|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre de la práctica** | **Estructuras** | | | **No.** |  |
| **Asignatura:** | **Métodos Numéricos** | **Carrera:** | **ISIC** | **Duración de la práctica (Hrs)** |  |

**Emiliano Arce de Jesús Gpo.3402**

1. **Competencias especificas**

**II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):**

1. **Material empleado:**

* **Lenguaje de programación estructurada Dev-C++**
* **Manuel de prácticas FO-aca**

1. **Desarrollo de la práctica:**

**Una estructura de datos esta compueta de elementos individuales que pueden ser del mismo tipo.**

**A cada uno de los elementos de la estructura se le denomina miembro**

**¿Cómo se declara una estructura?**

struct nombreEstructura{

tipoDato1 miembro1;

tipoDato2 miembro2;

tipoDaton miembron;

}

**Los miembros pueden ser de cualquier tipo excepto void**

* **Ejemplo:**

**struct atomo{**

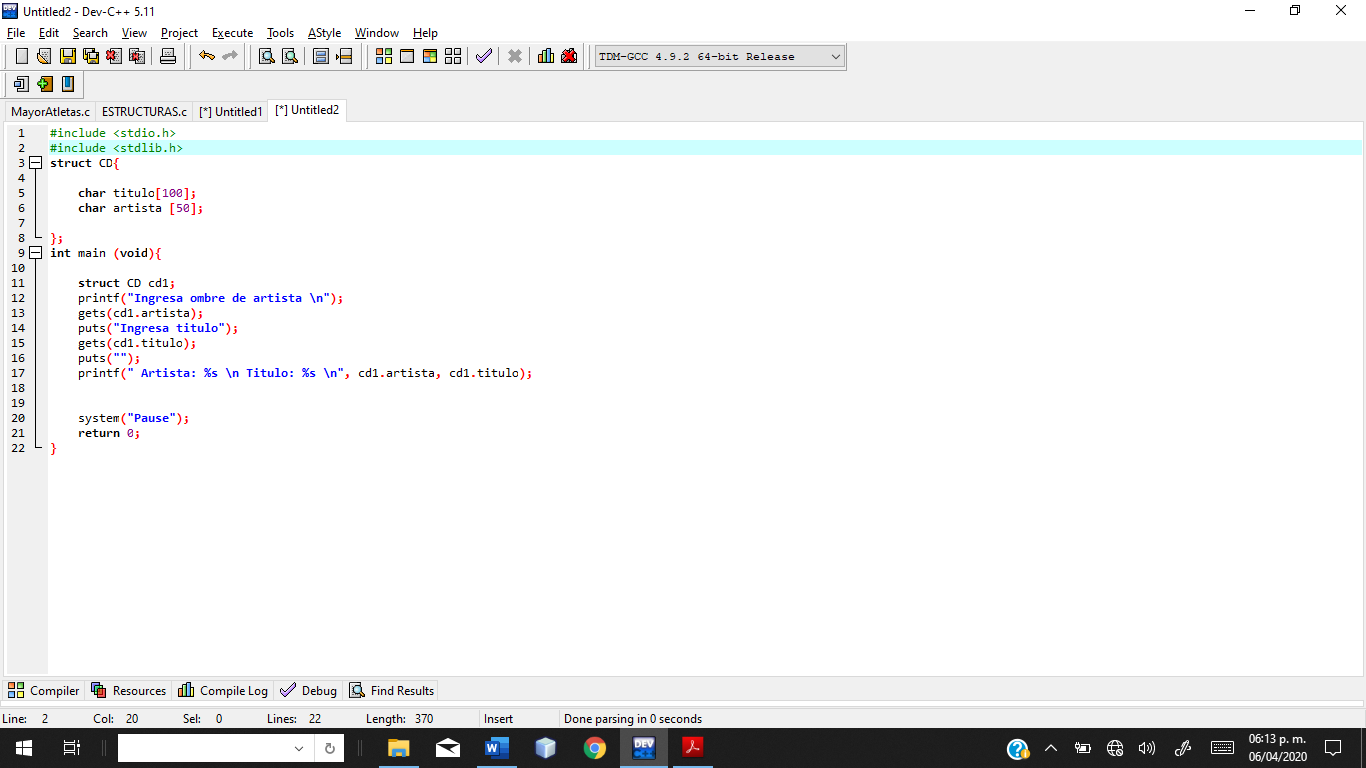
**int hidrogeno;**

**int oxigeno;**

**char nombre[15];**

**}**

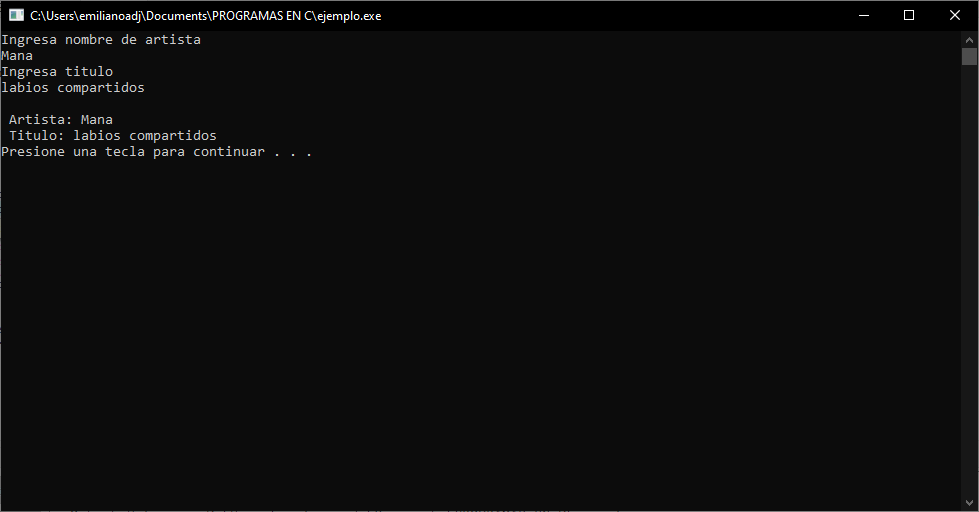
* **Ejemplo de Estructuras**



**En este ejercicio podemos notar que la estructura contiene dos variables de tipo arreglo (char) en las cuales se está guardando el título de una canción y el artista (línea 3-8).**

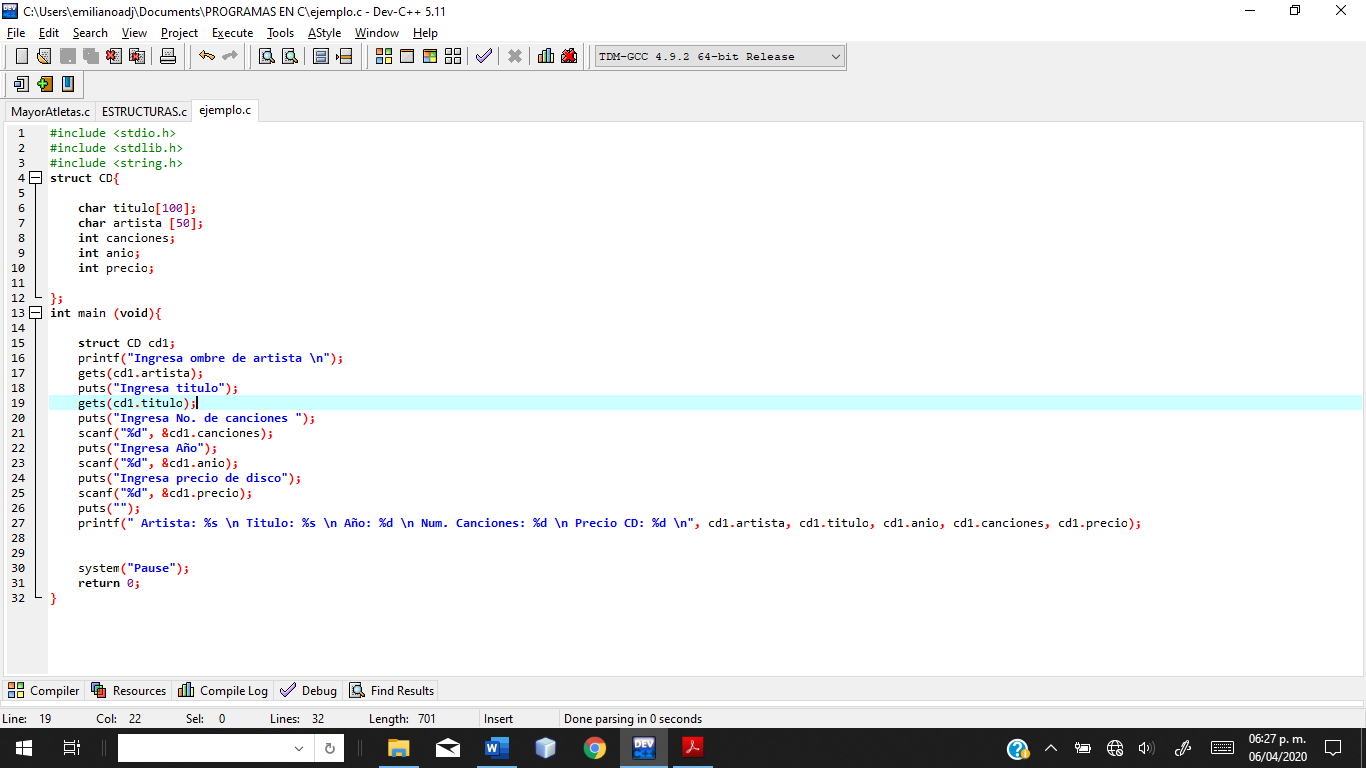
**La estructura se manda llamar en el método principal con la siguiente sintaxis: struct CD cd1; haciendo énfasis en que con cd1 podemos manipular las variables que se encuentran dentro de la estructura (línea 11).**

