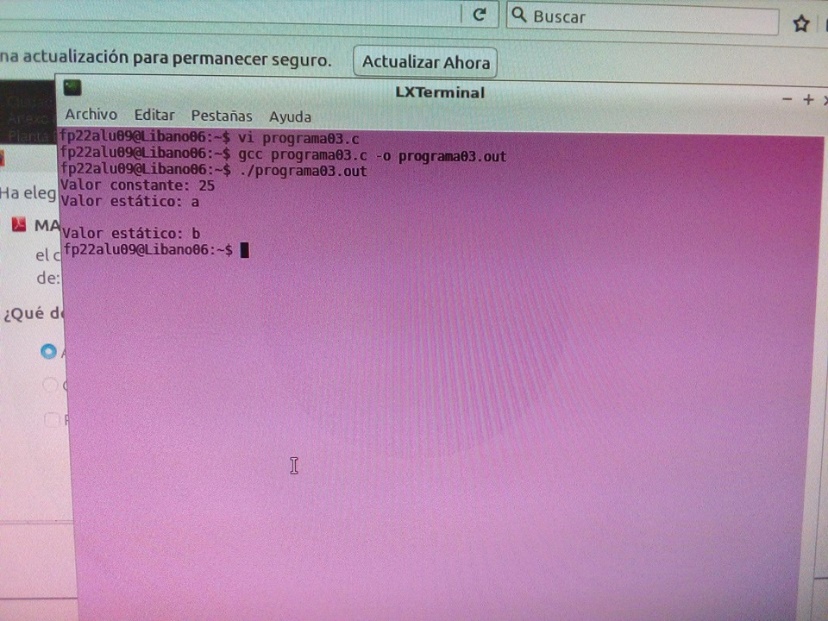
Dentro del mismo archivo, creamos otro programa llamado programa02.c, donde colocamos un código para almacenar e imprimir variables. Este programa muestra la manera en la que se declaran y asignan variables de diferentes tipos: numéricas (enteras y reales) y caracteres, así como la manera en la que se imprimen los diferentes tipos de datos. De igual manera compilamos y corremos el programa.

