|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NOMBRE DE LA PRÁCTICA** | **REPORTE 14 y 15 (Estructura y Apuntadores)** | | | **No.** |  |
| **ASIGNATURA:** | **MÉTODOS NÚMERICOS** | **CARRERA:** | **ISIC 3402** | **PLAN:** |  |

**ERIKA YAZMIN DIONISIO VELASCO**

**3402**

**I. COMPETENCIA(S) ESPECÍFICA(S):**

**II. MATERIAL EMPLEADO:**

* **Laptop**
* **Aplicación C++**
* **Laminas 12 y 13**

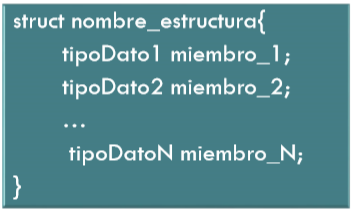
**III. DESARROLLO DE LA PRÁCTICA:**

Para comenzar con nuestra práctica tendremos que analizar antes que nada cada uno de los conceptos que les estaremos brindando, y a su vez ir aplicando cada uno de esos conceptos a los problemas propuestos. Sin más preámbulo comenzamos….

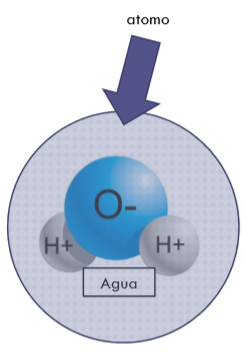
**ESTRUCTURAS (lamina 14)**

* Una estructura de datos esta compuesta de elementos individuales que pueden ser de distinto tipo.
* Cada uno de los elementos de una estructura se denomina miembro.

**DECLARACIÓN DE UNA ESTRUCTURA**



Los miembros pueden ser cualquier tipo excepto void



**EJEMPLO**

**struct atomo{**

int hidrogeno;

int oxigeno;

char nombre[15];

}

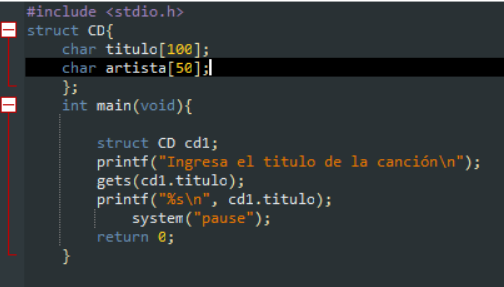
**PROGRAMA 1**

Los miembros pueden ser cualquier tipo excepto void

Declaración de la **estructura**

Creación de dos arreglos **titulo y artista** o también conocidos como miembros, asignándole a cada uno un número de posiciones.

Clase principal



**printf,** imprime el mensaje en pantalla.

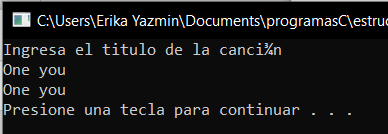
**gets,** imprime la cadena, seguido del nombre base para guardar lo asignado desde teclado.

Mandamos a llamar a la estructura **struct CD,**junto con un nombre base (**cd1**)para mandar a llamar a cada uno de los elementos creados en la estructura.

Fin del programa

Imprime lo que el usuario escribió en pantalla, asignando el especificador de conversión **%s (cadena de carácter)** y seguido de esto el nombre de la base y el miembro donde se guardó la cadena.

**EJECUCIÓN DEL PROGRAMA**



**PROGRAMA 2**

Modifica la estructura para que se pueda almacenar también.

* num\_canciones
* anio
* precio

**struct CD{**

**char titulo[100]; char artista[50]**

**};**

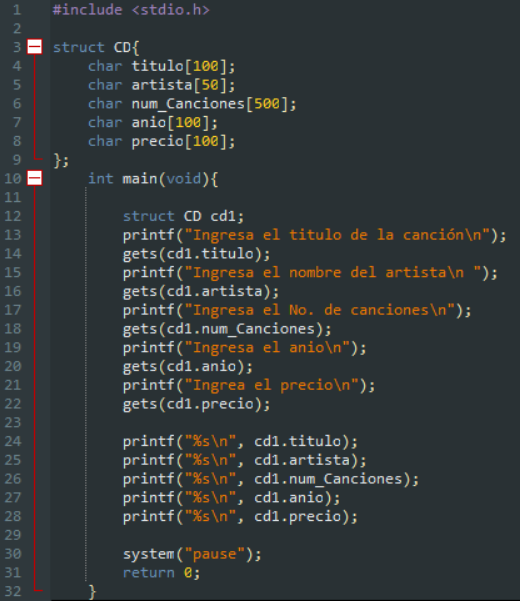
**Nota:** El usuario debe ingresar todos los datos por el teclado