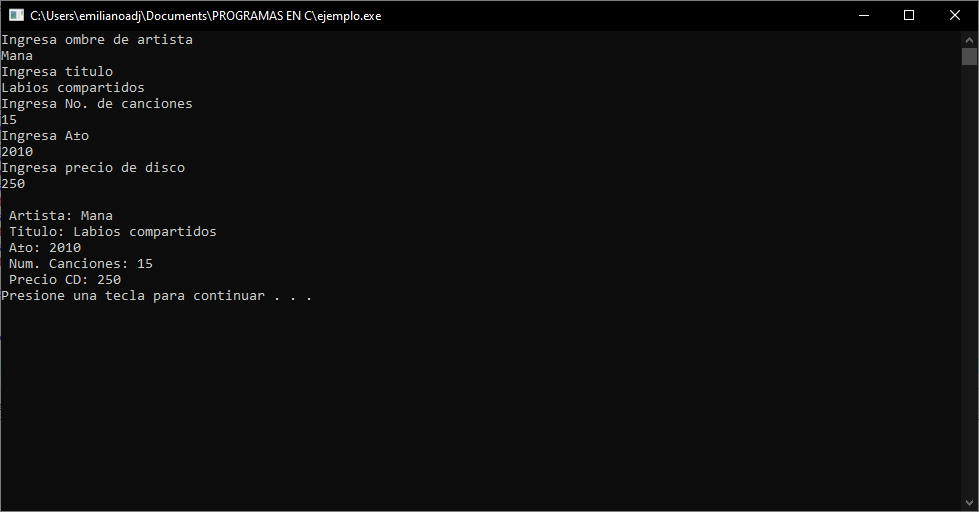
**Se piden los datos correspondientes y finalmente imprime los datos almacenados en el arreglo.**



**Estos son los datos que arroja cuando finaliza el programa.**

* **Modifica el programa para que almacene también numero de canciones, precio y año.**

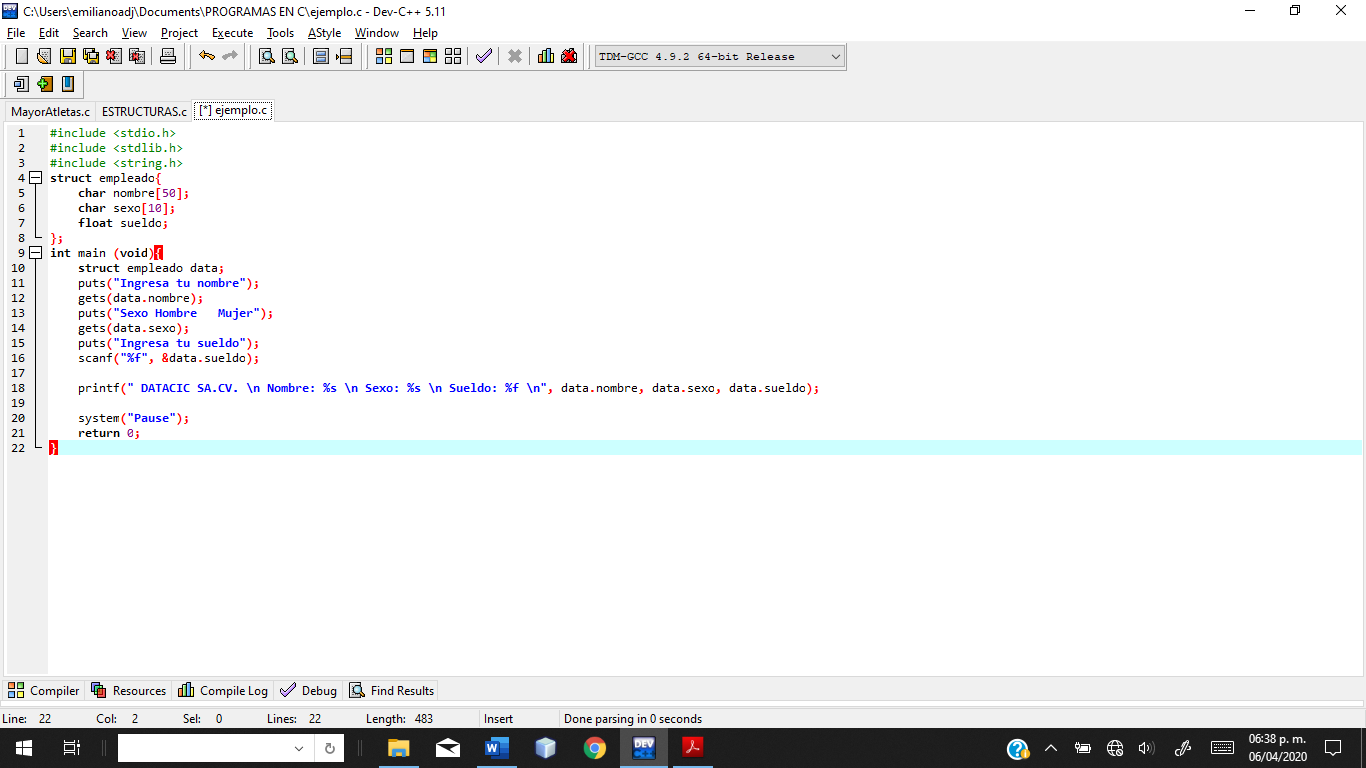




**Se imprimen los datos que se agregaron el la estructura, en el método main se piden los datos correspondientes para luego ser impresos.**

**Los datos fueron introducidos por teclado.**

* **Realiza un programa en C que almacene y muestre la información de un empleado de la empresa DATACIC en una estructura llamada “empleado”. La información con que se cuenta del empleado es: nombre, sexo y sueldo.**

