|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre de la práctica** | **Lenguaje C** | | | **No.** | **3** |
| **Asignatura:** | **Métodos numéricos** | **Carrera:** | **INGENIERIA EN SISTEMAS COMPUTACIONALES** | **Duración de la práctica (Hrs)** |  |

1. **Competencia(s) específica(s):** \* \* \* \* \* \* \* \* \*

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

* Aula

**III. Material empleado:**

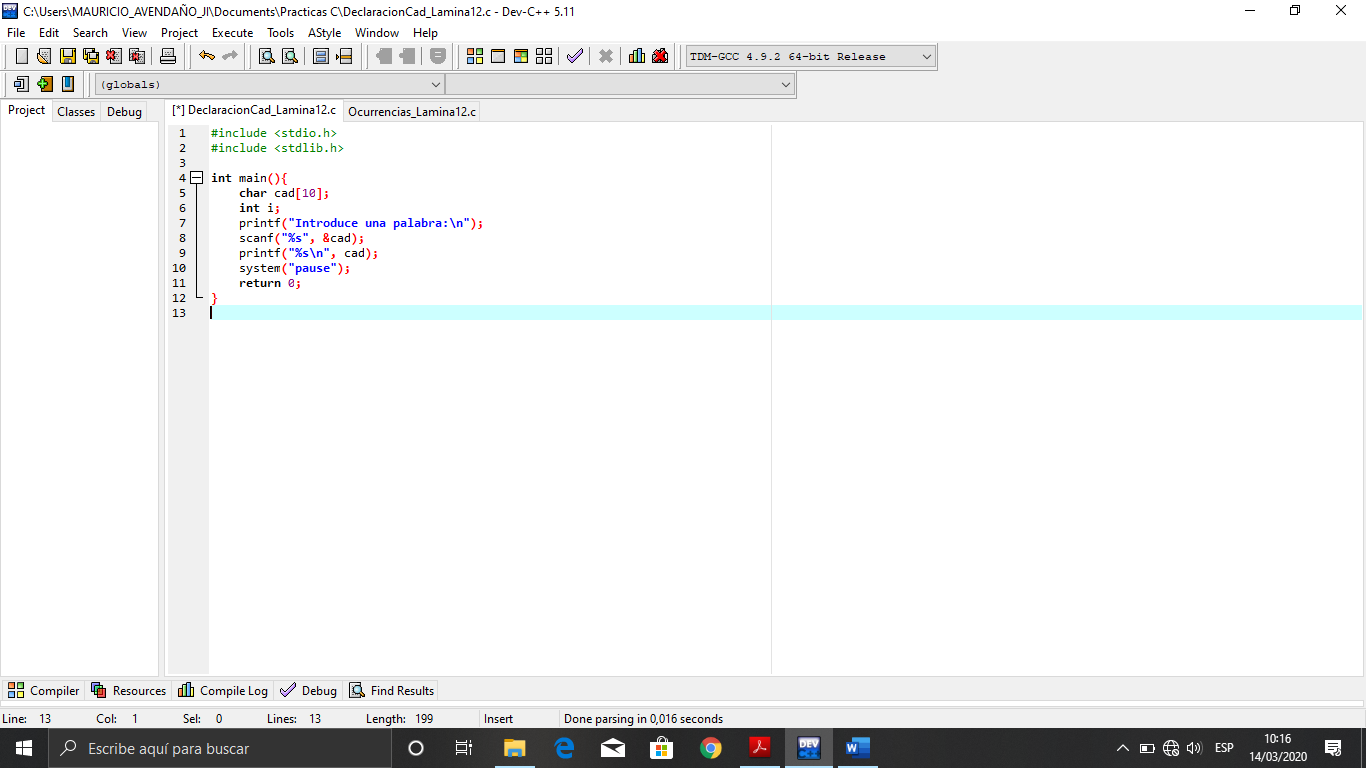
* Dev-C++
* Editor de texto
* Documentos

**IV. Desarrollo de la práctica:**

**Documento 12**

Ejemplo. Cadenas. Declaración de una cadena con tamaño

%s Especificador de conversión de caracteres



En primer lugar mostramos la instrucción al usuario con ayuda de printf(), después con scanf() guardamos el valor que el usuario nos de por medio de teclado, este valor se asigna al especificador de acceso para posteriormente guardar dicho valor en la variable correspondiente, en este caso, cad. Al final nuevamente con printf() mostramos lo que el usuario nos ingresó, e igual se asigna al especificador de conversión.

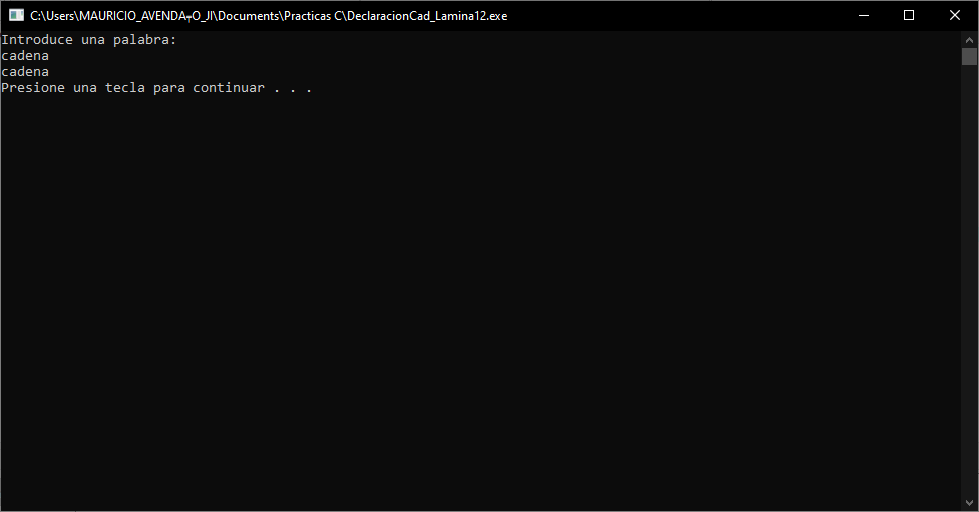
Declaración de una cadena, consta de su tipo de dato (char), el nombre (cad) y su tamaño (10), al igual que una variable llamada i de tipo entero, designada para el ciclo.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

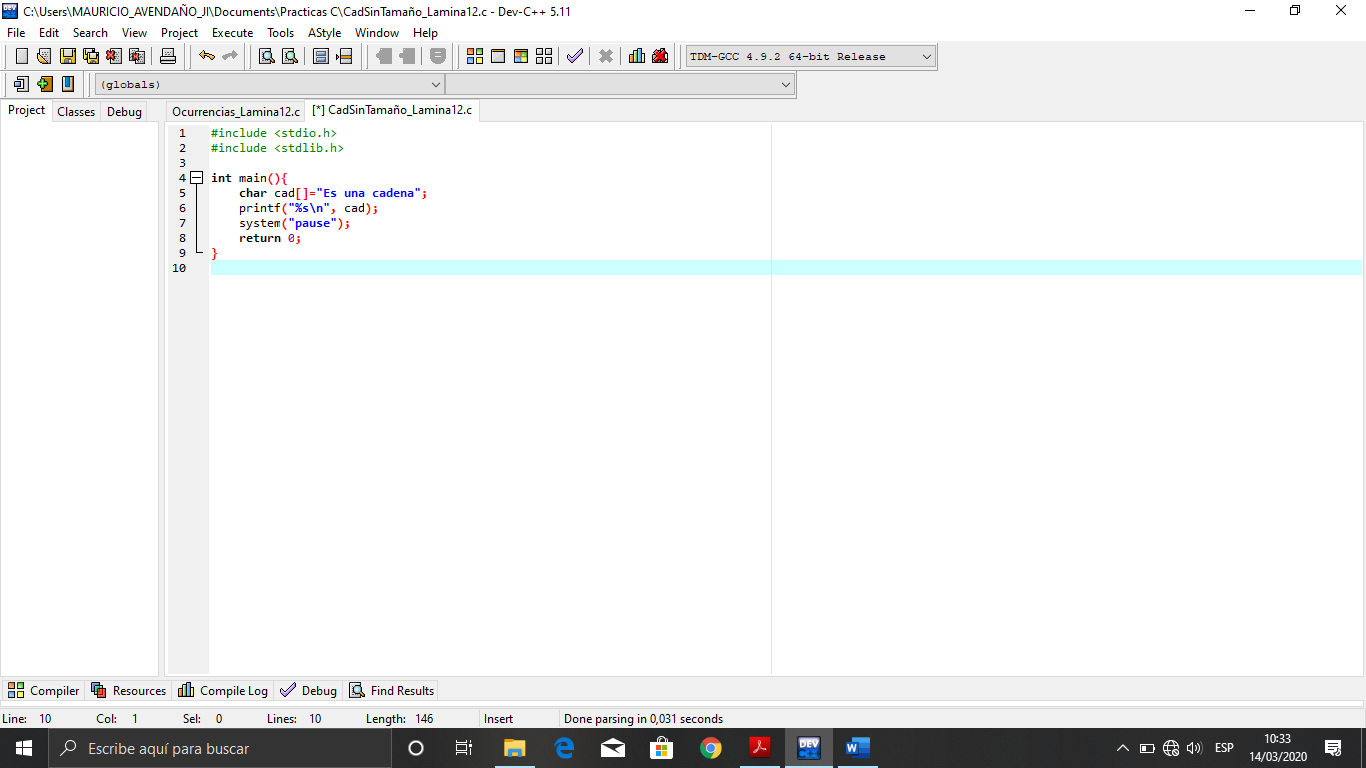
Valor que devuelve el método principal

Método principal

Bibliotecas



Ejemplo. Cadenas. Declaración de una cadena sin tamaño



Con printf() mostramos el valor de la cadena en pantalla, el valor se le asigna al especificador de conversión qu espera un tipo de dato char.

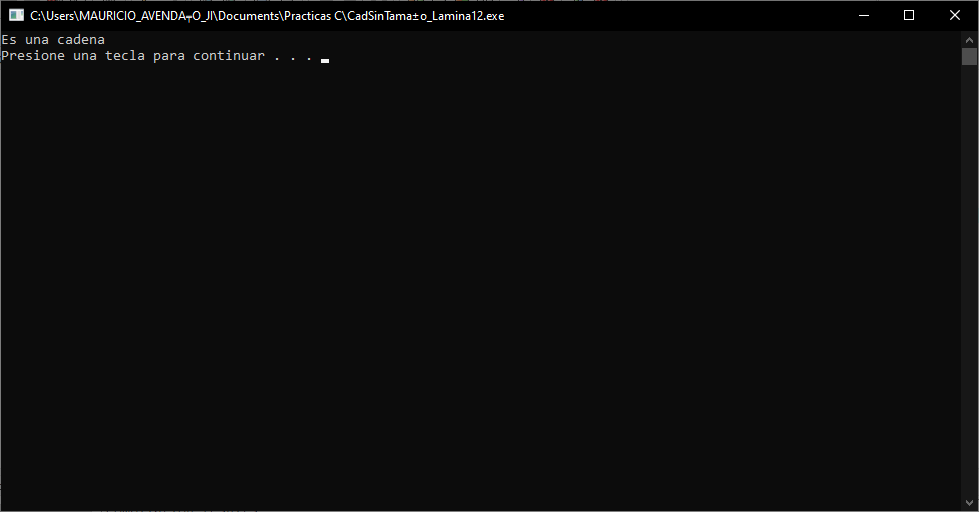
Declaración de una cadena sin tamañ0, consta del tipo de dato (char), su nombre (cad) y su contenido, el texto entre comillas.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

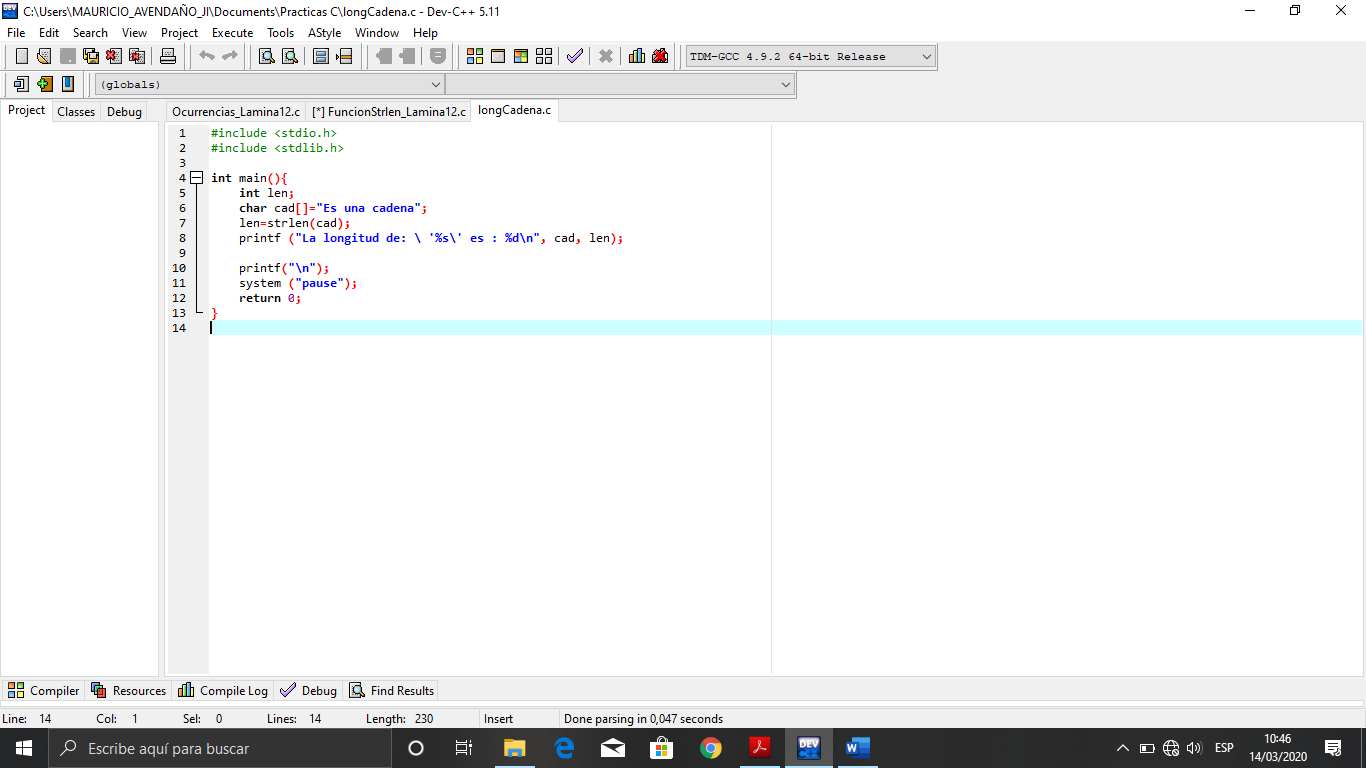
Valor que devuelve el método principal

Método principal

Bibliotecas



Ejemplo. Funciones de cadena: strlen. Devuelve la longitud de una cadena sin tomar en cuenta el carácter final de la cadena.