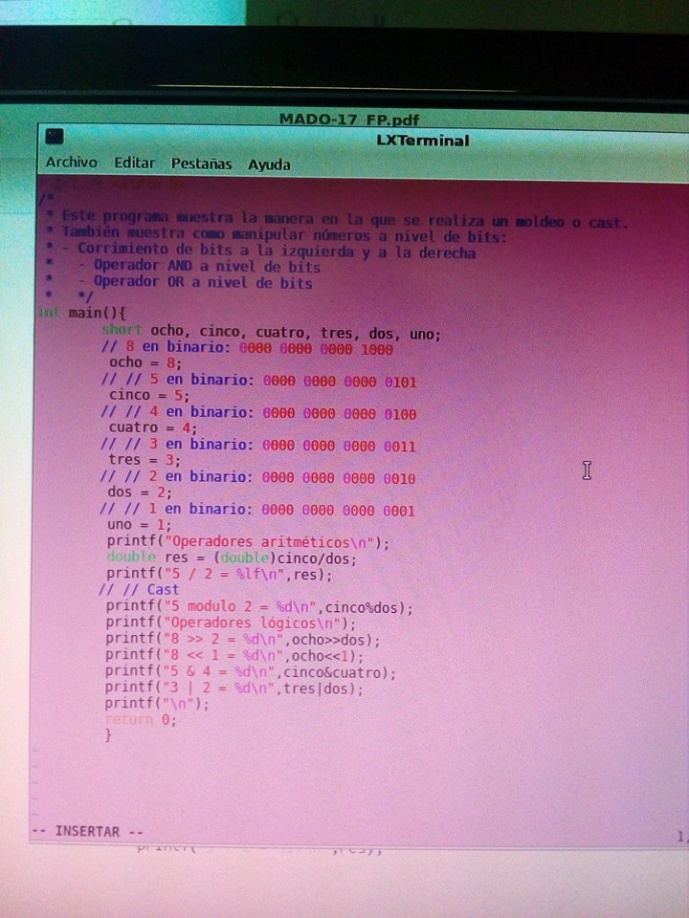


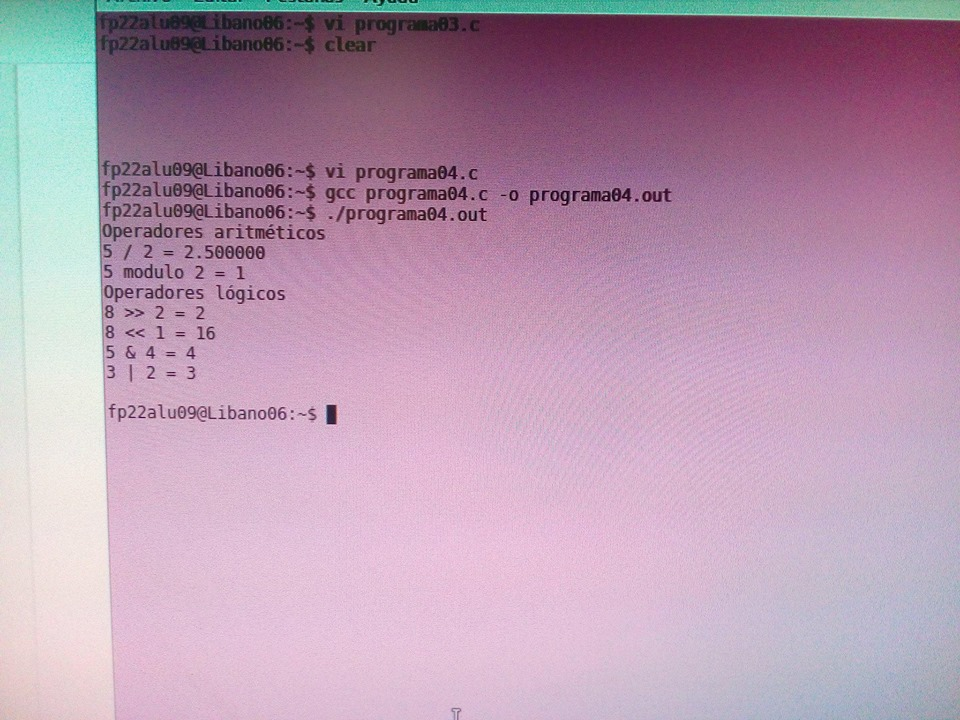
En un nuevo programa llamado programa03.c, codificamos un programa para declarar y asignar variables estáticas y las constantes.



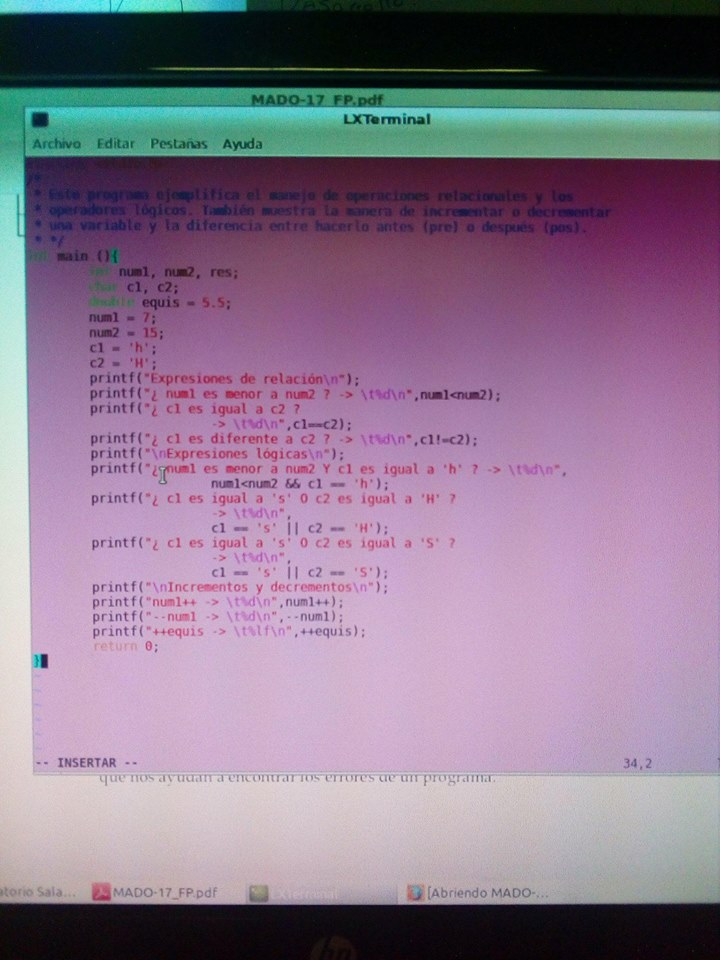
Posteriormente compilamos y corremos el programa.