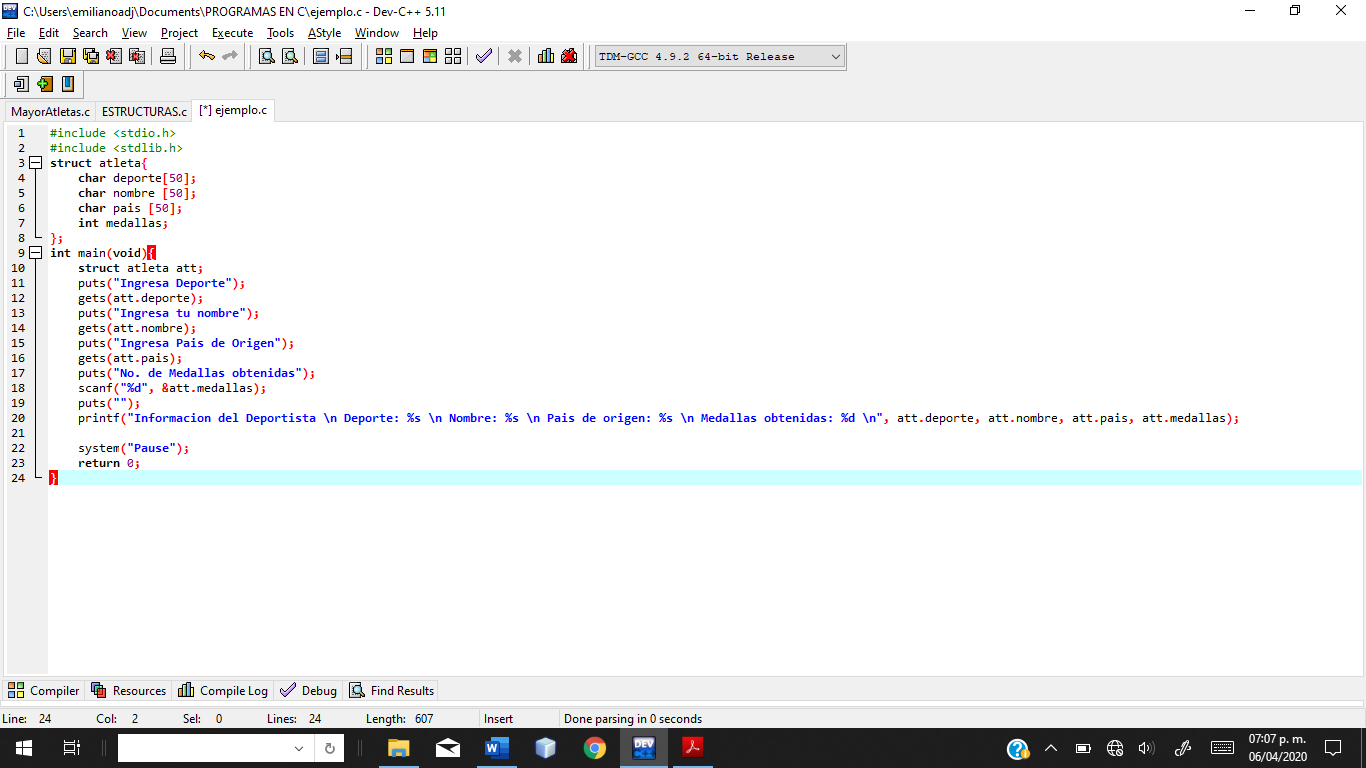


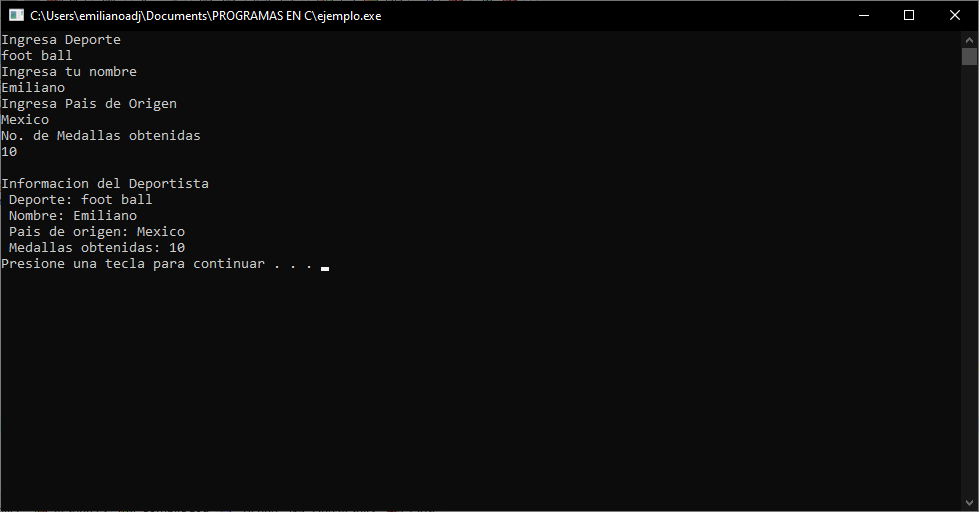
En este ejercicio, se declararon algunas variables de tipo char e int dentro de la estructura (línea 4-8).

Cuando se llama la estructura en el método main, se piden los datos correspondientes a las variables y se guardaran en las mismas.

Cuando se piden los datos, finaliza con la impresión de estos al término del código.

* **Realiza un programa en C que almacene y muestre la información de un atleta (deporte, nombre, país, n\_medallas).**

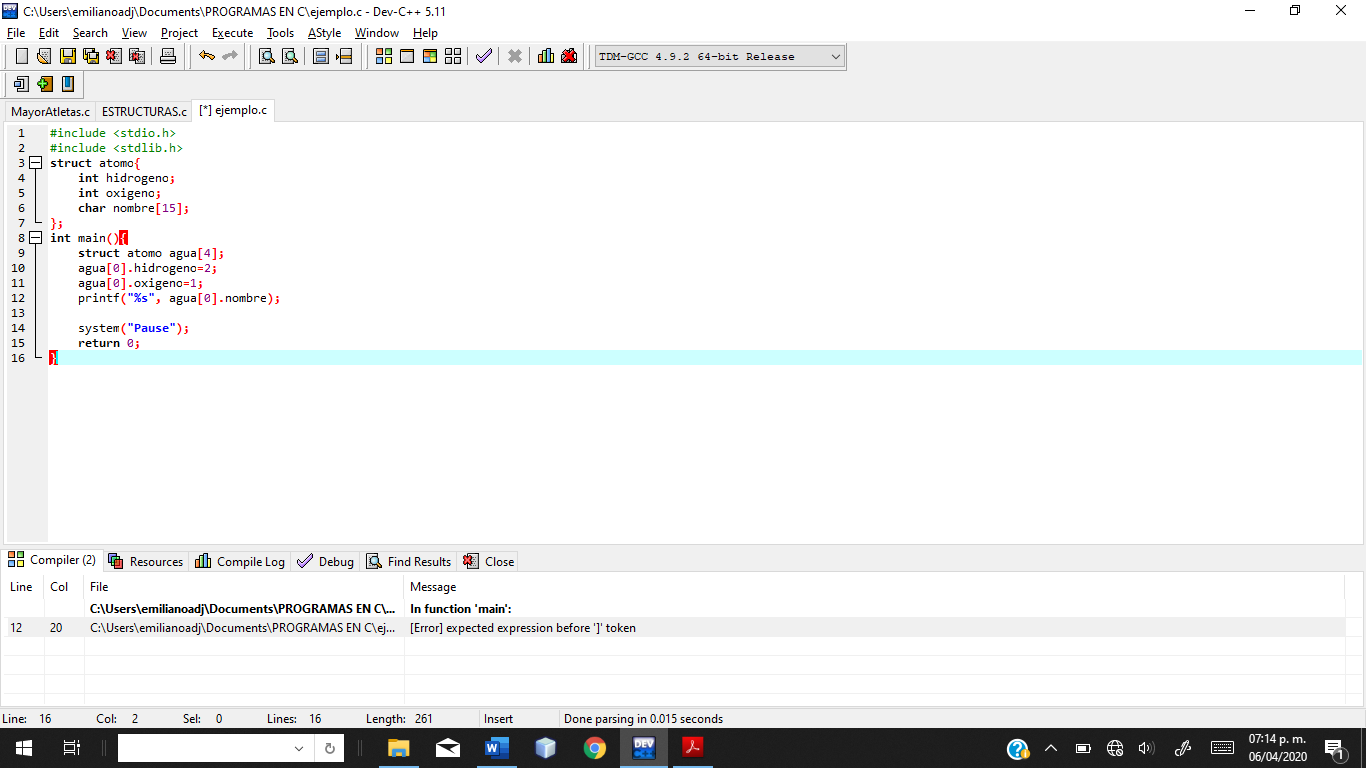




La sintaxis del programa anterior es similar a las que hemos visto, a diferencia que se piden nuevos datos y se imprimen distintas variables.