Con la función strlen obtenemos la longitud de cadena, es decir, cuantos valores tiene, y el resultado se guarda en la variable len, después mostramos la cadena y su longitud en pantalla con printf()

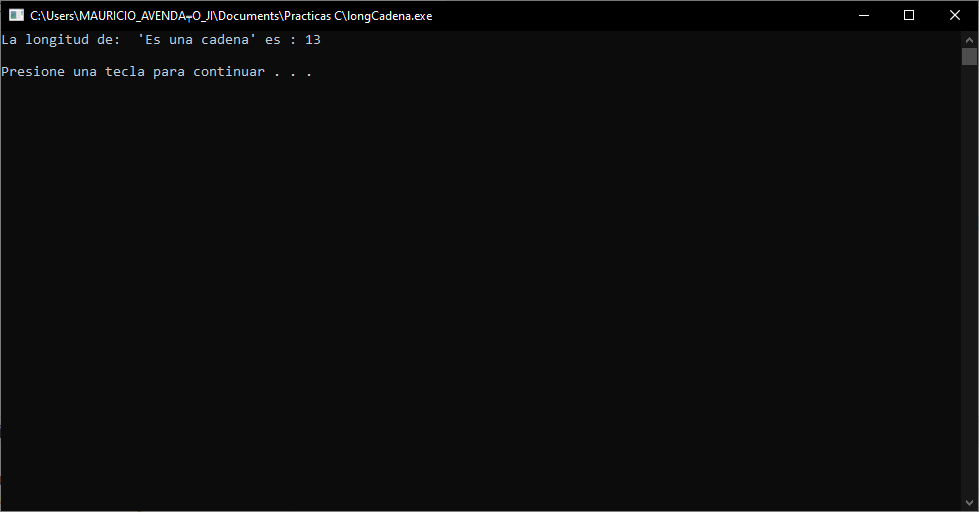
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Declaración de una cadena sin tamañ0, consta del tipo de dato (char), su nombre (cad) y su contenido, el texto entre comillas. Además de una variable llamada len de tipo entero destinada para guardar el valor de la longitud de la cadena.

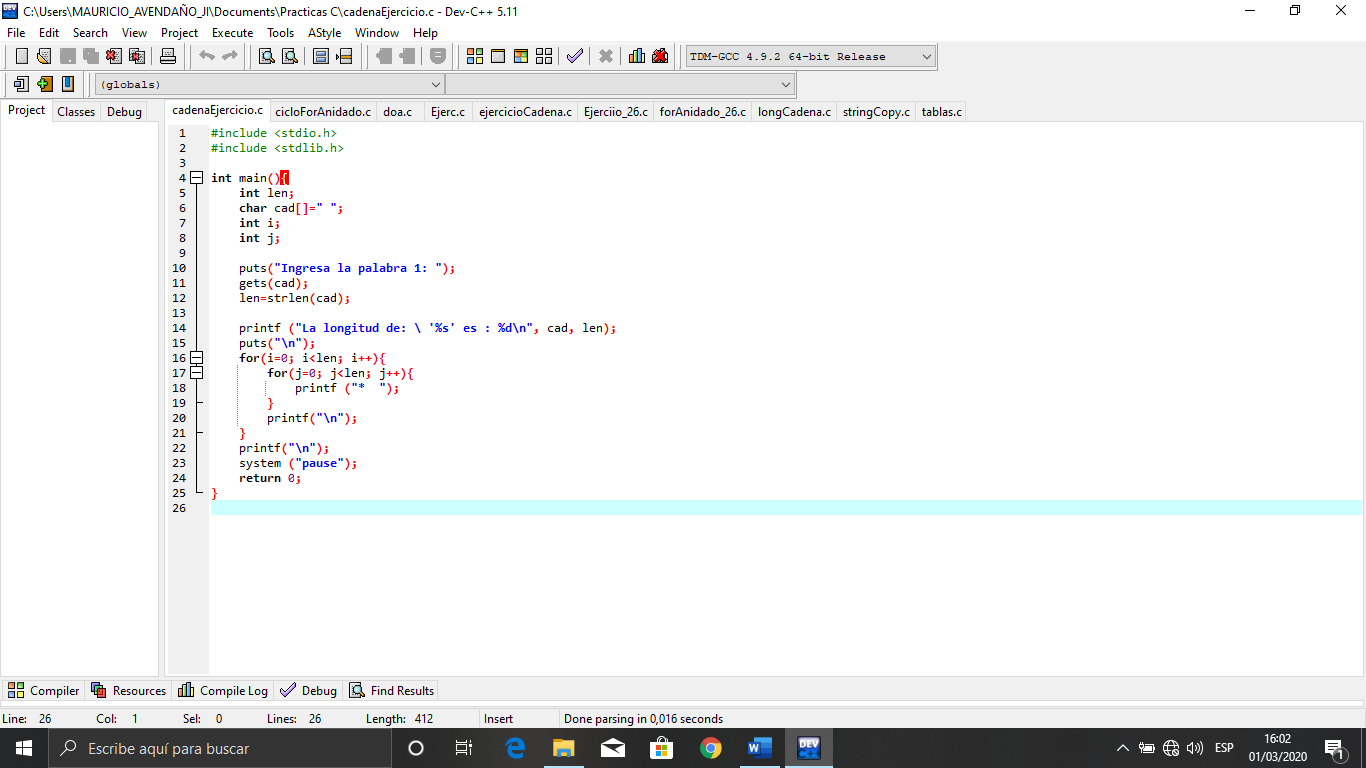
Valor que devuelve el método principal

Método principal

Bibliotecas



Ejercicio: Función strlen. Escriba un programa que reciba una palabra por teclado. De acuerdo a la longitud de la palabra (N) que se ingresó por teclado imprime un cuadrado de asteriscos (NxN).



Este ciclo anidado nos va a ayudar para imprimir un cuadrado con (\*) de n\*n dependiendo de la longitud de la palabra, ambos ciclos los inicializamos en cero, y la condición es que sea menor al tamaño del arreglo por eso están en cero al iniciar, y cada incremento va a imprimir un \*, cuando termine el primer ciclo imprime un salto de línea y así será siempre hasta que termine el ciclo externo.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Imprime el texto la longitud de y concatena el contenido del arreglo es y concatena el tamaño del arreglo, con ayuda de los especificadores de conversión que reciben valores

Declaración de las variables len, i, j de tipo entero, y un arreglo llamado cad de tipo char, inicializado en vacío.

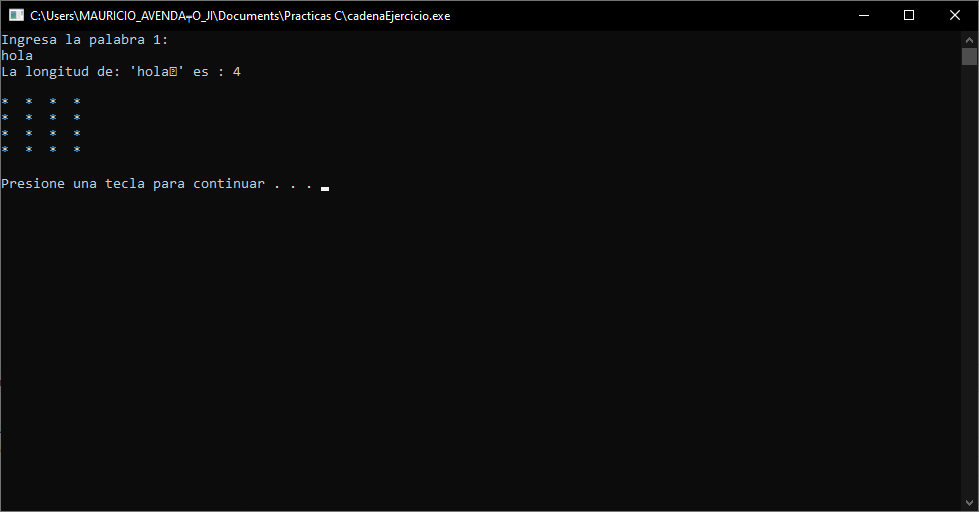
Con puts() vamos a hacer una impresión en pantalla con el mensaje de la palabra 1, con gets() vamos a obtener por teclado los datos y los guardamos en el arreglo cad, y en la variable len vamos a obtener la longitud de la(s) palabra(s), incluyendo espacios.

Valor que devuelve el método principal

Método principal

Bibliotecas

Texto ingresado por teclado, cada carácter ingresado se guarda en el arreglo cad.

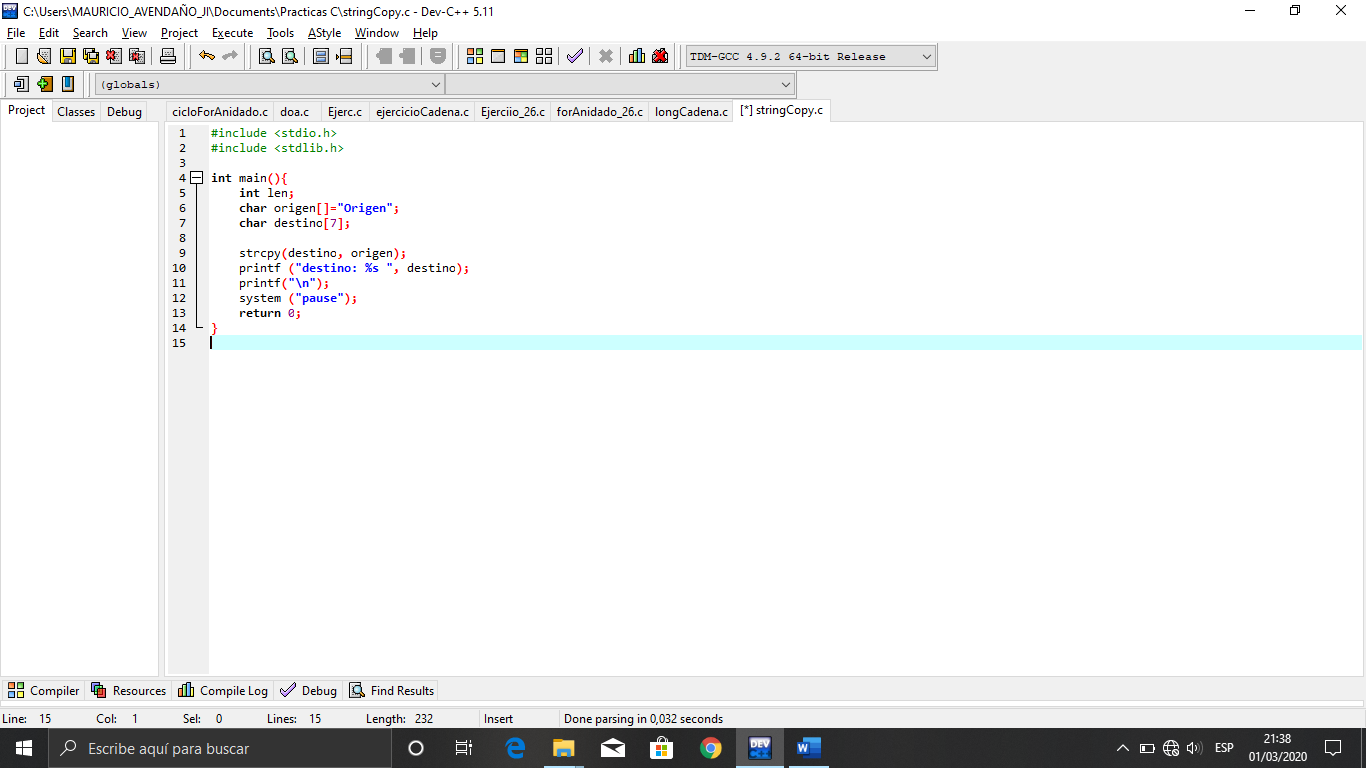


Aquí nos dice el tamaño de la palabra ingresada por teclado.