Declaración de la **estructura**

Creación de dos arreglos **titulo y artista,num\_canciones,anio y precio** o también conocidos como miembros, asignándole a cada uno un número de posiciones

Método principal

Mandamos a llamar a la estructura **struct CD,**junto con un nombre base (**cd1**)para mandar a llamar a cada uno de los elementos creados en la estructura.



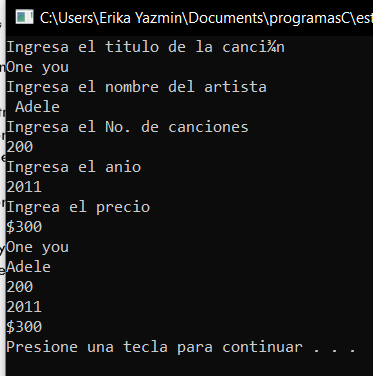
Imprime lo que el usuario escribió en pantalla, asignando el especificador de conversión **%s (cadena de carácter)** y seguido de esto el nombre de la base y el miembro donde se guardó la cadena

Fin del programa

**printf,** imprime el mensaje en pantalla.

**gets,** imprime la cadena, seguido del nombre base para guardar lo asignado desde teclado, al igual que el arreglo donde este se encuentra.

**EJECUCIÓN DEL PROGRAMA**



**PROGRAMA 3**

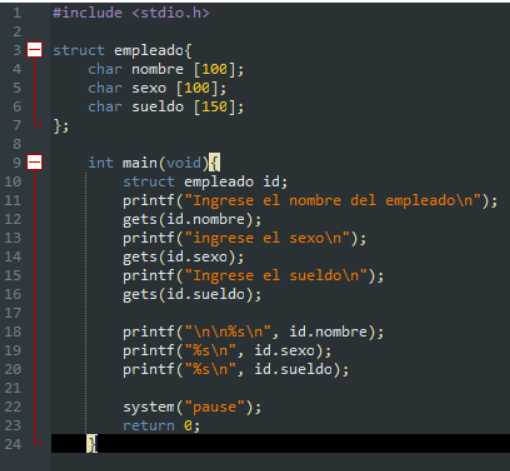
Realiza un programa en C que almacene y muestre la información de un empleado de la empresa DATACIC en una estructura llamada “empleado”.

La información con que se cuenta del empleado es: nombre, sexo y sueldo.

Crear la estructura **empleado**, seguido de sus miembros llamados **nombre, sexo y sueldo,** estos son arreglos de tipo **char**, asignándoles un número de posiciones.

Los miembros pueden ser cualquier tipo excepto void

Método principal



Imprime lo que el usuario escribió en pantalla, asignando el especificador de conversión **%s (cadena de carácter)** y seguido de esto el nombre de la base y el miembro donde se guardó la cadena

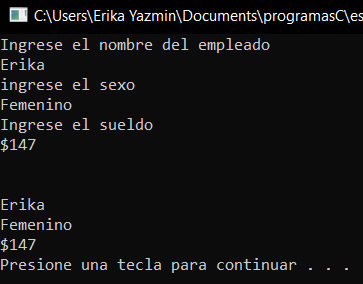
**printf,** imprime el mensaje en pantalla.

**gets,** imprime la cadena, seguido del nombre base para guardar lo asignado desde teclado, al igual que el arreglo donde este se encuentra.

Mandamos a llamar a la estructura **struct empleado,** junto con un nombre base (**id**)para mandar a llamar a cada uno de los elementos creados en la estructura.