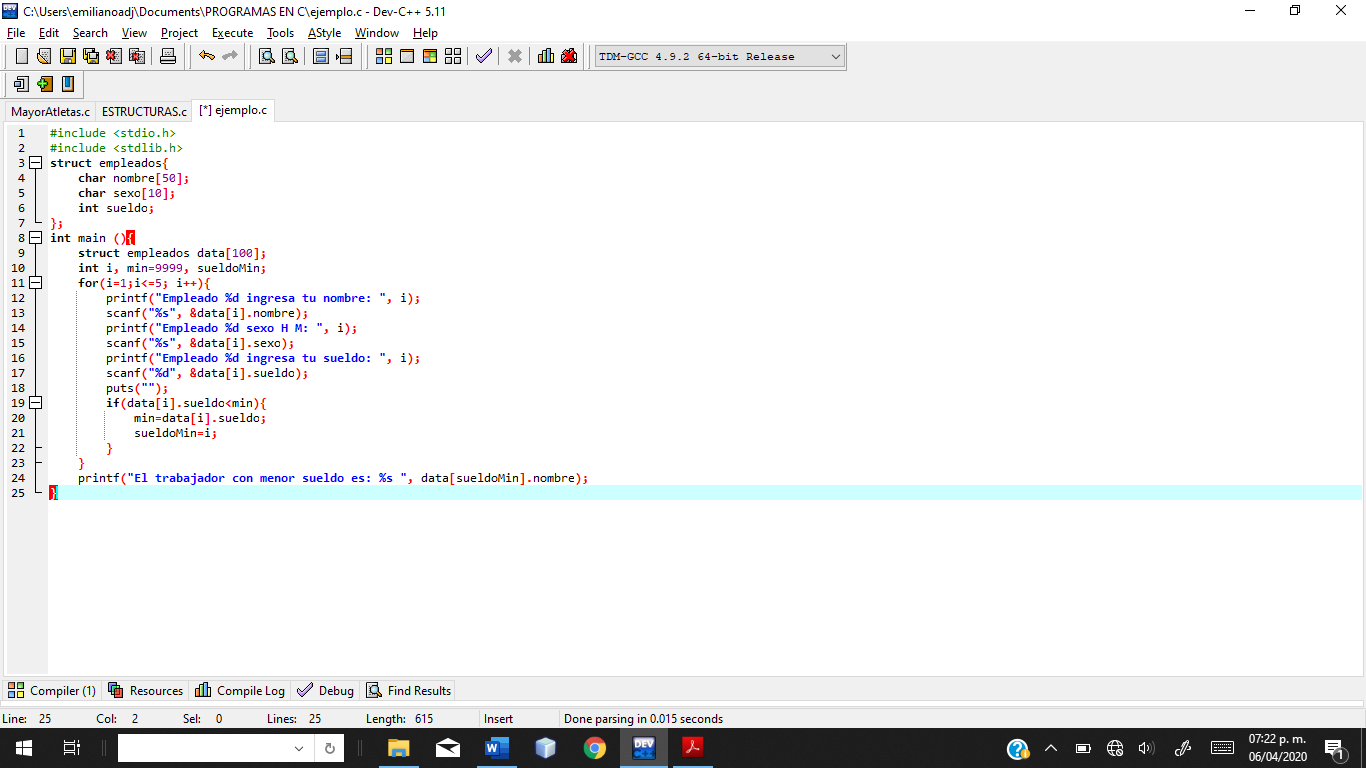
La estructura se llama deportista.

**\*\*\*ARREGLO DE ESTRUCTURAS\*\*\***

* **Ejemplo:**

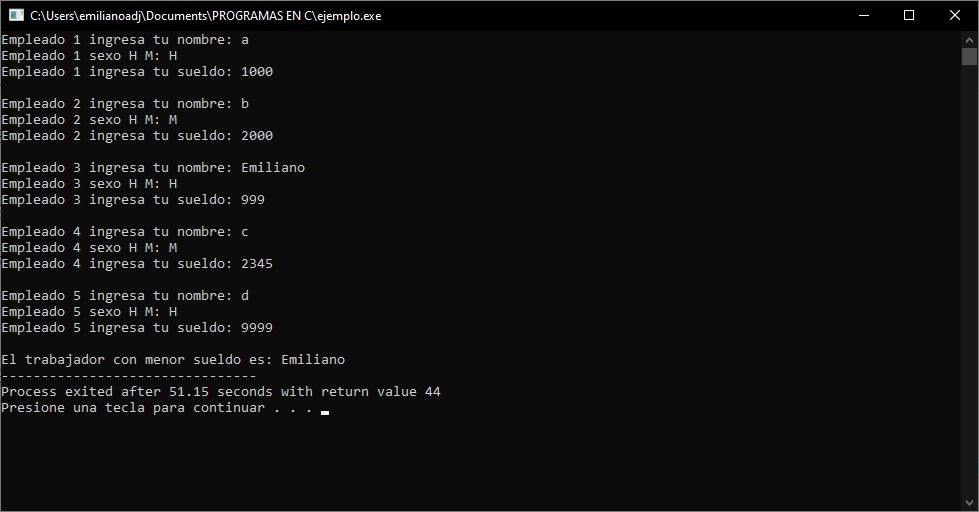
**Basicamente es un arreglo que almacena la misma estructura varias veces depenediendo de cual sea el propósito de cada una de estas.**

* **Realiza un programa en C que almacene y muestre la información de 5 empleado de la empresa DATACIC en una estructura llamada “empleado”. La información con que se cuenta del empleado es: nombre, sexo y sueldo. Muestra el empleado de menos sueldo.**



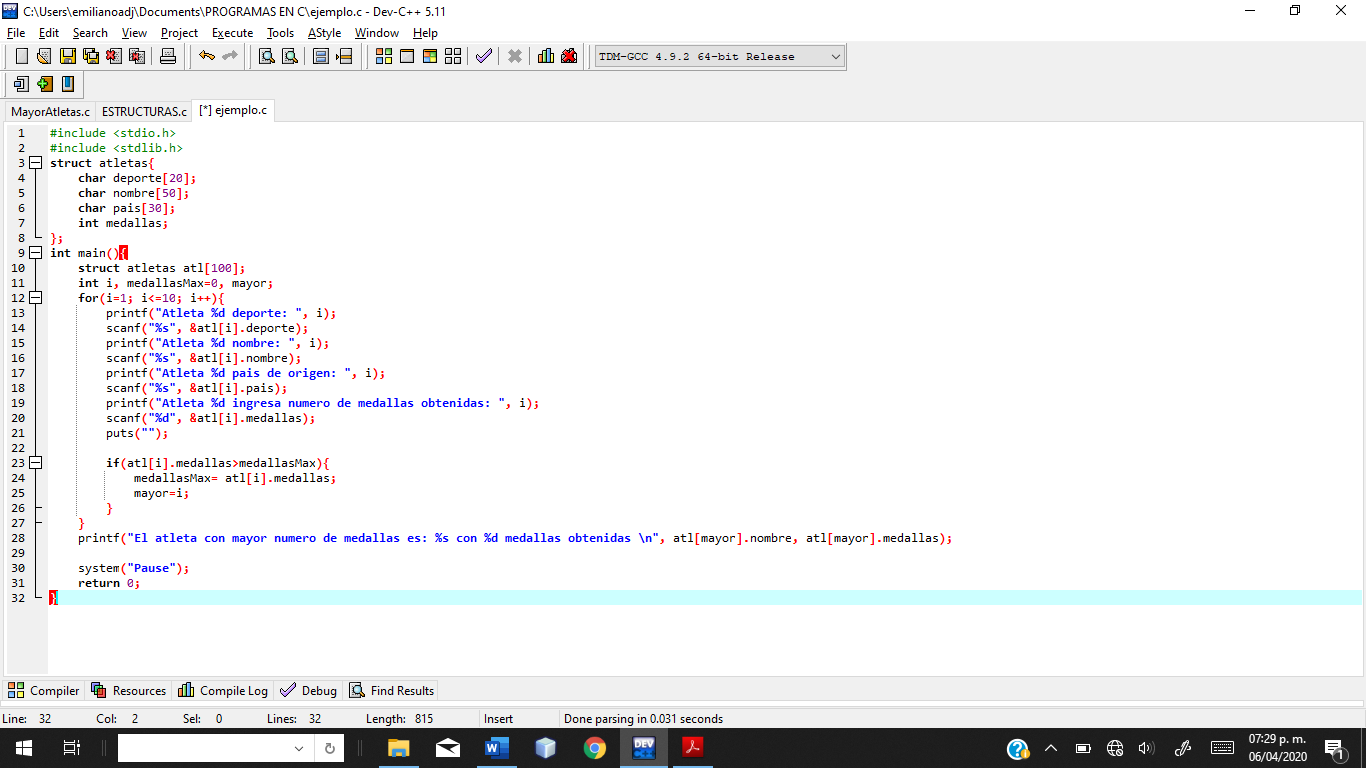
Como ya habíamos mencionado, un arreglo de estructuras almacena la misma estructura. En este caso se utilizo para almacenar los datos de 5 empleados y el programa debe deducir cual es el empleado con menor salario.