Como nos podemos dar cuenta, el tamaño de la palabra es de 4 entonces nos imprime un cuadrado de 4\*4 con (\*).

Ejemplo: Funciones de cadena strcpy.

Copia el contenido de la cadena origen a cadena destino.



Bibliotecas

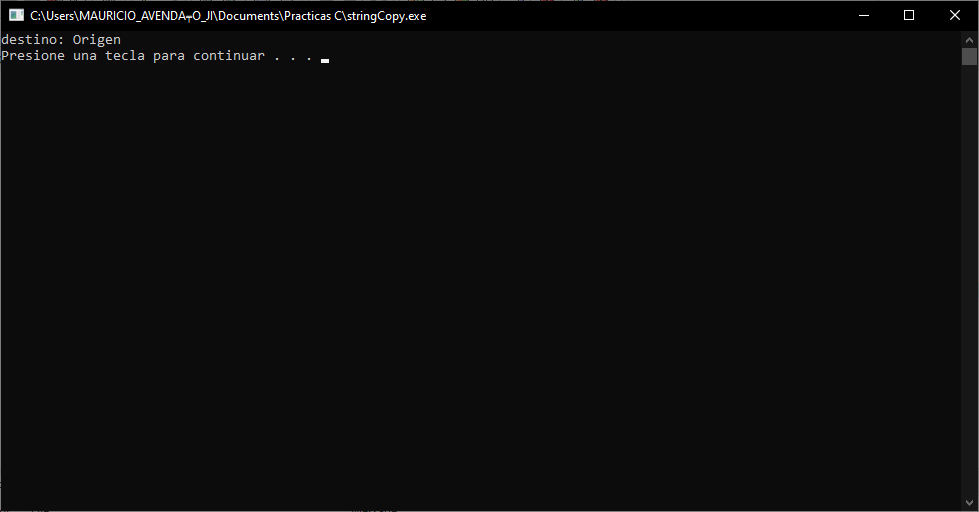
Método principal

Declaración de las variables len, y los arreglos origen y destino, ambos de tipo char, ya que contendrán caracteres, origen ya tiene el texto “Origen” y destino tiene un tamaño de 7.

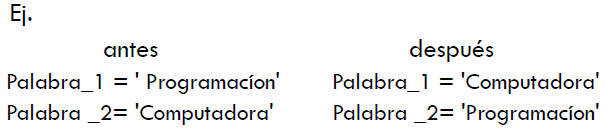
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

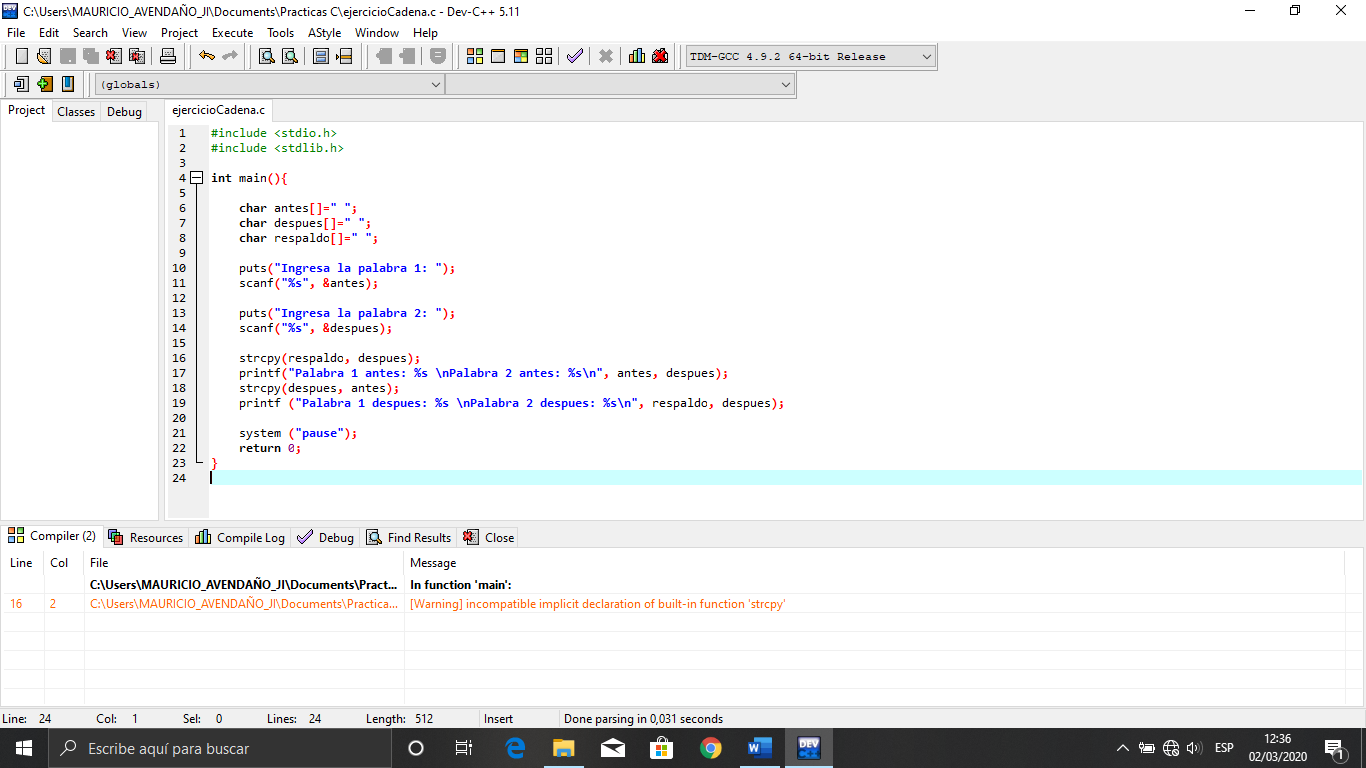
Valor que devuelve el método principal

Aquí hacemos uso de strcpy, su función es tomar lo que tiene una variable y copiarlo en otra, la primera variable hace referencia al destino y la segunda el origen, después va a imprimir lo que tiene la variable destino, ahora tendrá la palabra “Origen”, porque como mencionamos es la copia del arreglo origen que contiene “Origen”. Este valor se asigna al especificador de conversión que está esperando caracteres.



Ejercicio. strcpy. Escriba un programa que reciba por teclado dos palabras y cada una de ellas las almacene en un arreglo. Después intercambia sus contenidos, imprime el antes y el después.





Valor que devuelve el método principal

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Con strcpy, copiamos el valor de alguna variable origen a una variable destino, la sintaxis es: strcpy(variableDestino, variableOrigen);

Primero respaldamos lo que tiene la variable destino, ya que cuando hagamos el copiado se quedara sin valor, entonces respaldamos su valor, y después hacemos el copiado de lo que tiene la1 en la 2, y el respaldo del 2 en la 1, y ya habremos intercambiado los valores.

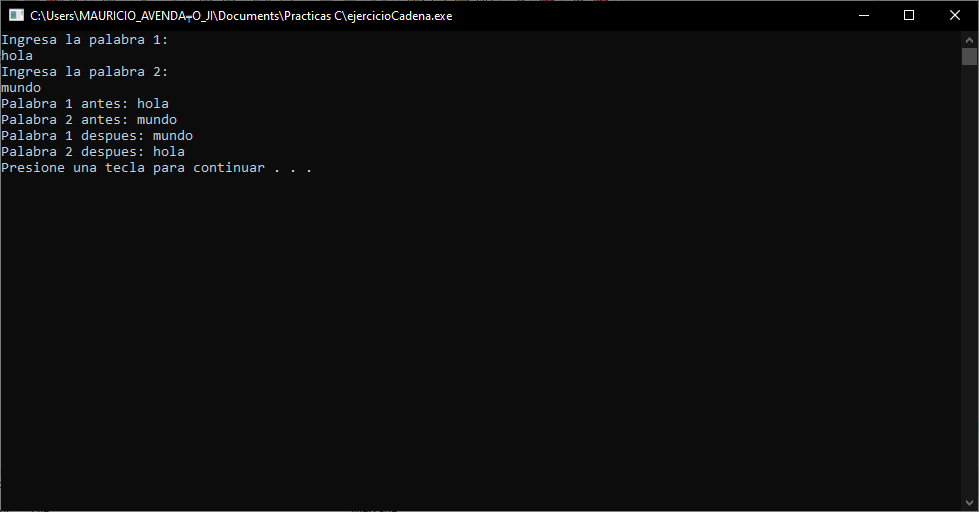
Con puts() podemos mandar un mensaje en pantalla con la instrucción para el usuario y con scanf() podemos obtener el dato que el usuario nos vaya a ingresar por teclado, este se asigna al especificador de conversión en la variable designada para ello.

Lo mismo pasa para pedir la segunda palabra o dato.

Declaración de los arreglos antes, después y respaldo, los tres de tipo char, inicializados con cadenas vacías, ya que aun no sabemos que tamaño tendrán.

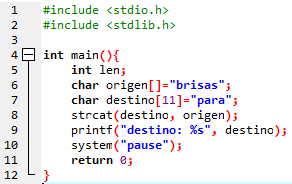
Bibliotecas

Método principal



Ejemplo. Funciones de cadena strcat. Concatena el contenido de una cadena a otra.

Con la función strcat concatenamos brisas a para, y el nuevo contenido de destino es parabrisas, en la siguiente línea, se imprime en pantalla, este valor se asigna al especificador de conversión.



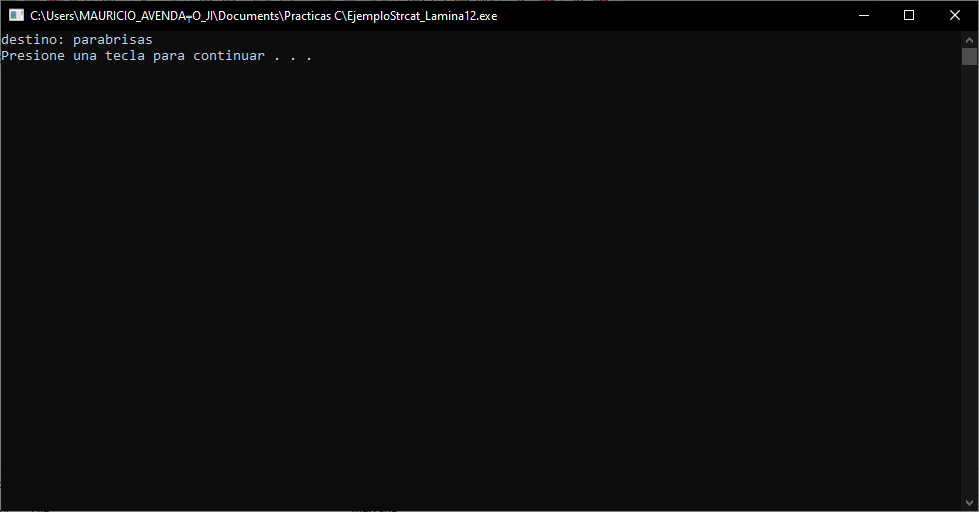
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Declaración de una variable llamada len de tipo entero, y dos cadenas (char), la llamada destino con el contenido “brisas” y la segunda cn un tamaño de 11, pero con un contenido inicial de solo “para”.