Fin del programa

**EJECUCIÓN DEL PROGRAMA**



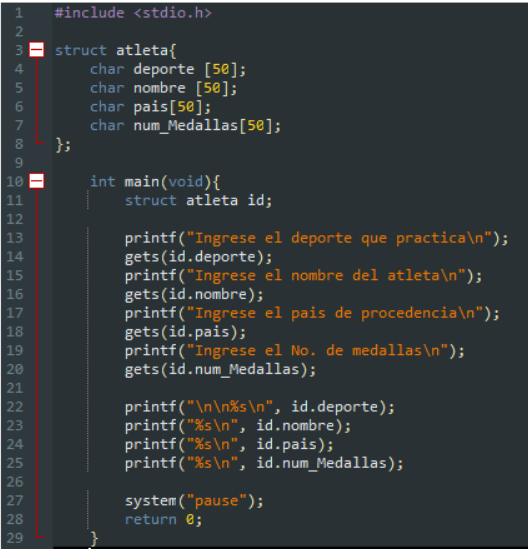
**PROGRAMA 4**

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

Crear la estructura **atleta**, seguido de sus miembros llamados **deporte, nombre, país y Num\_Medallas,** estos son arreglos de tipo **char**, asignándoles un número de posiciones.

Los miembros pueden ser cualquier tipo excepto void

Método principal



Imprime lo que el usuario escribió en pantalla, asignando el especificador de conversión **%s (cadena de carácter)** y seguido de esto el nombre de la base y el miembro donde se guardó la cadena

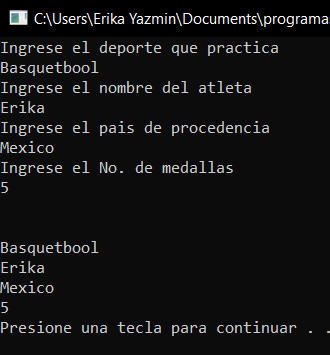
**printf,** imprime el mensaje en pantalla.

**gets,** imprime la cadena, seguido del nombre base para guardar lo asignado desde teclado, al igual que el arreglo donde este se encuentra.

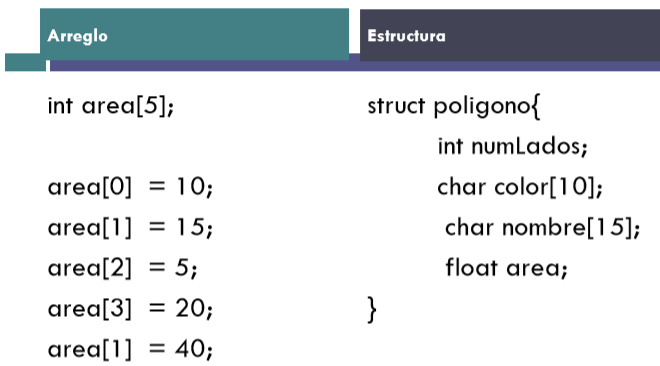
Mandamos a llamar a la estructura **struct atleta,** junto con un nombre base (**id**) para mandar a llamar a cada uno de los elementos creados en la estructura.

Fin del programa

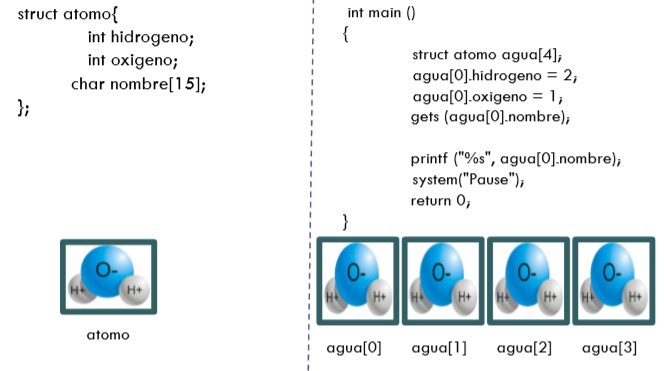
EJECUCIÓN DEL PROGRAMA



COMPARACIÓN DE ARREGLO Y ESTRUCTURA



**ARREGLO DE ESTRUCTURAS**



**PROGRAMA 5**

Realiza un programa en C que almacene y muestre la información de 5 empleado de la empresa DATACIC en una estructura llamada “empleado”.