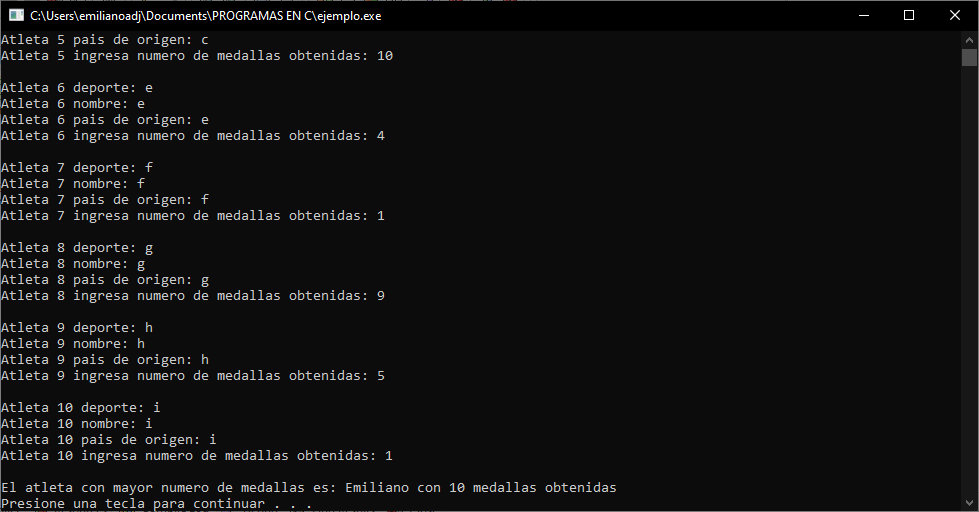
Es similar al anterior de empleado.



* **Realiza una programa en C que almacene y muestre la información de 10 atleta (deporte, nombre, país, n\_medallas) y te diga que atleta tiene mayor número de medallas.**

Este codigo es similar al anterior de atleta, a diferencia que en este se utiliza un arreglo de estructuras y el mismo programa debe identificar cual es el deportista con mayor número de medallas.





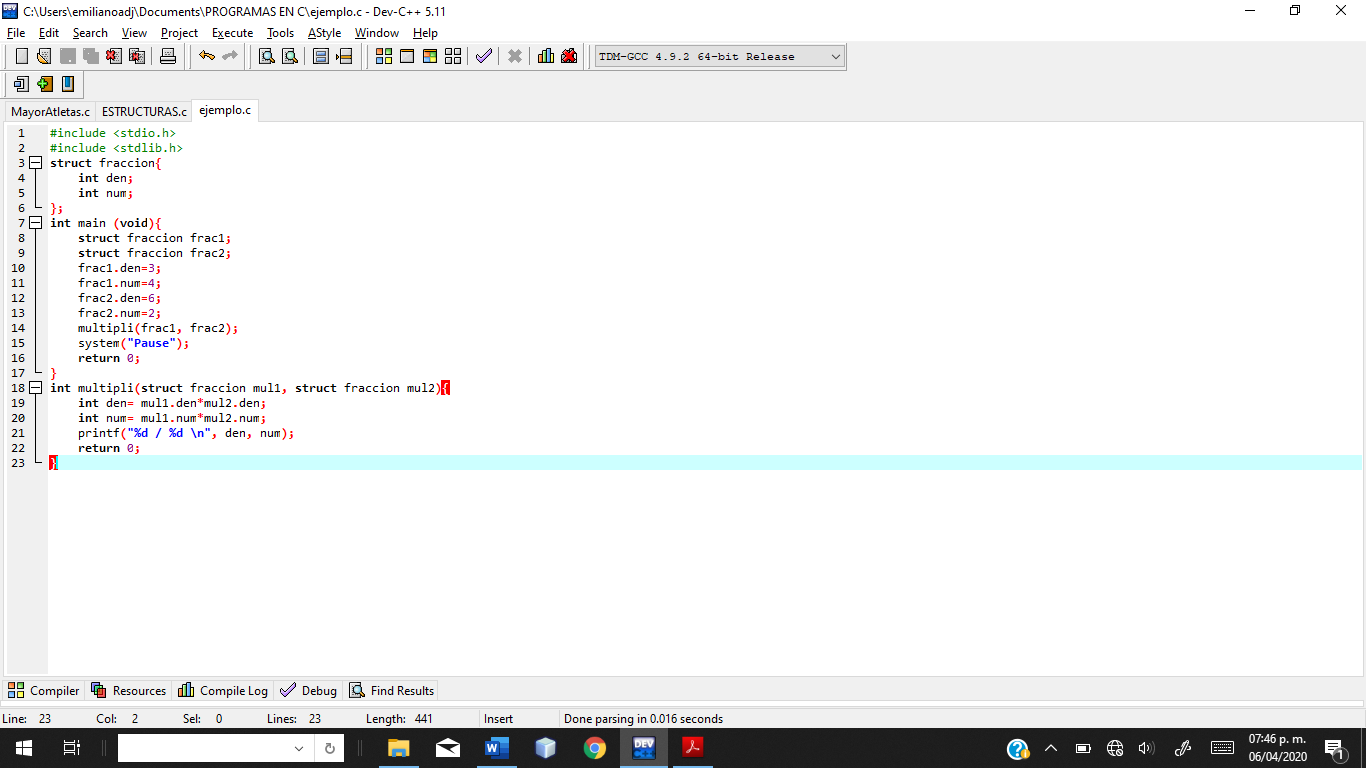
**Los datos que ingresamos fueron al azar, el propósito es mostrar al deportista con mayor número de medallas.**

**\*\*\*ENVIO DE ESTRUCTURAS A FUNCIONES\*\*\***

En los ejemplos siguientes vamos a observar cómo se le pasa como atributo de una función, una estructura para realiza las operaciones básicas de fracciones.