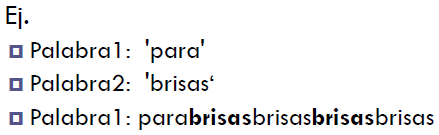
Método principal

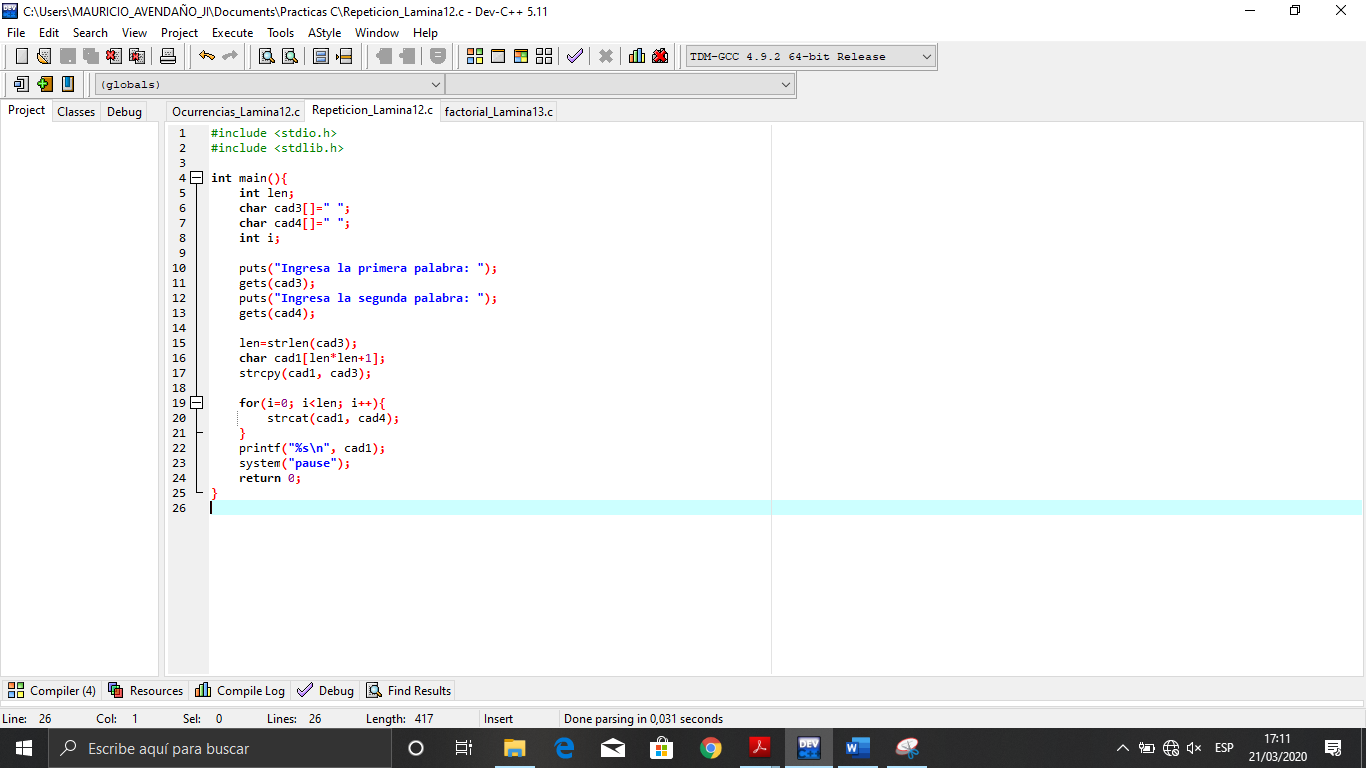
Bibliotecas

Valor que devuelve el método principal



Ejercicio. Strcat. Escriba un programa que reciba por teclado dos palabras y concatene N veces la segunda palabra a la primera palabra. Donde N es la longitud de la primera palabra.





Obtenemos la longitud de cad3 con strlen y se guarda en len, luego se declara otra cadena con el tamaño de len\*len, porque se va a guardar el número de len veces su longitud, y copiamos el contenido de cad4 a cad1, en el ciclo recorremos el numero de veces su longitud y concatenamos la cadena 4 a la cadena 1, y al final se imprime en pantalla.

Arreglos en los que se guardara la cadena ingresada por teclado

En estas 4 líneas, se manda un mensaje al usuario para que ingrese una cadena con ayuda de puts() y con gets() obtenemos la cadena por teclado y la guardamos en arreglo correcto.

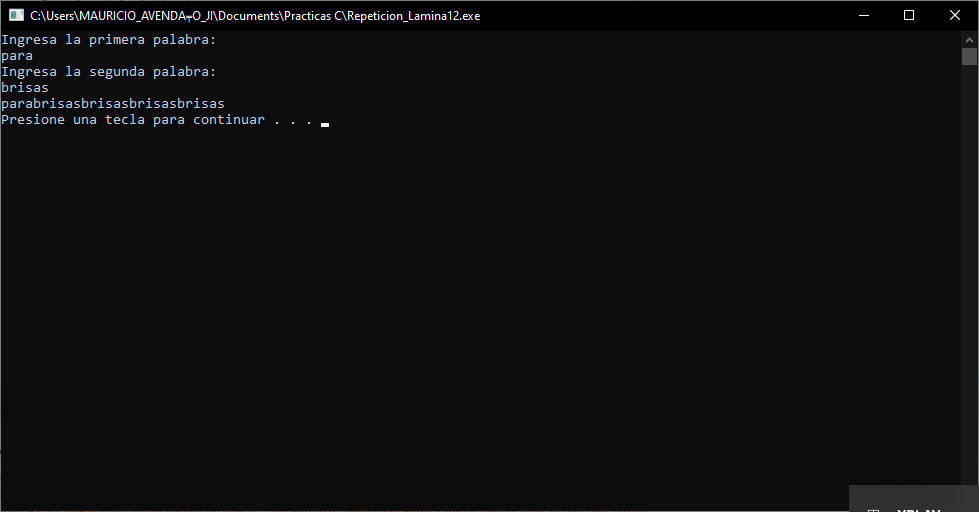
Declaración la variable len de tipo entero que guarda la longitud de cad3, cad3 y cad4 son cadenas (char) e i de tipo entero es el identificador del ciclo. En el código intermedio hay otra declaración de una cadena, esta se hace ahí porque necesitamos el tamaño de cad3, que se obtiene después de haberla ingresado, antes no e hubiera podido porque no sabíamos que longitud tenia.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Método principal

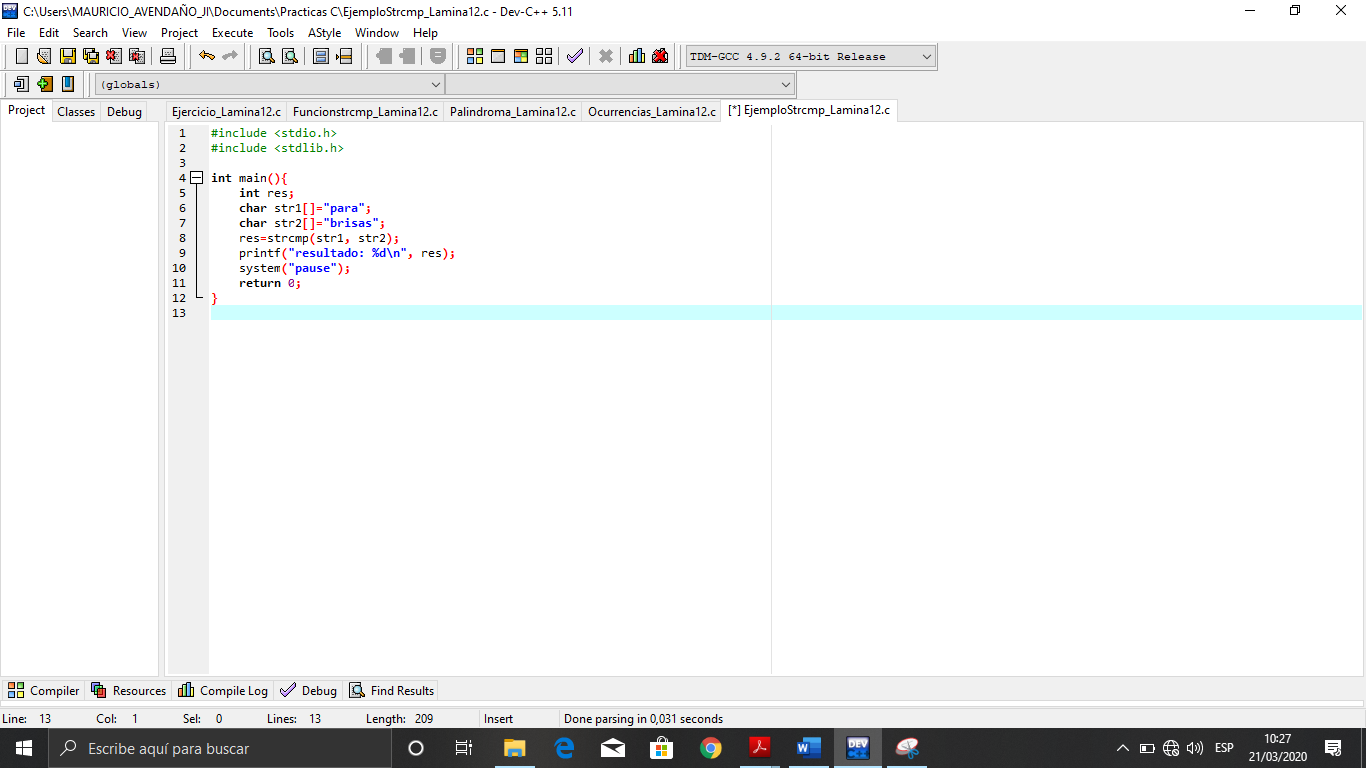
Bibliotecas

Valor que devuelve el método principal



Ejemplo. Funciones de cadena strcmp. Compara dos cadenas y devuelve un 0 si son iguales, un numero negativo si la cadena 1 es mayor que la 2, y un numero positivo si la cadena 1 es menor.

Con la función strcmp comparamos “para” y “brisas”, y el resultado de esa comparación se guarda en la variable res, que después se imprime en pantalla, el valor se guarda en el especificador de conversión de tipo decimal



Declaración de una variable llamada res de tipo entero, y dos cadenas (char), la llamada str1 con el contenido “para” y la segunda str2 “brisas”

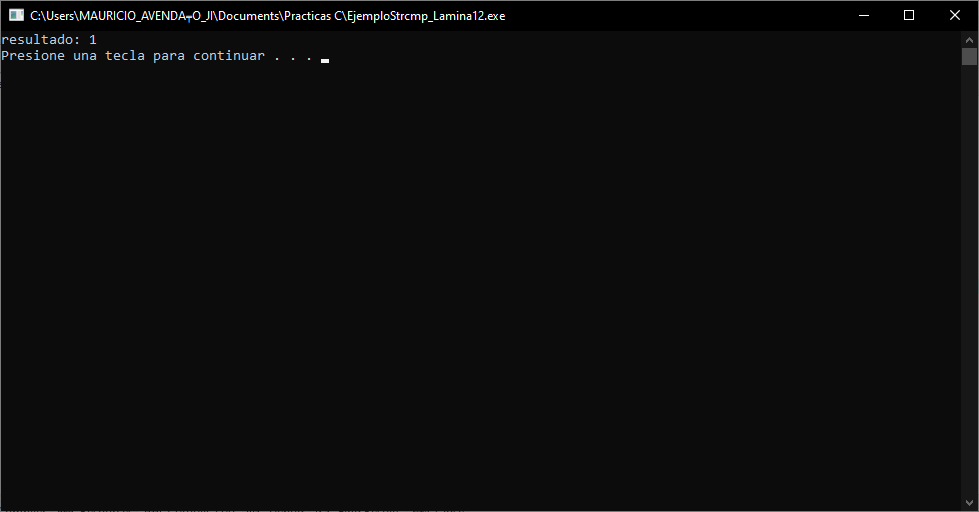
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método principal

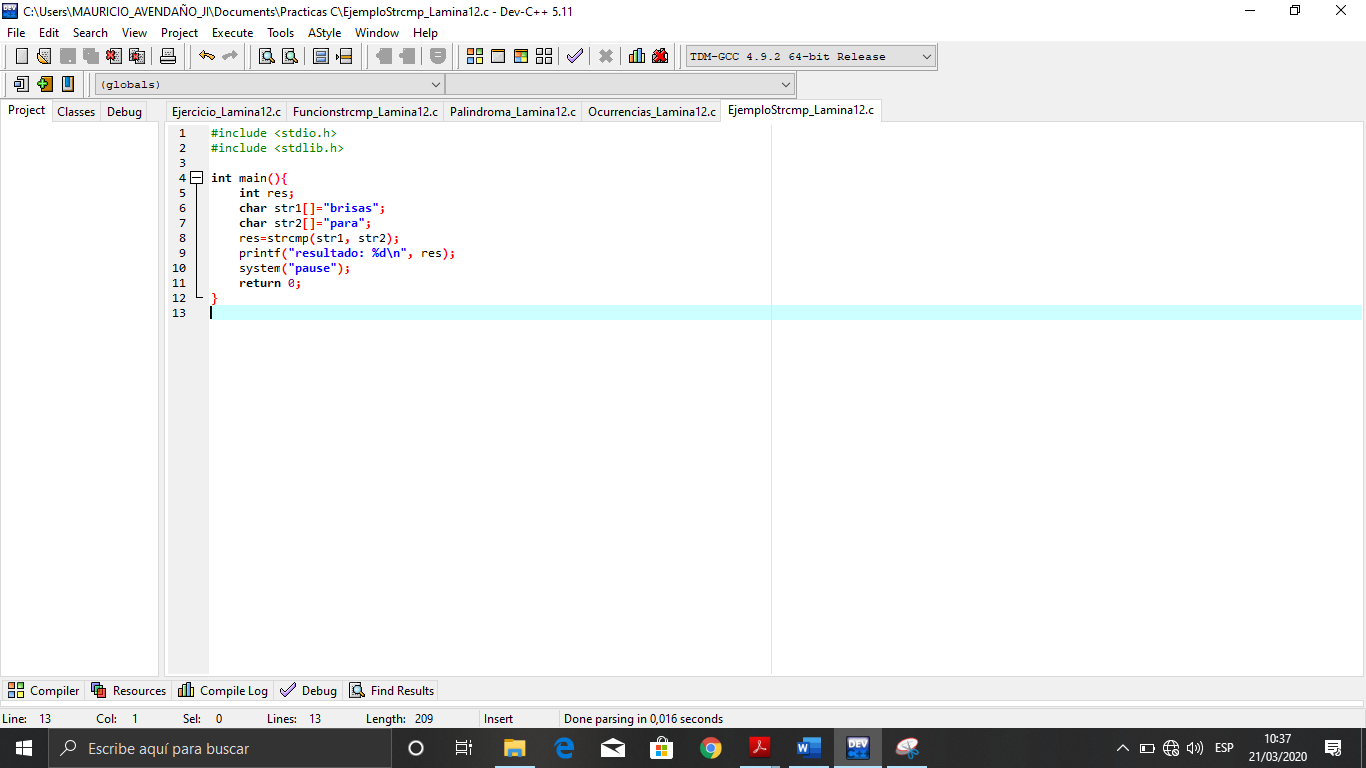
Método principal

Bibliotecas

En este caso, para es menor que brisas, entonces devuelve un numero positivo.