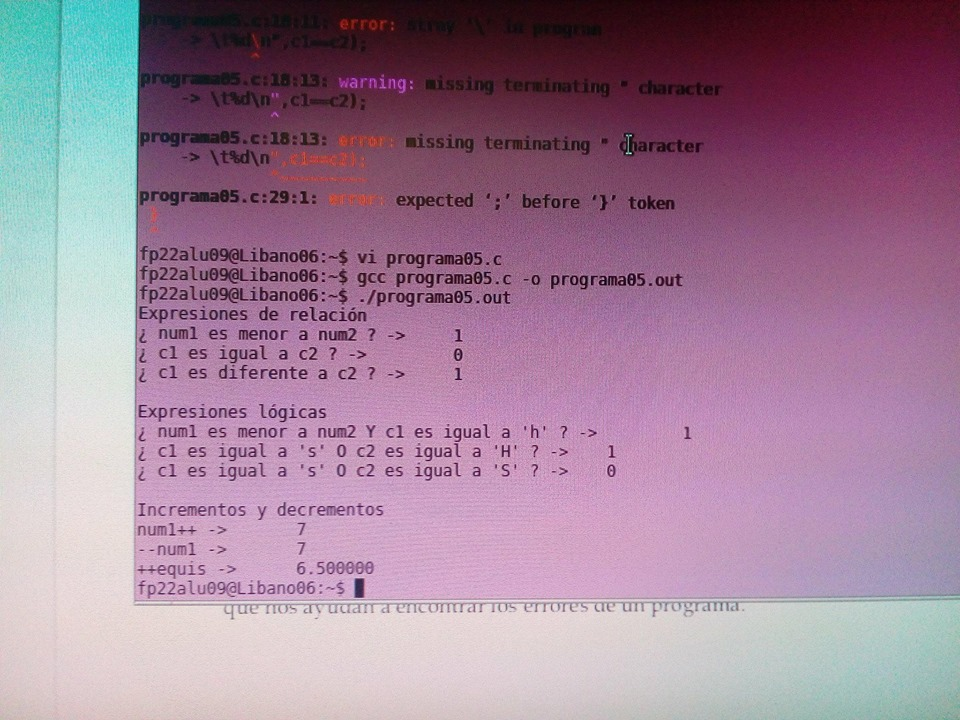
En el cuarto programa, codificamos uno para mostrar códigos operadores. De tal modo que este es el código. Compilamos y ejecutamos el programa, y como podemos ver en la siguiente imagen, corre bien.



Códigos de expresiones lógicas. Este programa ejemplifica el manejo de operaciones relacionales y los operadores lógicos. También muestra la manera de incrementar o decrementar una variable y la diferencia entre hacerlo antes (pre) o después (pos).



Y compilamos y ejecutamos el programa.