1. Multiplicación
2. Suma
3. Resta
4. División

* **Ejemplo:**

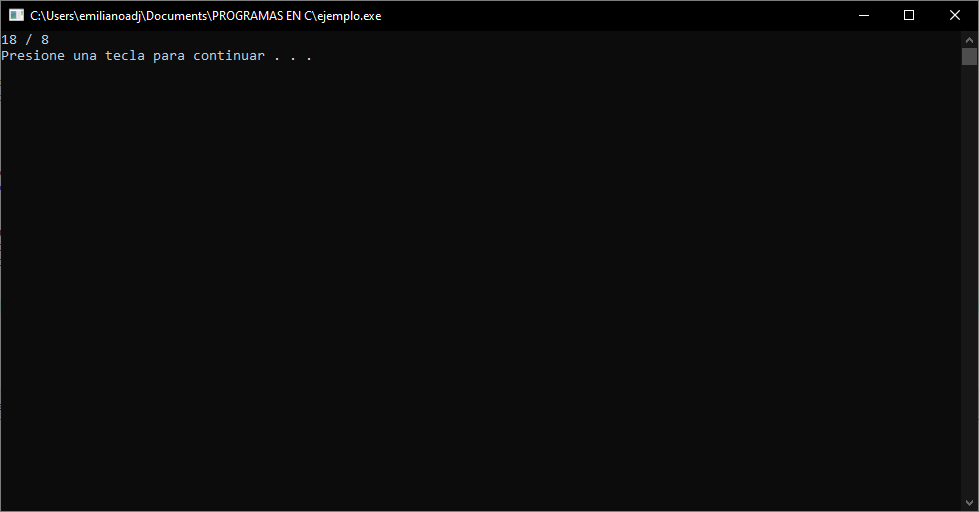


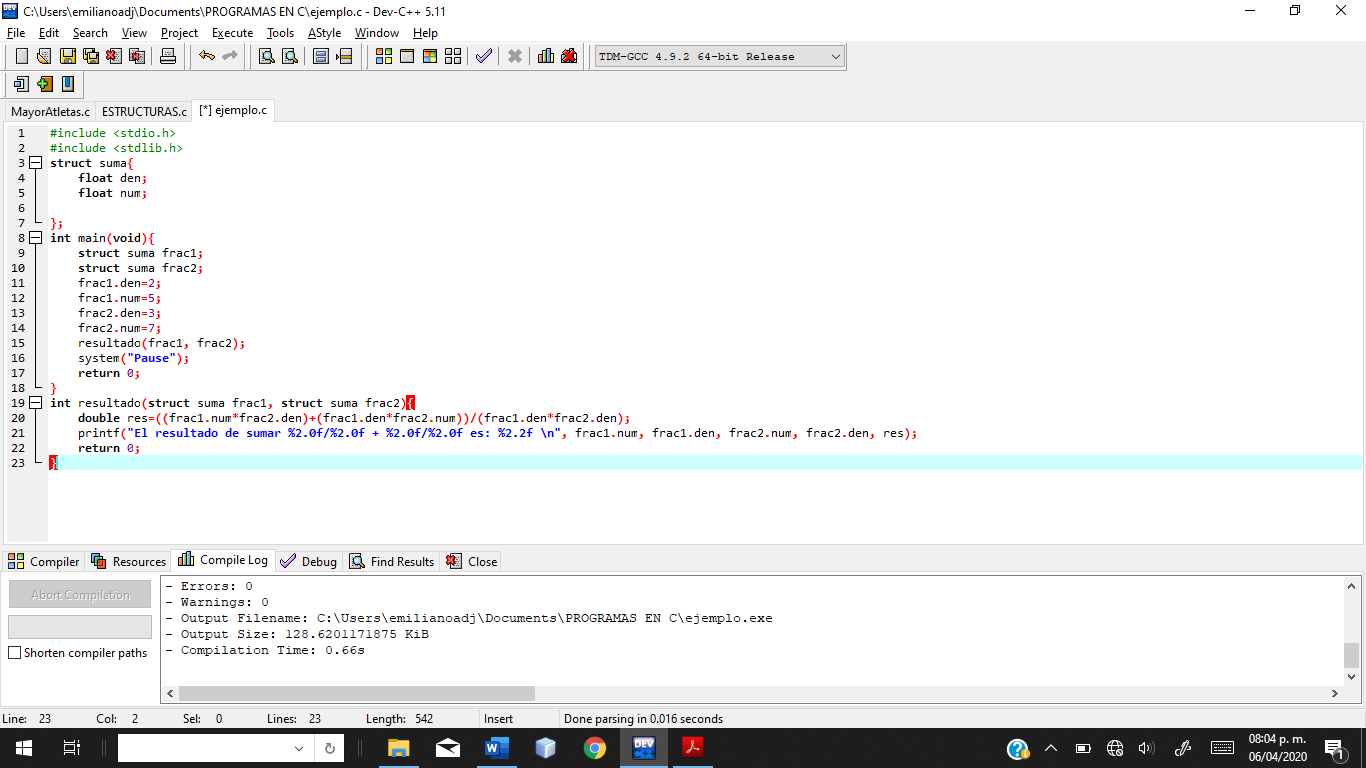
**Se realizara una multiplicación utilizando una función llamada fracción con miembros de tipo entero llamados num (numerador) den (denominador) (línea 3-6).**

**Se le pasara a la función llamada multipli como atributos (línea 18) la función que hemos mandado llamar en al método main (línea 8-8).**

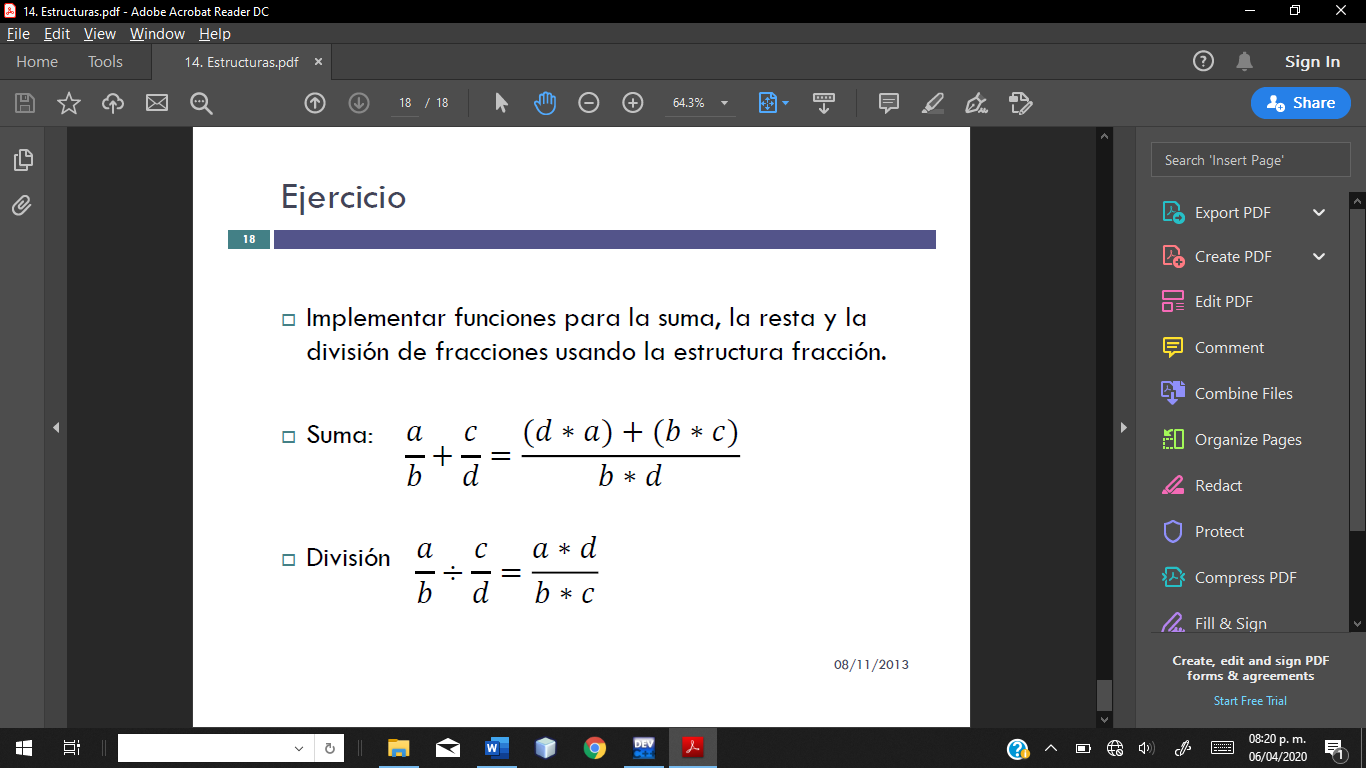
**Se hacen las operaciones dentro de la función y se imprime el resultado de dicha operación.**

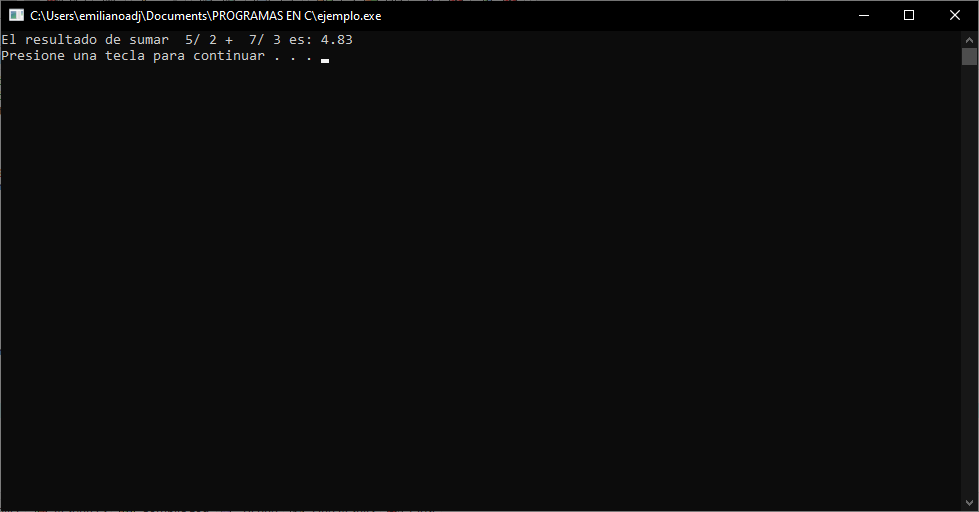
**Este sera el resultado de la operación, como estamos trabajando con fracciones, en este caso es la multiplicacion.**