Con la función strcmp comparamos “brisas” y “para”, y el resultado de esa comparación se guarda en la variable res, que después se imprime en pantalla, el valor se guarda en el especificador de conversión de tipo decimal



Método principal

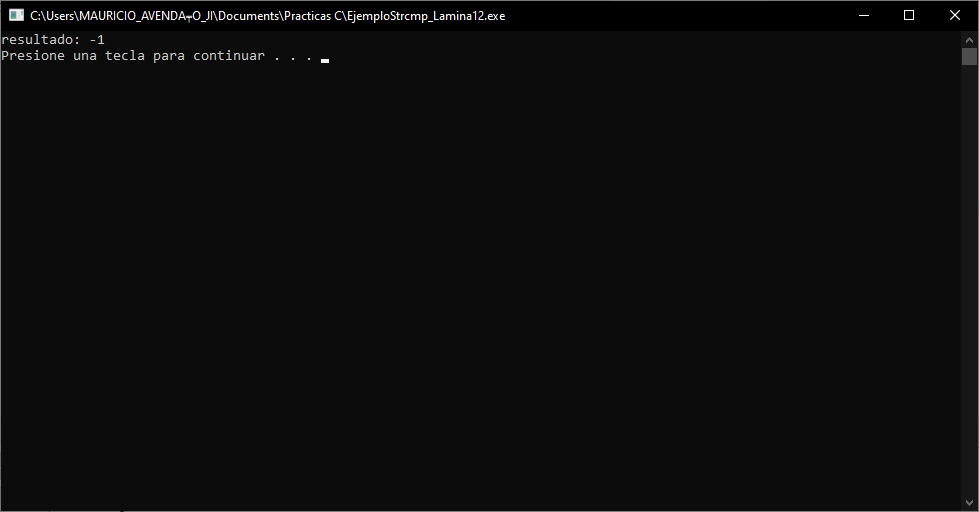
Declaración de una variable llamada res de tipo entero, y dos cadenas (char), la llamada str1 con el contenido “brisas” y la segunda str2 “para”

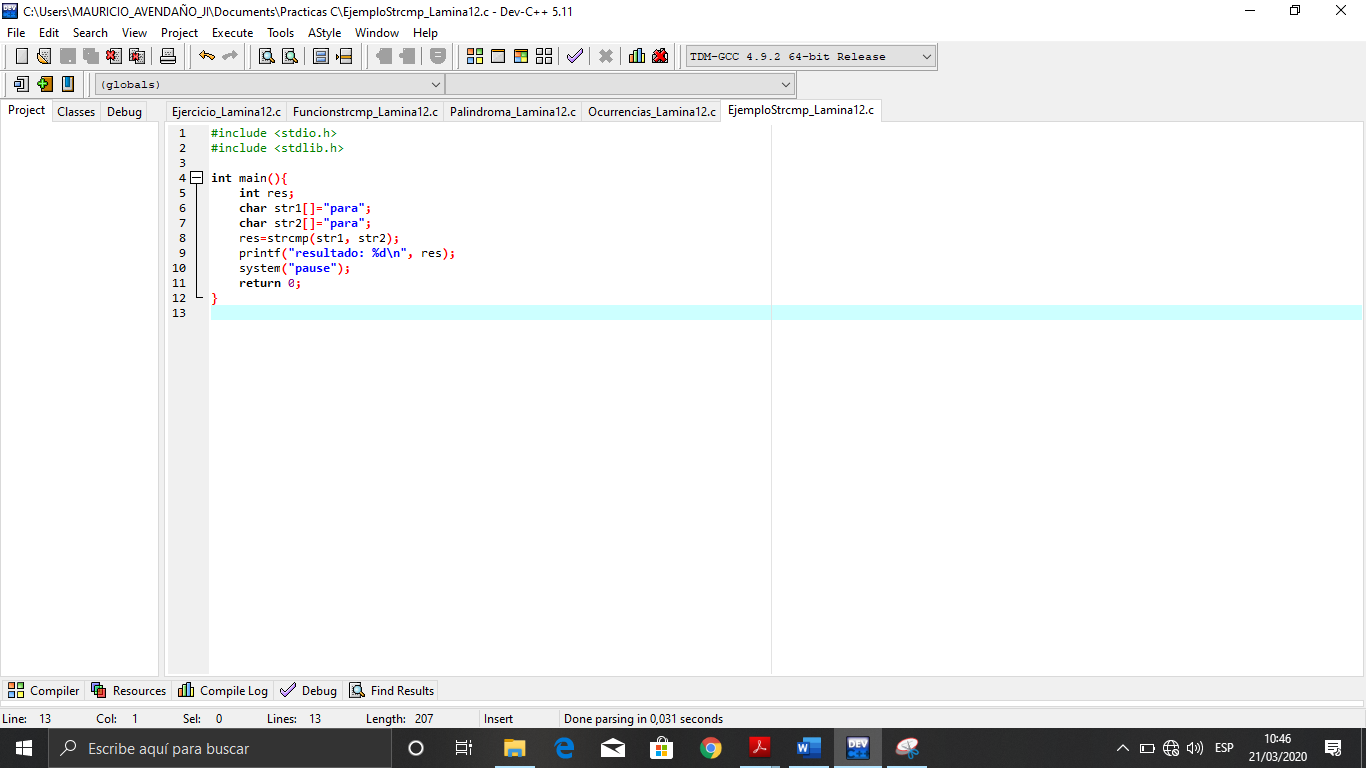
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método principal

Bibliotecas

En este caso, brisas es mayor que para, entonces devuelve un numero negativo.





Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Método principal

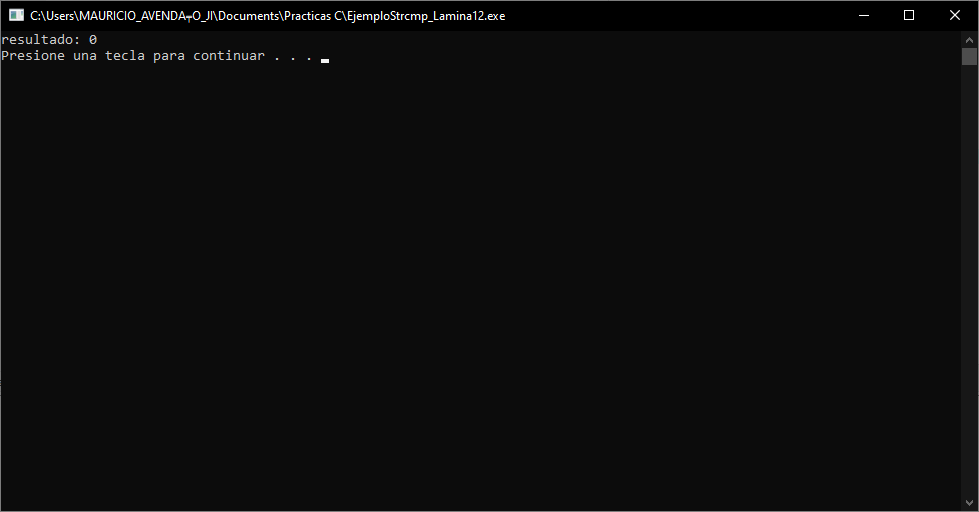
Declaración de una variable llamada res de tipo entero, y dos cadenas (char), la llamada str1 con el contenido “para” y la segunda str2 “para”

Bibliotecas

Valor que devuelve el método principal

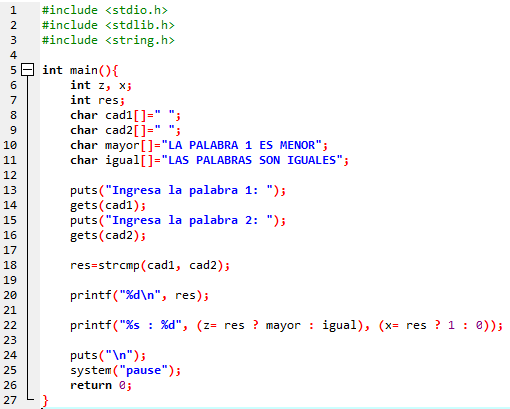
Con la función strcmp comparamos “para” y “para”, y el resultado de esa comparación se guarda en la variable res, que después se imprime en pantalla, el valor se guarda en el especificador de conversión de tipo decimal.

En este caso, para es igual a para, entonces devuelve un 0.



Ejercicio. Strcmp. Escribe un programa que reciba por teclado dos palabras y te indique cual de ellas es mayor y cual es la menor. En caso de ser iguales, que imprima la leyenda “ambas palabras son iguales”.

En estas líneas, con puts() mandamos un mensaje en pantalla, y con gets() podemos obtener los valores, en este caso la cadena que el usuario nos va a ingresar por teclado, el valor se va a guardar en la variable cad1 y cad2 respectivamente, luego con strcmp() se comparan las 2 cadenas y el resultado se guarda en res y posteriormente se imprime.



en esta parte usamos el operador ternario, que nos va a servir para mandar el mensaje adecuado, como sabemos este, utiliza tres valores, el que va a hacer la comparación, y los otros que se van a ejecutar de acuerdo al valor, si es 0 el primero, se ejecuta el 2, si es 1, se ejecuta el 1, es decir, se muestra la cadena mayor o igual, así como el valor.

Declaración de la variable z, x y res de tipo entero, las cadenas cad1, cad2, mayor e igual, como son cadenas son char, cad1 y cad2, están iniciadas como una cadena vacía, esto es porque aun no sabemos que tamaño tendrán estas, ya que se ingresaran por teclado, y mayor e igual tienen un mensaje, después de la comparación va a tomar cualquiera de ellas para mostrarla en pantalla como resultado.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método principal

Método principal

Bibliotecas

Por ejemplo, si las cadenas son iguales, el valor de res va a ser 0 y como la función del operador ternario dice que si es falso (0) ejecutara el segundo valor, y aquí el segundo valor es la cadena igual, si la cadena 1 es mayor que la 2, res va a adoptar el valor de 1, y con el operador ternario si es verdadero (1) entonces ejecuta el primer valor, aquí el primer valor es la cadena mayor. Y así es como, se usa este operador, que su funcionamiento es parecido al de una condicional (if).

El resultado de la operación se asigna al especificador de conversión correspondiente y se imprime en pantalla, el primer especificador es de cadena, ya que la primera operación devuelve un mensaje y el segundo de tipo entero porque devuelve un numero la segunda parte.

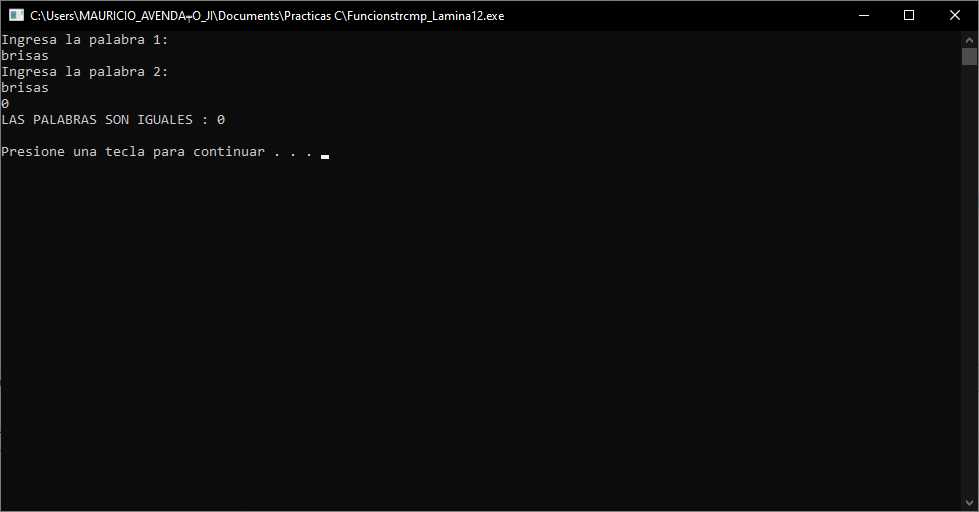
Esta parte funciona exactamente igual que en la primera parte, es otra operación con el operador ternario, solo que en vez de devolver mensajes devuelve un numero

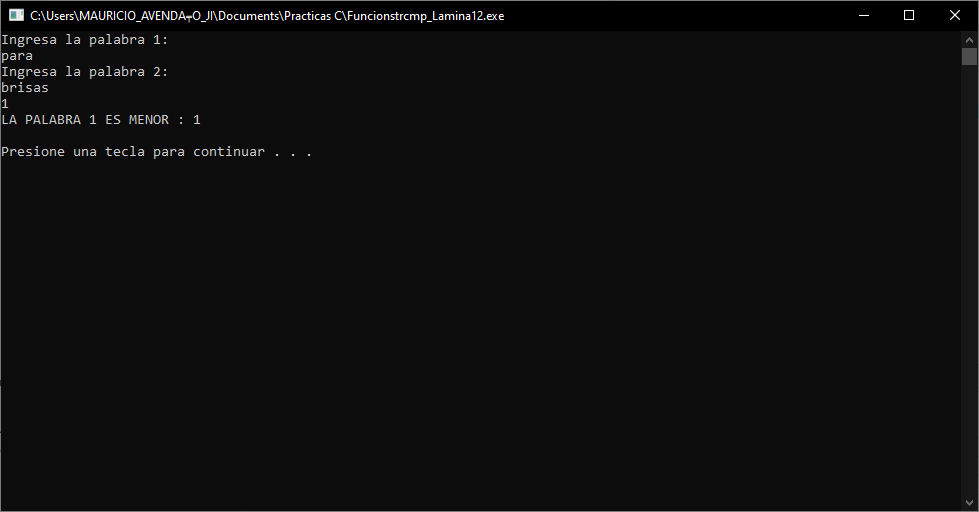


Segundo valor, se ejecutará en caso que el primero sea verdadero (1).