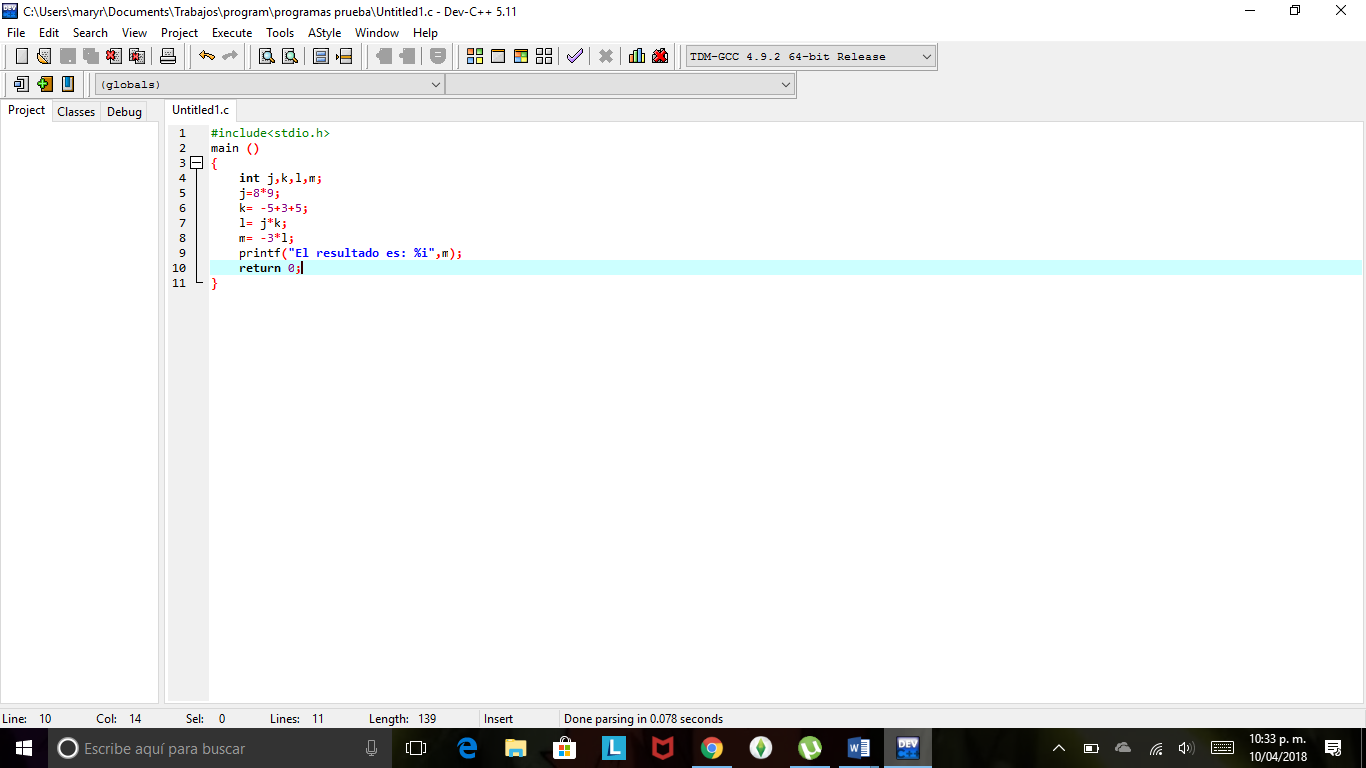
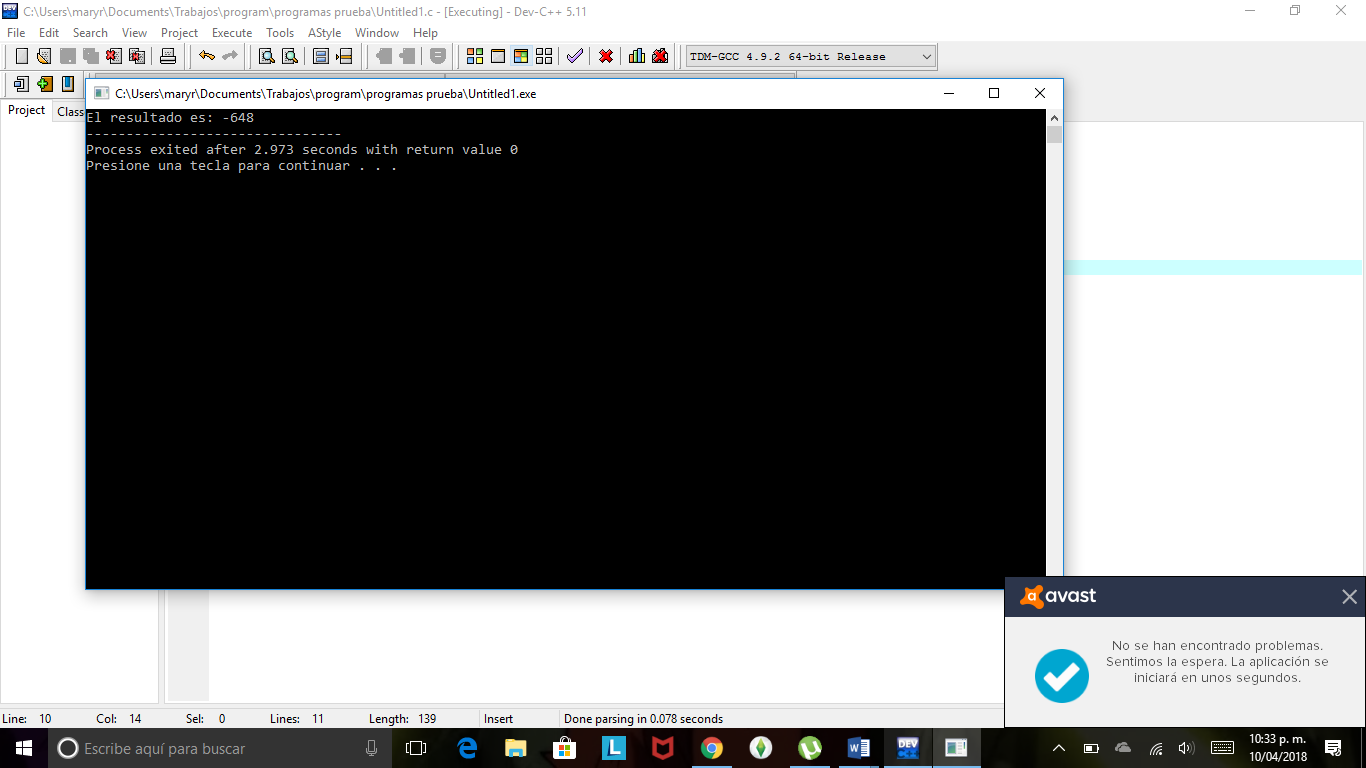
Ejercicios propuestos.