****

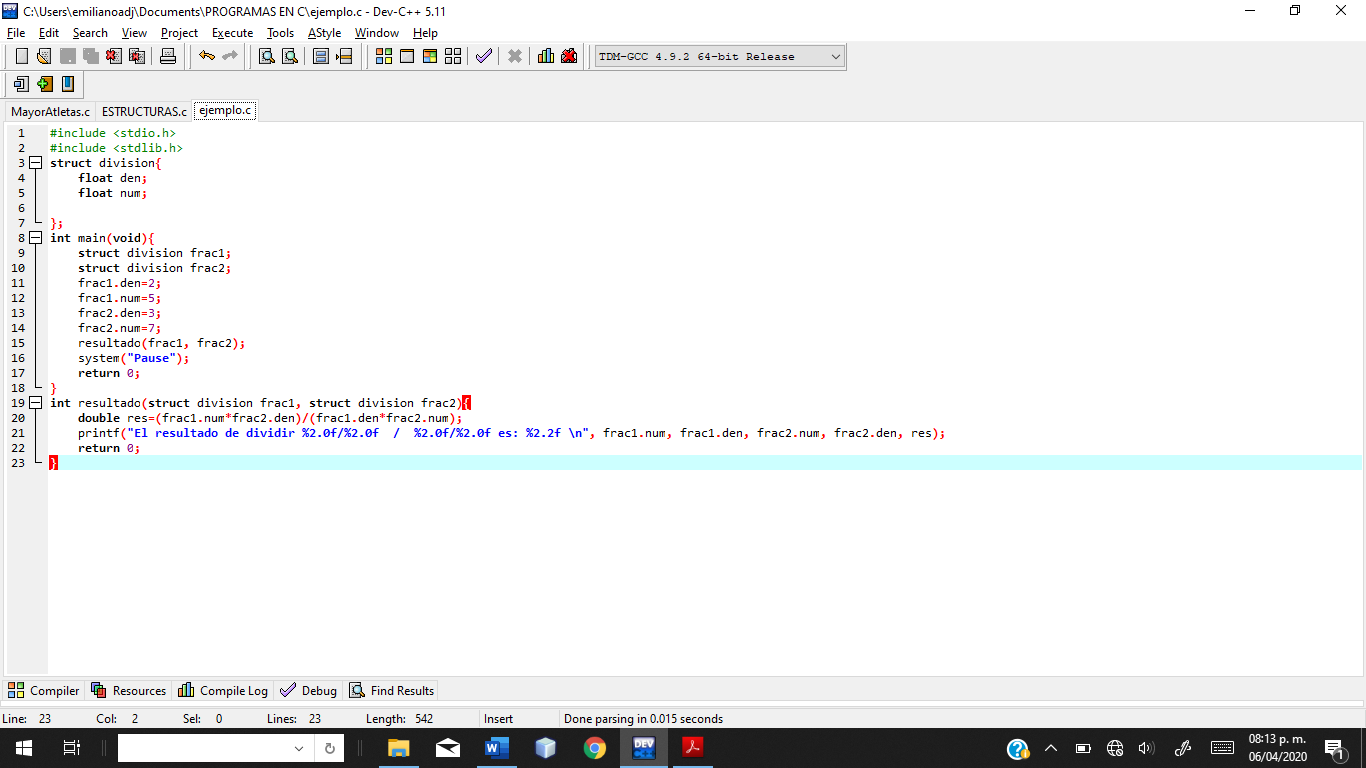
* **SUMA**

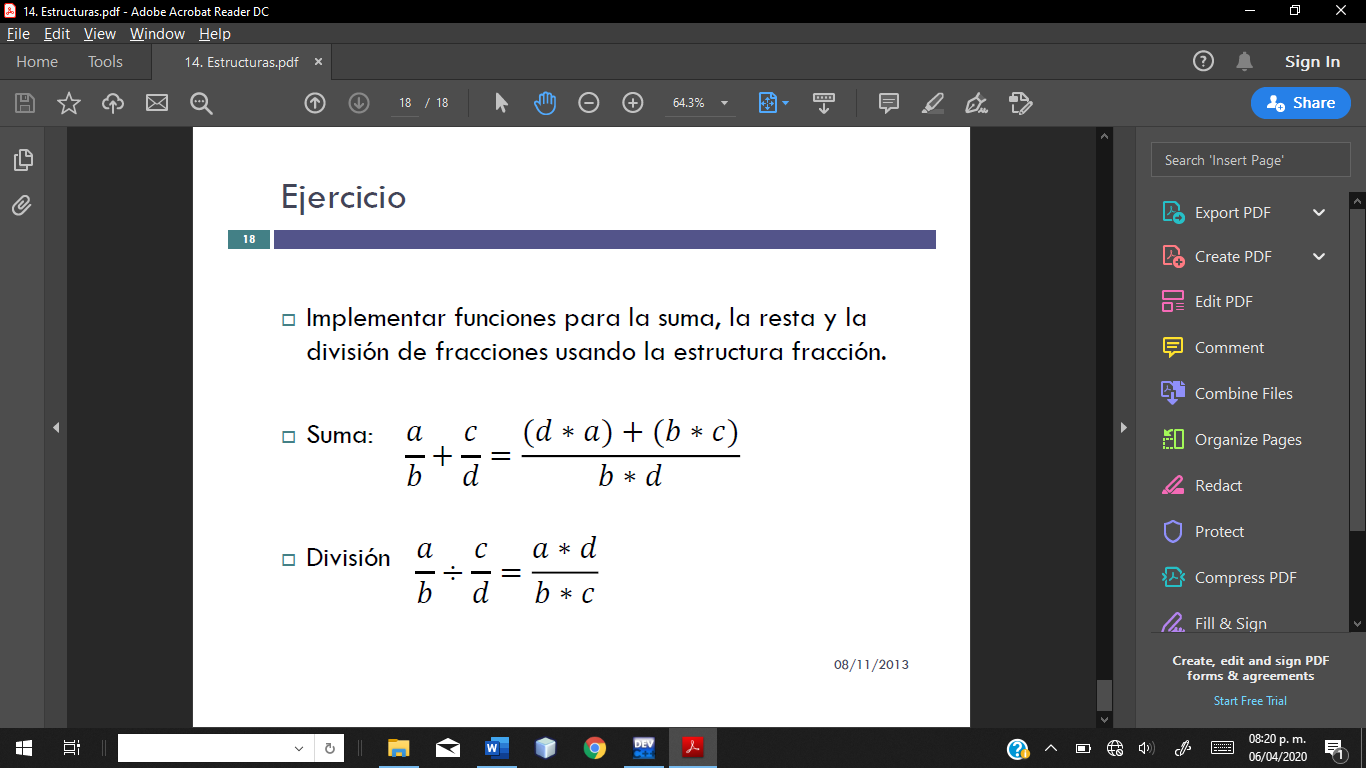
**La formula de la suma será por mera lógica diferente a la de la multiplicación, por eso se hace una operación más larga.**



**En este caso los valores para realizar las operaciones fueron ya establecidos, sin embargo se puede modificar para que el usuario los ingrese en caso de un problema diferente.**