Segundo valor, se ejecutará en caso que el primero sea verdadero (1).

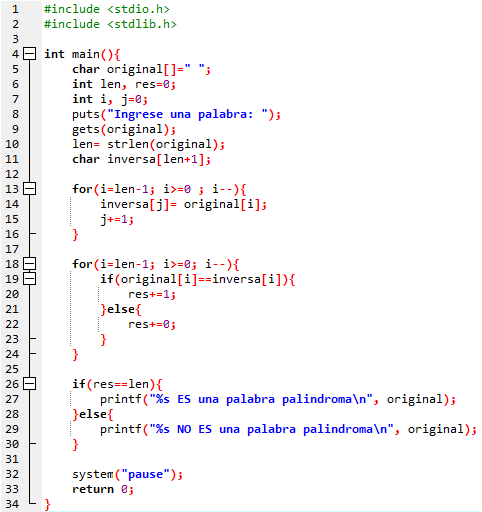
Primer valor, es el que determina que valor entre el 2 y el 3, se va a ejecutar. Dependiendo si su valor en 0 o 1, este trabaja con valores booleanos.





NOTA: el operador ternario solo puede arrojar dos valores, en este caso, o las palabras son iguales o la palabra 1 es menor.

Ejercicio. crea un programa que detecte una palabra palíndroma. Los palíndromos son palabras que se leen igual de izquierda a derecha que de derecha a izquierda. Ejemplo: ala, rotor, salas.



Al último se hacen las comparaciones, si el resultado del acumulador es igual a la longitud de la cadena quiere decir que todas las letras fueron iguales y que en efectivo, la palabra es palíndroma, pero si no, no lo es.

En este ciclo, se hacen las comparaciones, igual se inicia desde atrás hacia adelante del arreglo. Y se comparan las letras, si son iguales se va incrementando el acumulador en 1, si no son iguales los caracteres no incrementa en nada.

En este ciclo, guardamos el contenido de la cadena original en la otra cadena de manera invertida, por eso el ciclo comienza desde atrás hacia adelante, es decir, el ultimo valor de la cadena 1 se guarda en el primer valor de la cadena 2, y así sucesivamente, esto es para saber si son iguales de derecha que de izquierda.

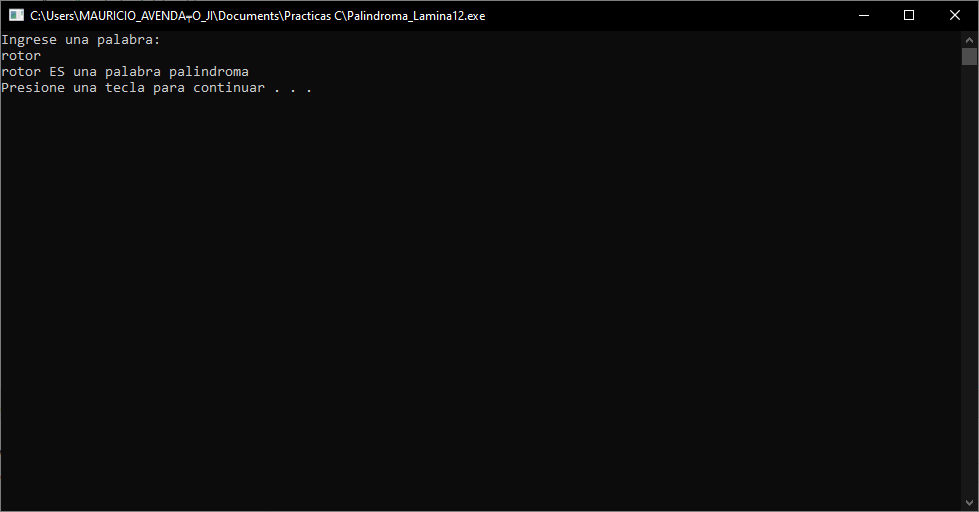
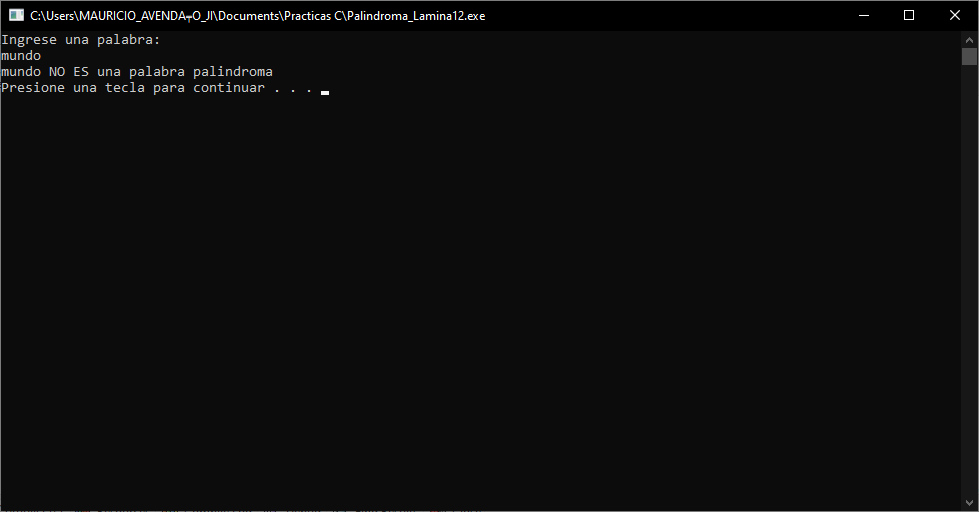
Declaración de la cadena original como una cadena vacía de tipo char, len, res, j e i, de tipo entero, len es para guardar la longitud de la caden original, la obtenemos mediante la función strlen(), res es un acumulador, y j e i, son los identificadores de los ciclos, y la otra cadena llamada inversa guardamos el contenido de cadena original pero al reves y con puts() mandamos un mensaje en pantalla.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

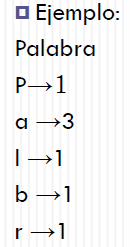
Valor que devuelve el método principal

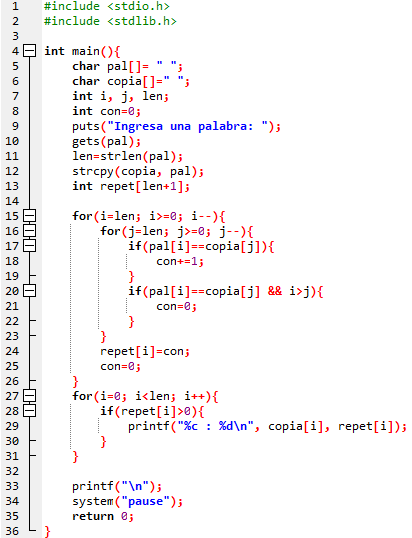
Método principal

Bibliotecas

Ejercicio. crea un programa que cuente cuantas ocurrencias de cada letra contiene una palabra.





En este último ciclo vamos a recorrer de manera normal las cadenas, la que tiene a la palabra y el de cuantas veces se repitió cada letra, nos referimos a manera normal cuando el arreglo se recorre desde el inicio, aquí, se comparan las posiciones actuales, si el valor de cada posición es mayor a 0 se imprime de lo contrario pasa a la siguiente posición.

En el ciclo anidado, empezamos a recorrer las cadenas desde el final y vamos comparando posición por posición si alguna de las letras se encuentra más de una vez, si es así, el contador va a incrementar, pero hay que tener cuidado, si la letra es igual pero ya hay una antes entonces se le va a asignar un valor de 0 para que ya no se imprima una vez más, solo una vez esa letra, después el valor que haya tomado cont se le asignara como valor a la posición indicada del arreglo repet, y así sucesivamente hasta recorrer todo el arreglo

Con ayuda de la función puts() mandamos un mensaje al usuario, con gets() obtenemos por teclado la cadena que el usuario ingrese, strcpy() nos sirve para copiar el mismo contenido a otra cadena en este caso de pal a copia y con strlen() obtenemos la longitud de la cadena pal, la cual le asignaremos este valor+1 como tamaño al arreglo repet de tipo entero.

Declaración del arreglo pal y copia iniciados como una cadena vacía de tipo char, las variables i, j, len y con de tipo entero, con es un contador, y en len se guardará la longitud de pal, y j e i son identificadores del ciclo.

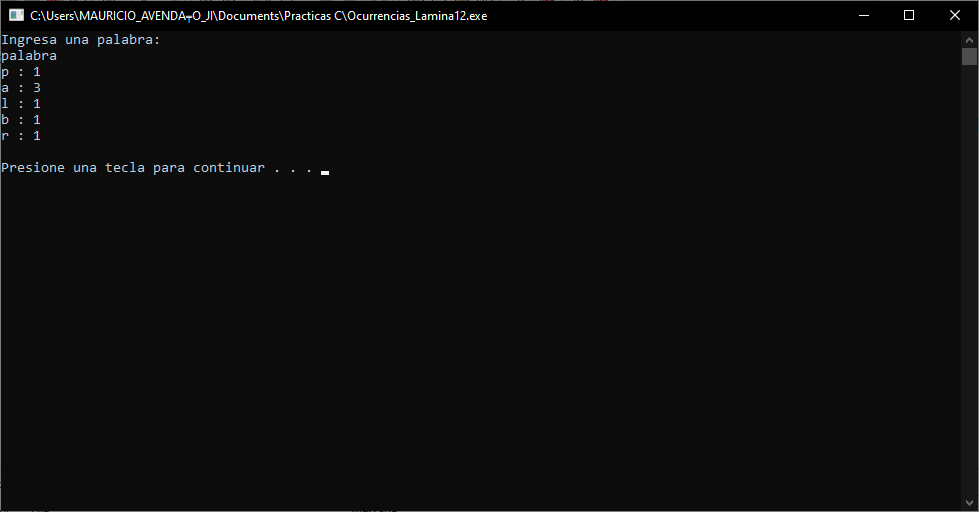
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método principal