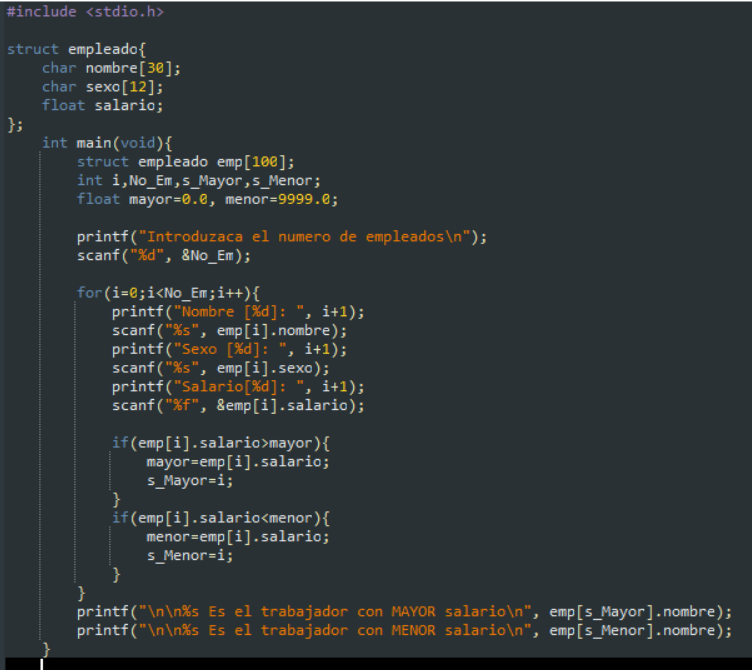
La información con que se cuenta del empleado es: nombre, sexo y sueldo. Muestra el empleado de menos sueldo.

Creación de la estructura **empleado**, junto con sus miembros llamados **nombre, sexo** de tipo **char** y asignándole un tamaño y por último **salario** de tipo **float.**

Mandamos a llamar a la estructura, seguido de la creación de un arreglo con un tamaño asignado

Los miembros pueden ser cualquier tipo excepto void

Declaración de 4 variables en donde guardaremos el resultado del No. de empleados, inicialización de nuestro for, y así mismo asignaremos el sueldo mayor y menor del trabajador



Ciclo **for**, aquí diremos si nuestra variable inicializada en 0 es menor al No. de empleados, entonces imprimirá:

**printf ,** el mensaje en pantalla con el No. de empleado asignado

**scanf ,** recopila el valor obtenido en printf, para que este asigne los valores de nombre, sexo y salario acorde al No. de trabajador que vayamos a asignar.

Aunque en salario tenemos una excepción ya que esta en vez de recopilar una cadena, con ayuda de **&**, recopila el valor numérico.

**printf,** Manda el mensaje en pantalla

**scanf**, Recopila el valor obtenido desde teclado.

Declarar 2 variables de tipo **float** en donde a la variable menor y mayor le asignaremos un valor para después compararlo con el resultado del sueldo.