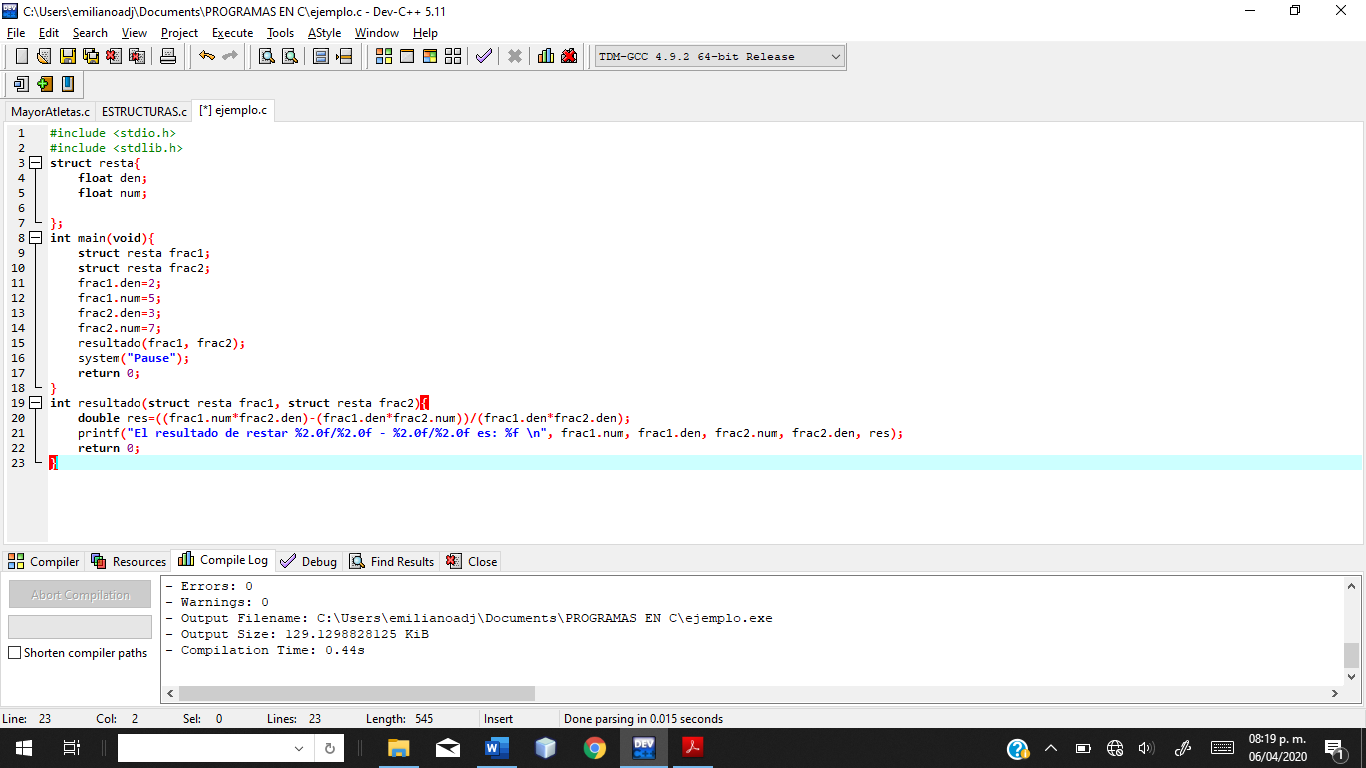
* **DIVISION**

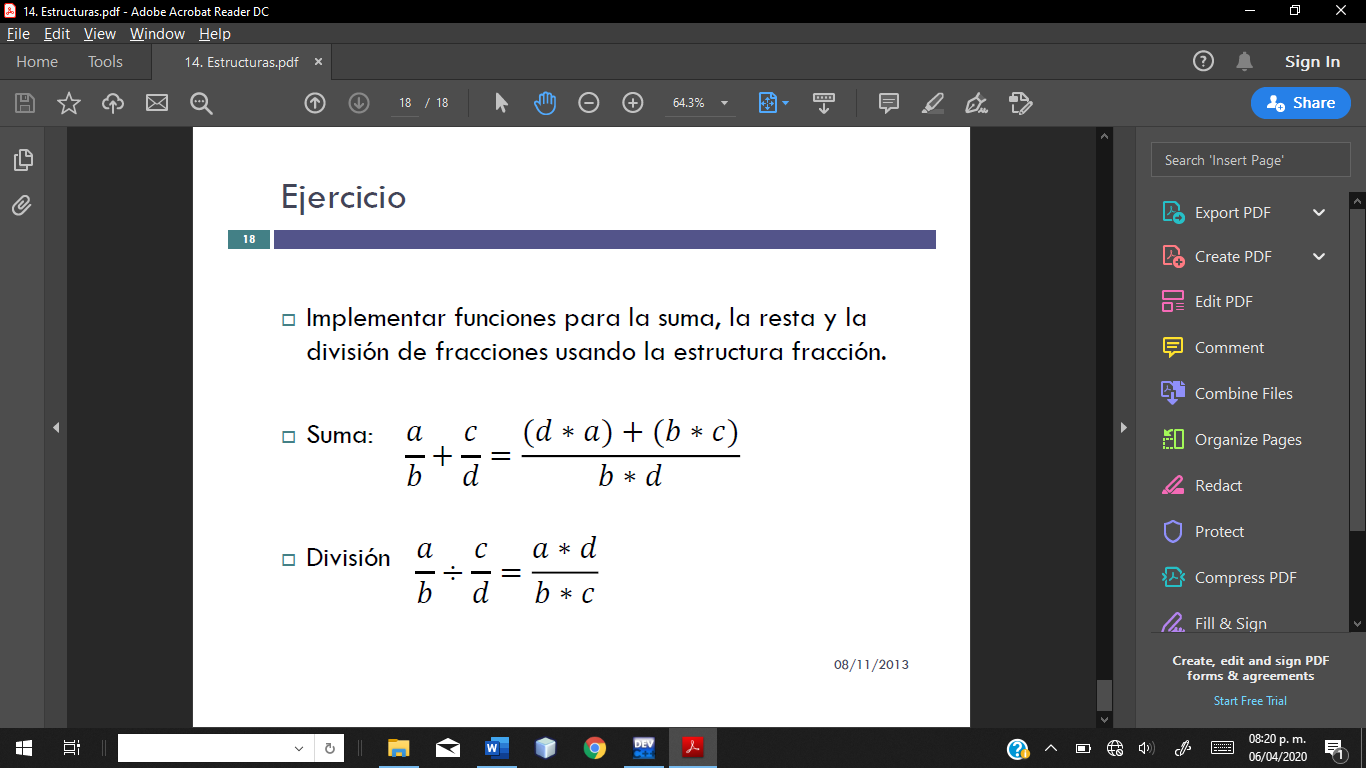


**La formula de la división es mas simple que las demás, es por eso que la operación es mas corta, solo se multiplica el numerador de la frac1 por el denominador de la frac2 que será el nuevo numerador, y el denominador de la frac1 por el numerador de la frac2 que sería el denominador y así obtener el resultado.**

**Este sería básicamente el resultado de la división.**

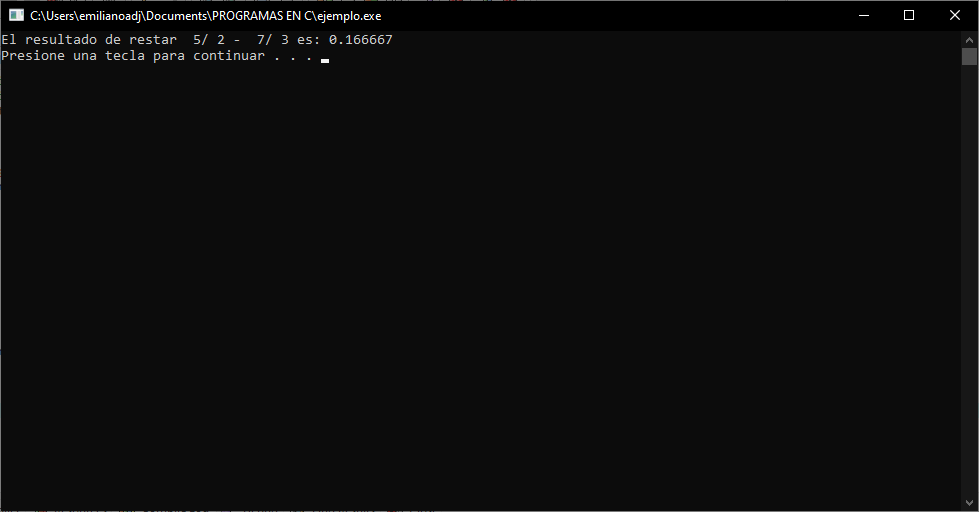


* **RESTA**

**Algebraicamente la formula de resta con fracciones es similar a la de la suma, a diferencia que se cambia el signo, en vez de + se pone -.**

-

-



**Este seria el resultado de la resta, como ya habíamos mencionado, los datos de las variables fueron establecidos desde la programación del código pero se puede modificar para que el usuario ingrese los que desee.**

1. **CONCLUSIONES:**

**Las estructuras son formas de administrar un programa, es decir, una forma de darle organización y de esta manera sea mas legible, entendible y hasta cierto punto editable.**

**Las estructuras en relación con las funciones son importantes, ya que de esta forma podemos obtener un programa más fluido y organizado de acuerdo a cada programador.**