1. Desarrolle un programa que resuelva por pasos la siguiente expresión

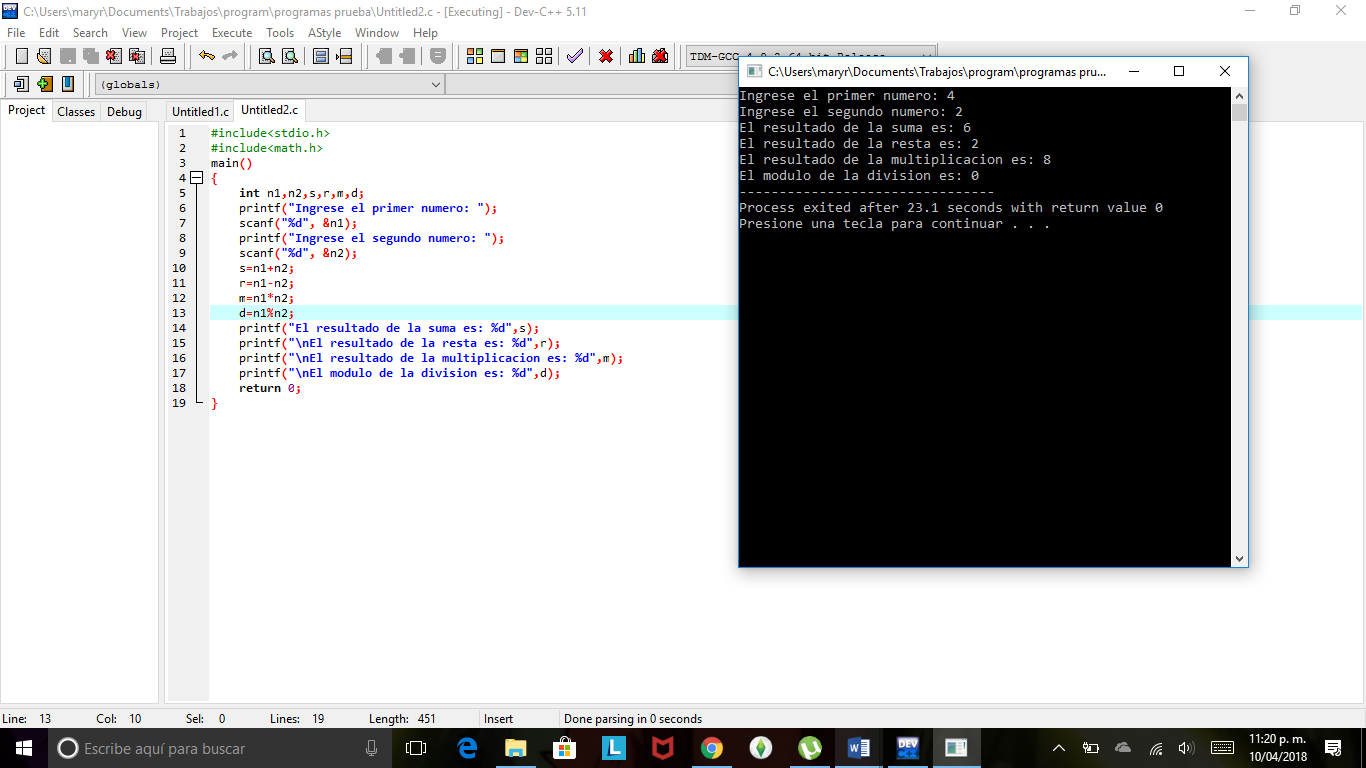
-3[-5+3(8\*9)+5]

Pseudocódigo.