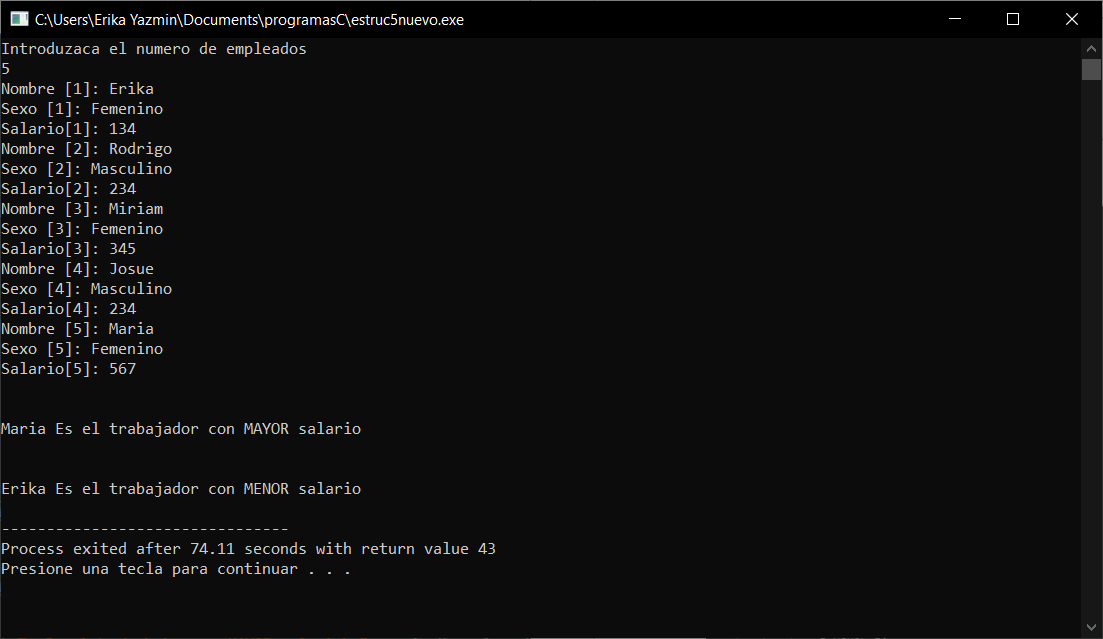
Ahora solo imprimirá el nombre del trabajador con ayuda del especificador de conversión **%s (cadena de carácter),** el mensaje y por último la asignación de nuestro arreglo en donde guarda el salario menor o mayor, seguido de la variable salario

Condiciona, si el valor de salario, asignado en cada trabajador es menor a 0, entonces la posición en donde se encontró se guardara dentro de la variable **s\_Mayor.**

Si de lo contrario el salario en la posición encontrada es menor que 999.0, entonces esta se guardará en **s\_Menor.**

7777

EJECUCIÓN DEL PROGRAMA



**PROGRAMA 6**

Realiza una programa en C que almacene y muestre la información de 10 atleta (deporte, nombre, pais, nmedallas) y te diga que atleta tiene mayor número de medallas.

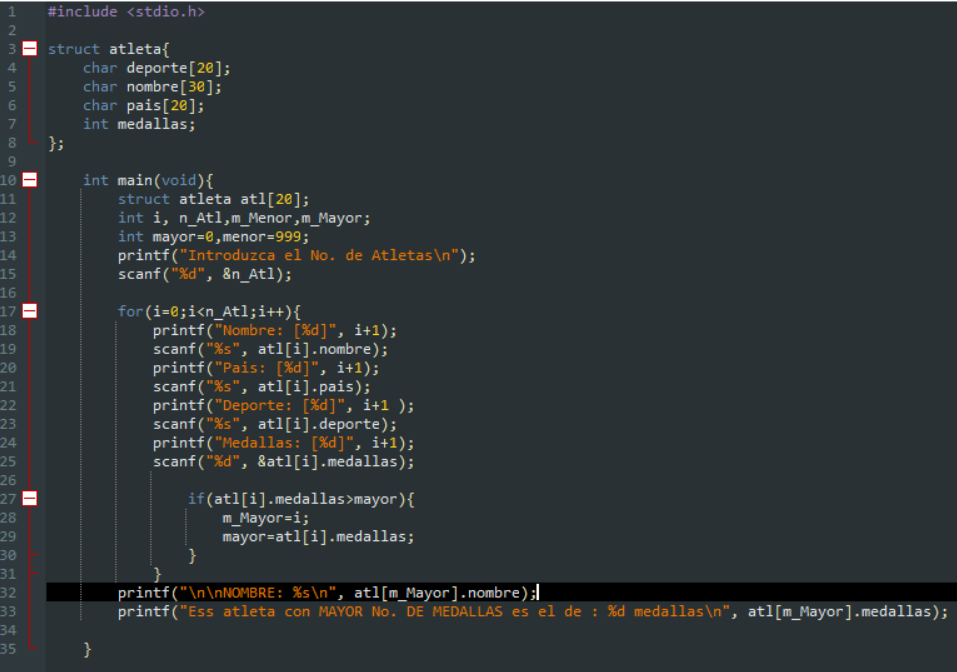
Manda a llamar a la estructura, seguido de un arreglo con 20posiciones.