Método principal

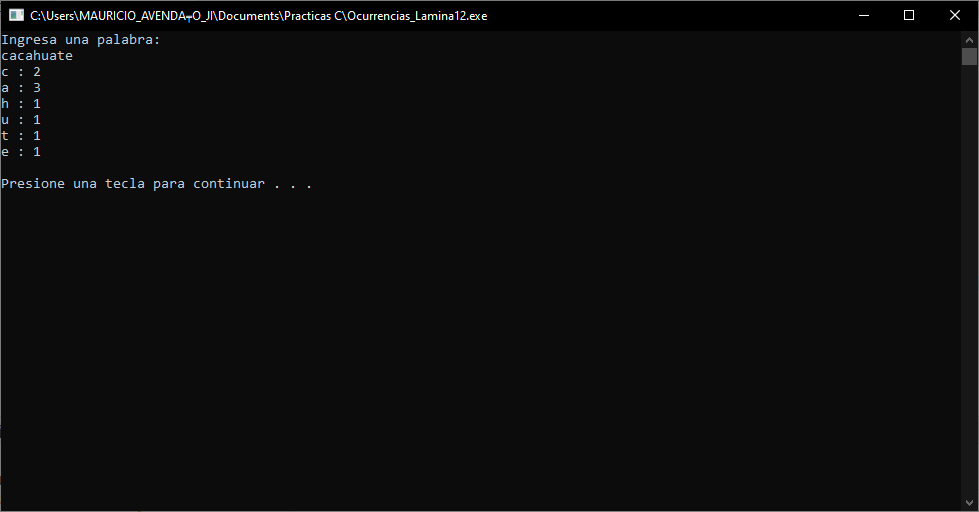
Bibliotecas

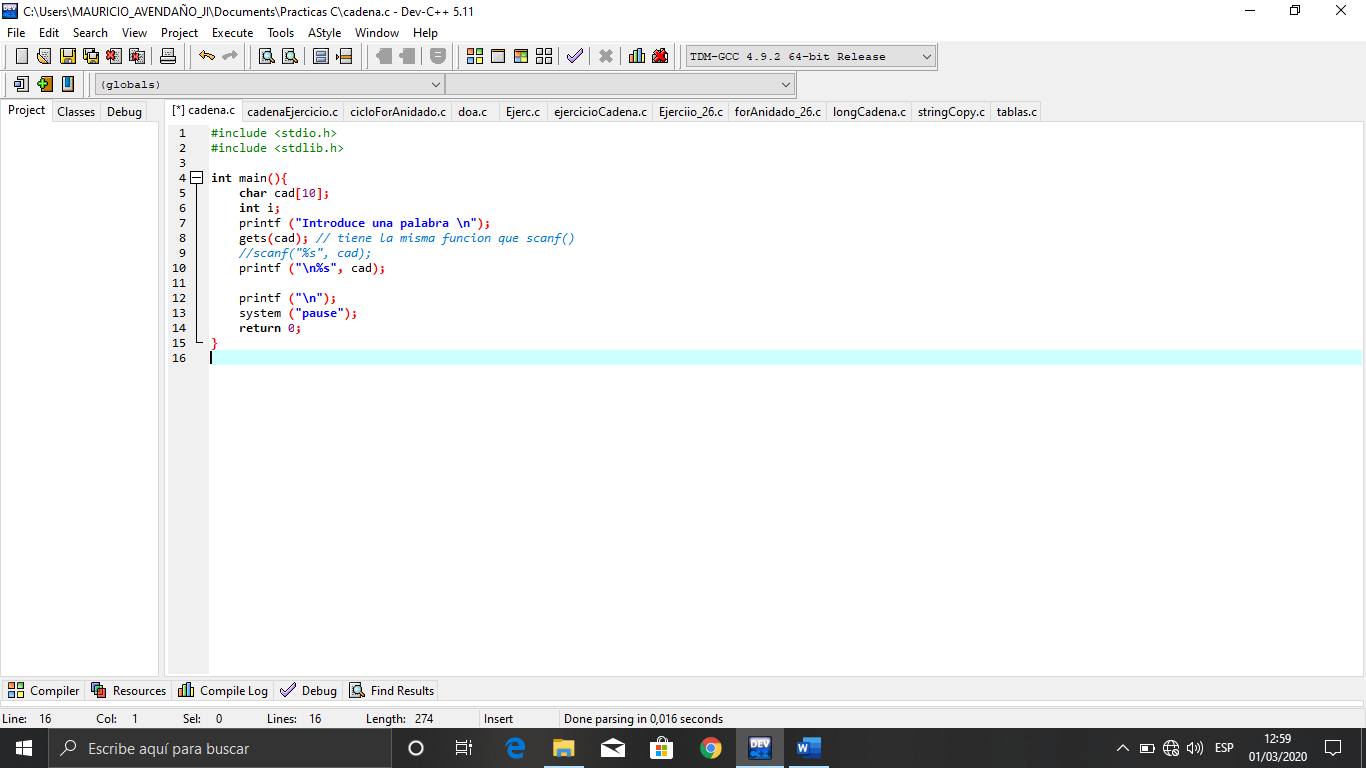
Palabra ingresada por el usuario



A esto nos referíamos cuando se dijo que cuando esa letra ya había sido encontrada antes no incrementaba el contador sino solo le asignaba un 0, esto era para que no entrara en la condición y no se imprimiera, como nos damos cuenta solo aparece una a, y dice que hay 3 de ellas, y el resto solo una, y nos damos cuenta que es correcto

Otro ejemplo:





Bibliotecas

Salto de línea (secuencia de escape)

Comentario, no afectan la ejecución del programa.

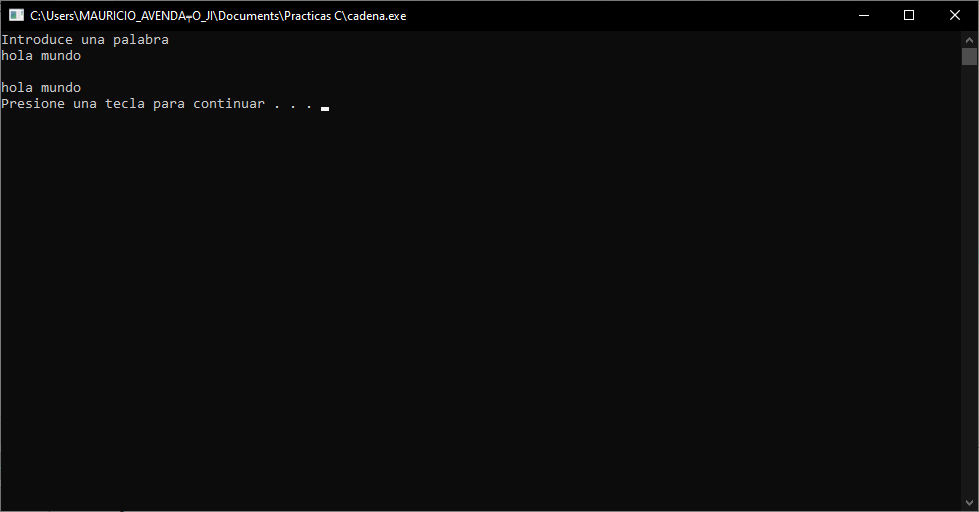
Printf() hace referencia a una impresión en pantalla con la instrucción de ingresar una palabra, esta con ayuda de la función gets(), va a permitir ingresarla por teclado y los caracteres los va a ir guardando en el arreglo cad, después le asigna ese arreglo al especificador de conversión de caracteres y los muestra en pantalla. Es decir, imprime lo que contiene el arreglo.

Método principal

Declaración de un arreglo para caracteres con un tamaño de 10 llamado cad y la variable i de tipo entero.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método principal



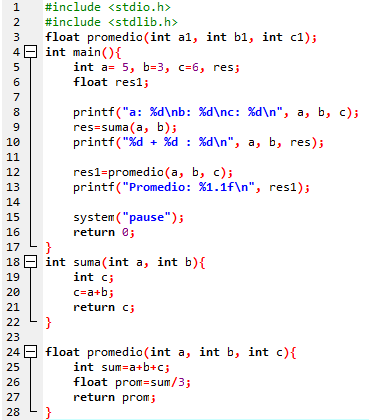
Impresión del arreglo, este contiene lo que anteriormente ingresamos por teclado, cabe mencionar que cada vez que utilicemos un arreglo debemos dar un espacio más como tamaño ya que si no se hace así habrá un desbordamiento de memoria, ya que el ultimo hará referencia al 0, como el final del arreglo.

Texto ingresado por teclado, cada carácter ingresado se guarda en el arreglo cad.

**Documento 13**

Ejemplo. Funciones

Declaración de la función promedio como prototipo.



En el método main, se muestran en pantalla las 3 variables con su valor, en la variable res, guardamos el valor que el método suma nos devuelve, en la llamada le pasamos el valor de a y b, lo mismo para res1, se guarda el valor que devuelve el método promedio, se le pasa el valor de a, b y c.

En el método suma, obtenemos 2 valores como parámetro, en la variable sum de tipo entero, sumamos esos números, y devolvemos el valor de la suma.

Algo semejante pasa en el método promedio, obtenemos 3 valores como parámetros y en la variable sum, sumamos a estos 3, luego dividimos la suma entre los 3 números, y el valor lo devolvemos como el promedio.

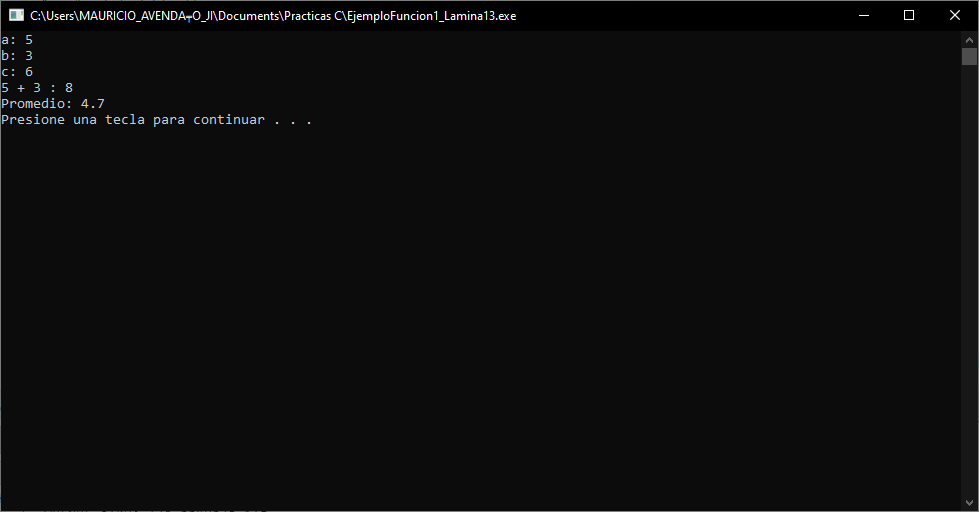
Declaración de variables, 4 de tipo entero llamadas a, b, c, las tres on un valor y res, y una de tipo float res1 que es donde se guardara el promedio.

Valor que devuelve el método.

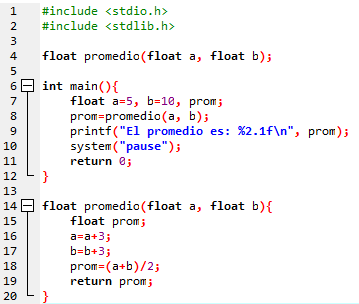
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Método principal

Bibliotecas



Ejercicio. Funciones



En el método imprimir con tipo de retorno flotante, recibe dos parámetros de tipo float, es decir, dos valores que son pasados al momento de llamarlo, en su interior declara una variable llamada prom, aquí a y b funcionan como un acumulador, al mismo valor de a se le agregan 3 e igual a b, el resultado de ambos se suma y se divide entre 2, ese último valor se guarda en la variable prom y será el valor devuelto por el método.

En la variable prom, se guarda el valor que devuelve el método promedio, es decir, ahí se hace la llamada del método y se le pasa el valor de a=5 y b=10 como parámetro, al hacer la llamada este método se ejecuta y devuelve un valor que es el que se guardará en la variable prom, después de ello, se muestra en pantalla, el valor se asigna al especificador de conversión que espera un dato de tipo flotante.

Declaración de las variables a, b y prom de tipo flotante

Declaración de la función promedio como prototipo.

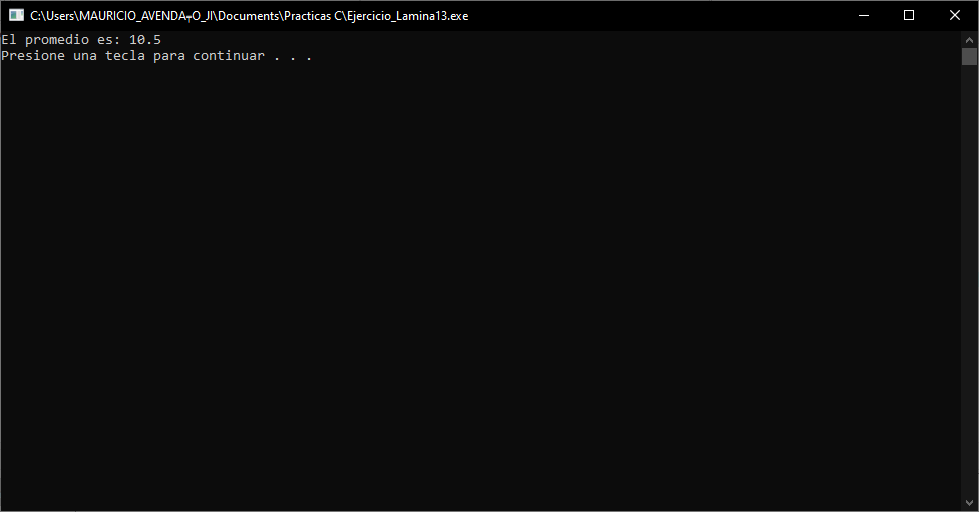
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método.

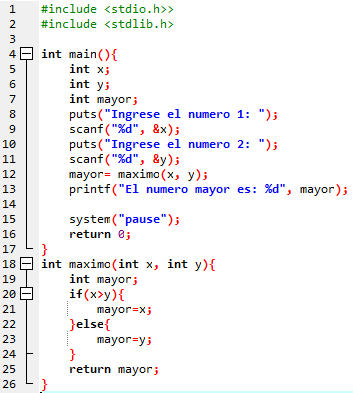
Valor que devuelve el método.

Método principal

Bibliotecas



Ejercicio. Funciones. Escribir una función que se llame máximo, que reciba dos números por parámetros y que regrese el mayor de ellos.



El método máximo con tipo de retorno entero, contiene dos parámetros que son valores que se tomaran a partir de la llamada por el método main, estos valores tienen que ser obligatoriamente, asignados, dentro de este método tenemos una declaración de variable llama mayor de tipo entero, luego se compara xi x es mayor a y, si es así, el valor de x se guarda en mayor y será el valor que se devuelva, si no, entonces será y devuelto.