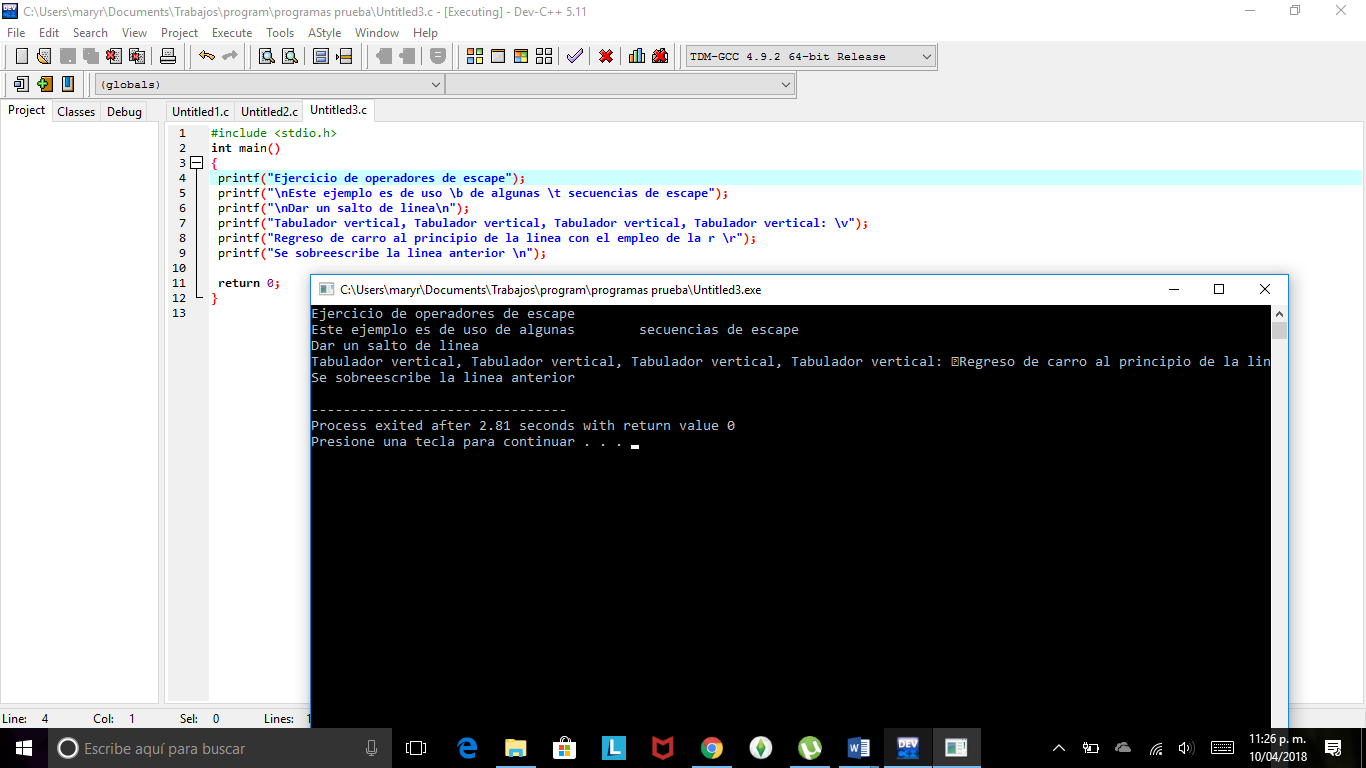
INICIO.  
j,k,l,m: ENTERO  
j:=8\*9  
k:= -5+3+5  
l:= j\*k  
m:= -3\*l  
ESCRIBIR: “El resultado es:”m  
FIN.

3.Programar una calculadora de dos operandos que permita realizar la suma, resta, multiplicación, Modulo y que se muestren en pantalla los resultados.



5.Ejecutar el siguiente código y explicar que hace cada secuencia de caracteres de escape



|  |  |
| --- | --- |
| \t | Tabulador horizontal |
| \n | Salto de línea |
| \v | Tabulador vertical |
| \r | Regreso de carro |
| \f | Avance de hoja |

Conclusiones.

Con esta práctica he mejorado en la cuestión de programar en Linux, de los comandos que sirven para compilar y ejecutar, y también las instrucciones de control de secuencia. Y las llamadas a funciones externas.