Declaración de **i(for),No. de atletas y** variables donde almacenara el atleta con mayor medallas.

Declaración de la variable **mayor**=0, para poder hacer la comparación de las medallas de cada atleta.

Crear la estructura llamada **atleta**, seguido de sus miembros llamados **deporte, país y nombre**, y asignándole a cada uno posiciones y por último una variable de tipo entero llamada medallas

Los miembros pueden ser cualquier tipo excepto void



Condiciona, si el valor de medallas, asignado en cada atleta es mayor a 0, entonces la posición en donde se encontró se guardara dentro de la variable **s\_Mayor.**

Ciclo **for**, aquí diremos si nuestra variable inicializada en 0 es menor al No. de atletas, entonces imprimirá:

**printf ,** el mensaje en pantalla con el No. de atletas asignado

**scanf ,** recopila el valor obtenido en printf, para que este asigne los valores de nombre, país, deporte y medallas acorde al No. de atletas que vayamos a asignar.

Aunque en salario tenemos una excepción ya que esta en vez de recopilar una cadena, con ayuda de **&**, recopila el valor numérico.

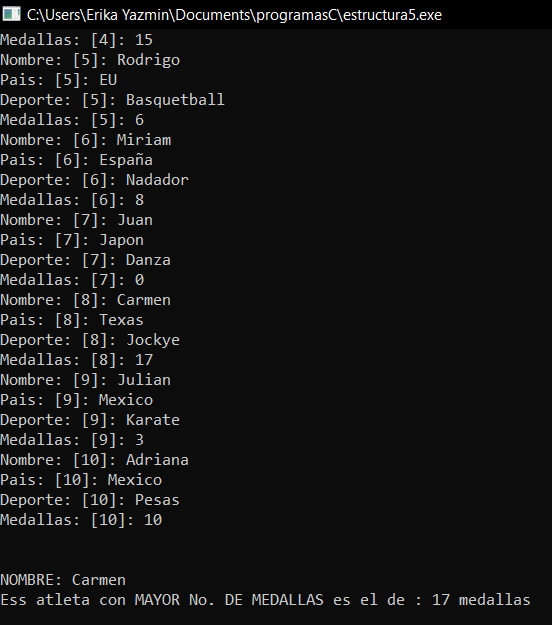
**printf,** Manda el mensaje en pantalla

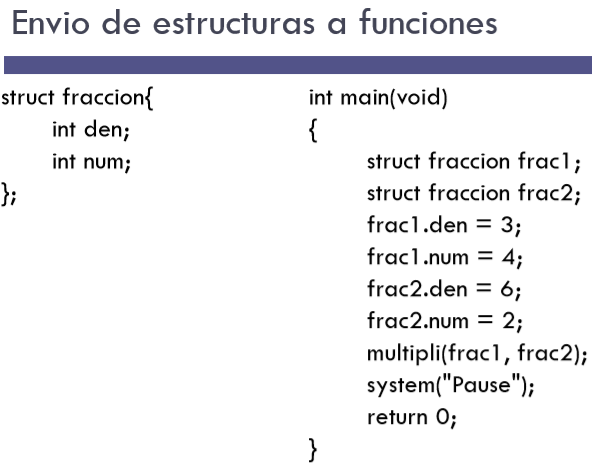
**scanf**, Recopila el valor obtenido desde teclado.

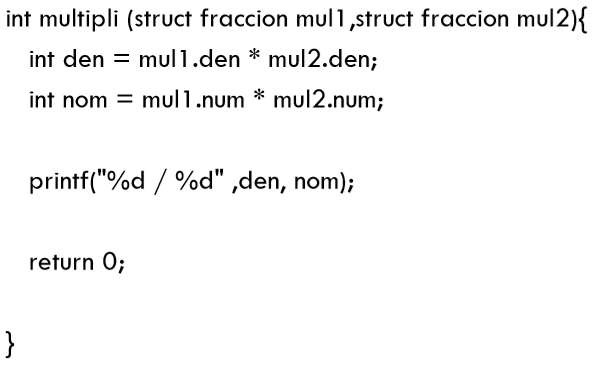
Al final, imprime el nombre del atleta con ayuda del especificador de conversión **%s (cadena de carácter),** seguido del arreglo donde se guarda el nombre.

Y ahora para imprimir las medallas con mayor numero solo colocaremos el mensaje y con ayuda del especificador de conversión **%d (valor entero),** sabremos que llamara al mayor número que encuentre, gracias a la posición en la que el arreglo se encuentre.

COMPILACIÓN DEL PROGRAMA

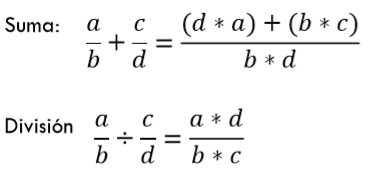






**PROGRAMA 7**

Implementar funciones para la suma, la resta y la división de fracciones usando la estructura fracción.

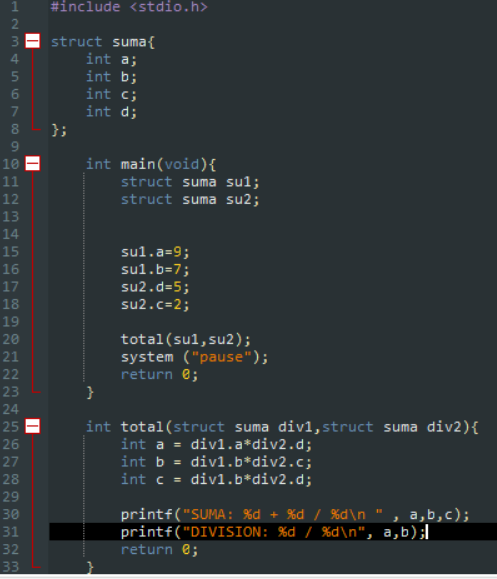


Los miembros pueden ser cualquier tipo excepto void

Creación de la estructura llamada **suma,** seguido de 4 elementos de tipo entero

Llamar a la estructura dos veces, seguido de un identificador

Solicitar el identificador 2 veces a cada uno en donde se guardarán los valores asignados las 4 variables utilizadas en la suma y división presentada a continuación



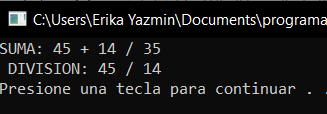
Ahora, declarar 3 nuevas variables, en donde realizaremos las operaciones correspondientes a la suma, en este caso será (d\*a)+(b\*c)/(b\*d) , recordando que en la primera variable será lo mismo que coloquemos nuestros identificadores, seguido de la variable con el valor asignado y por ende esta le asignara a cada variable el valor correspondiente para hacer la suma y de la misma forma la división

Crear un método en donde llame a la estructura, seguido de dos nuevos identificadores, pero estos toman el lugar de los primeros.

Ahora imprimirá 3 valores con ayuda de los especificadores de conversión **%d**, y las tres variables en donde se alojan los resultados

Retornará en el nombre de la estructura y comparara cada uno de os elementos donde se alojó el resultado para así solo imprimir

**EJECUCIÓN DEL PROGRAMA**



**IV. CONCLUSIONES:**

Para concluir con nuestro manual de ESTRUCURAS tengo que decir que está demasiado relacionado con algunos conceptos vistos anteriormente, lo que lo hace demasiado especial es que cada problema se ejecuta de una manera distinta pero sin en cambio todos llegan a un mismo fin, como lo vimos anteriormente cada estructura está explicada de una manera distinta, no es lo mismo la declaración de una estructura con una estructura relacionada con arreglo, sin en cambio podemos darnos cuenta que algunos conceptos, por más difíciles que sean, estos simplemente tienen una manera más sencilla de resolver los problemas, así como acortar cada día más y más las líneas de código.

Para finalizar puedo decir que me gustó mucho este tipo de prácticas ya que además de ponerte a pensar te ayuda a reforzar tus conocimientos adquiridos.