Mandamos una impresión por pantalla para que el usuario ingrese el número 1 y 2 a comparar, esto se hace por medio de puts(), mediante scanf() nosotros podemos obtener el valor que el usuario nos ha ingresado por teclado, y el valor se guarda en la variable correspondiente, en la variable mayor se va a guardar el valor que el método máximo va a devolver, aquí es donde se hace la llamada a este método, pasándole como parámetros los num. 1 y2, que se encuentran en la variable x y y, después solo imprimimos el valor arrojado por el método máximo.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

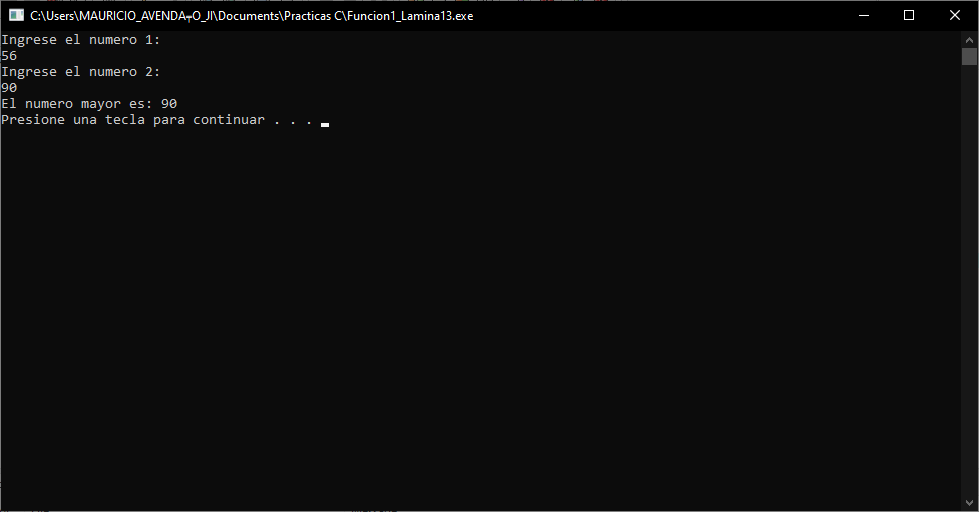
Valor que devuelve el método.

Valor que devuelve el método.

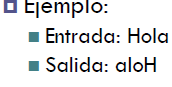
Declaración de las variables x, y y mayor de tipo entero.

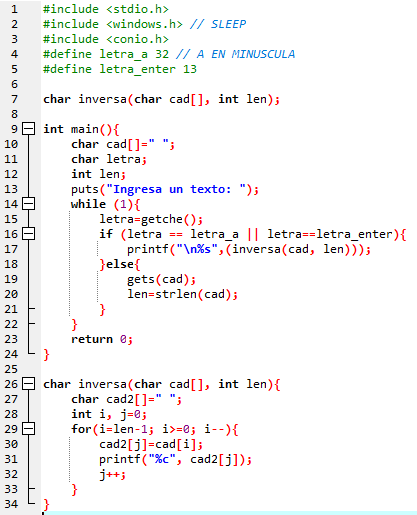
Método principal

Bibliotecas



Ejercicio. escribir una función que reciba caracteres del teclado hasta recibir un espacio o salto de línea (enter) y a continuación mostrar todos los caracteres en orden inverso.





Aquí se hace la llamada al método, pasándole como parámetro a la cadena y la longitud de ella.

El método inversa se encarga de imprimir al revés la cadena ingresada por teclado, primero declara una segunda cadena donde va a ir la cadena al revés, luego declara a i y j de tipo entero, y el ciclo comienza desde la última posición, y va decrementando, en cambio en cad2 se va iniciando desde 0 e incrementa, le asigna la posición i a la j de la cad2, y la imprime, j incrementa.

Declaración de la cadena cad, iniciada como vacía, la variable letra de tipo char y len de tipo entero.

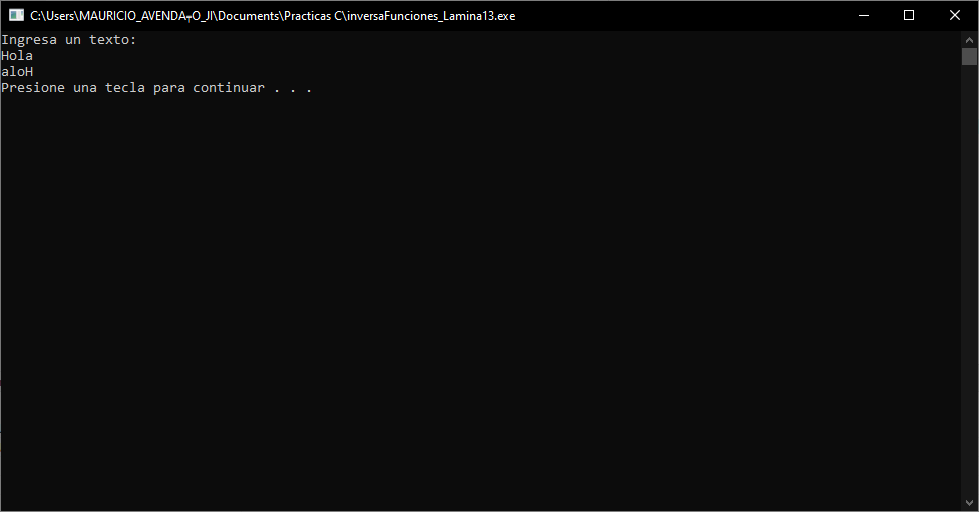
Primero se manda un mensaje al usuario, luego dentro de un ciclo se introducen los datos, pero aquí hacemos uso de una biblioteca para obtener el valor de la tecla pulsada usando el código ASCII, como el espacio es 32 y enter es 13 los declaramos como constantes y comparamos si la letra es igual a uno de esos valores si si es asi, llamamos al método inversa pero si no seguimos pidiendo los datos y cada vez se va a ir contando la longitud de la cadena ingresada

Método principal

Valor que devuelve el método.

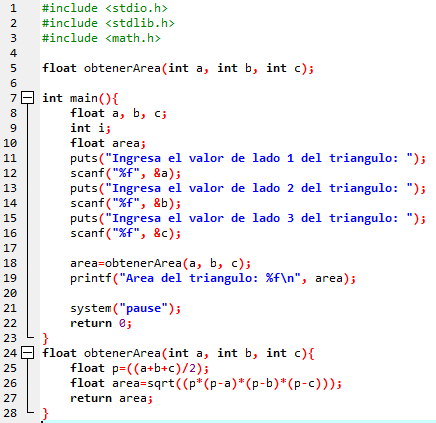
Declaración de la función inversa como prototipo.

Bibliotecas



Ejercicio. Escribir una función que tome como parámetros las longitudes de los tres lados de un triángulo (a, b, c) y devuelva el área del triángulo. Formula de Heron:





En la variable área, se guarda el valor que retorna el método obtenerArea, asi mismo en la llamada le pasa como parameros el valor de a, b y c, y después imprime el valor que adopta área.

El método obtenerArea con tipo de retorno float, contiene 3 parámetros, que son los valores de los lados del triángulo, valores que serán proporcionados por el método main, al momento de hacer la llamada, dentro se declara a p de tipo float, que guarda el resultado de la división de la suma de todos sus lados entre 2,

Con la función puts() podemos mostrar un mensaje en pantalla, con scanf() obtenemos el valor introducido por teclado por parte del usuario, el valor se asigna a la variable correcta.

Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Valor que devuelve el método.

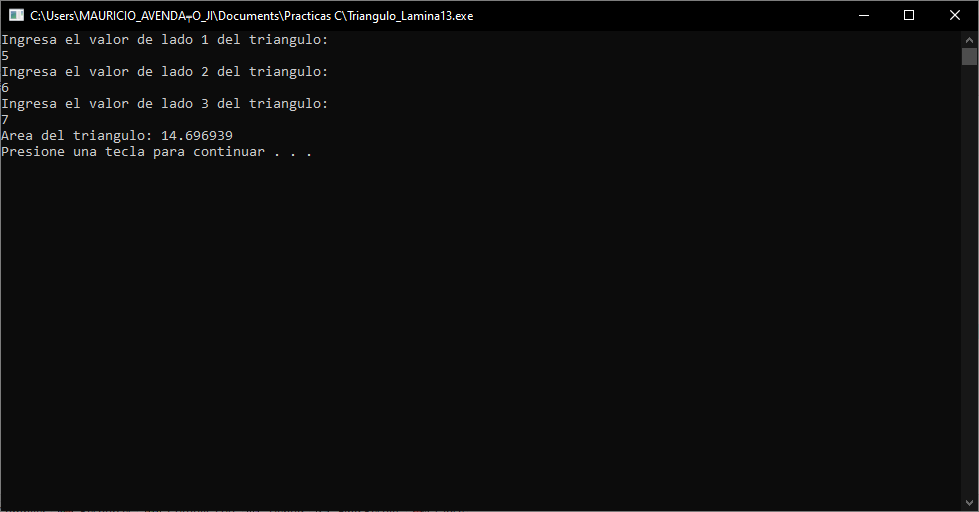
Declaración de las variables a, b y c de tipo flotante, i de tipo entero y área igual de tipo flotante.

Declaración de la función obtenerArea como prototipo.

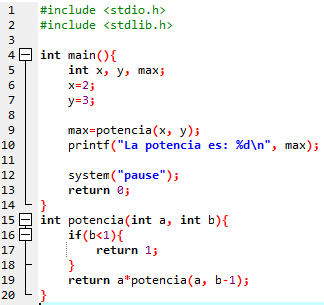
Método principal

Bibliotecas

Valor que devuelve el método.



Ejemplo. Funciones recursivas. Calcular la potencia de forma recursiva.



Declaración de las variables x, y y max, asi mismo hay una asignación de un valor a x y y,

Dentro del método main, se manda a llamar al método potencia que es recursivo, aquí se devuelve un valor final, que se va a guardar en le variable max, después se imprime, esta variable mediante el especificador de conversión %d

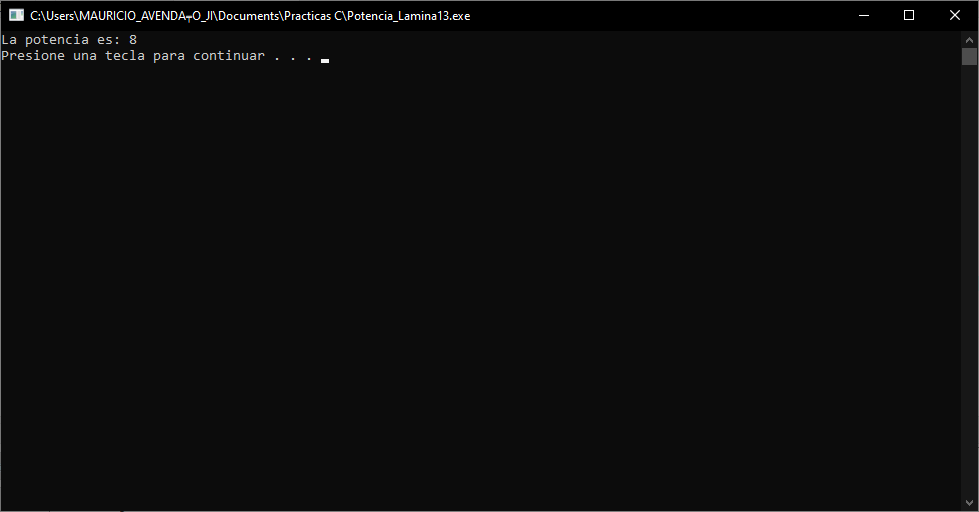
Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Método principal

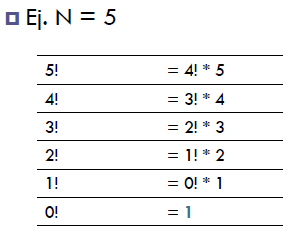
Valor que devuelve el método.

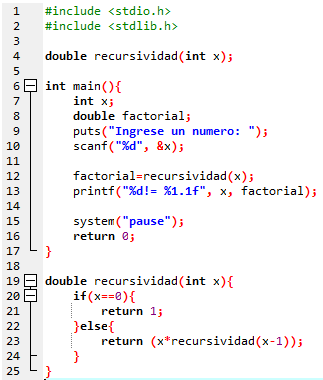
Bibliotecas

El método potencia es un método recursivo, es decir, que su función es parecida a la de un ciclo, pero debe terminar alguna vez, es por ello que se auxilia de una condición, su función es realizar un proceso y volverse llamar a si mismo. En este ejercicio nos pide la potencia, nos dice que x vale 2 y yo vale 3, entonces estos valores se le pasan al método potencia, y hace que a tome el valor de x y b el valor de y. ahora bien compara si b es menor a 1, como no lo es va a seguir el otro proceso, va a retornar el valor de b -1, ahora vale 2, en la siguiente va a pasar lo mismo, ahora vale 1, y lo mismo pasa, en la última llamada le paso el valor de 0, ahora va a ir liberando los números generados, libera el 0 y lo multiplica una vez a a\*b, puesto que se decremento 3 veces, entonces, se va a multiplicar a por ese número de veces, primer 2\*1=2, luego 2\*2=4 y al final 4\* 2=8, y ese es el resultado de la potencia.



Ejercicio. Funciones recursivas. Haz un programa con funciones recursivas que calcule la factorial de un numero n ingresado desde teclado.





Función que nos permite detener la ejecución del programa y visualizar los resultados en pantalla.

Como ya habíamos mencionado anteriormente, se hace la llamada al método recursividad pasándole el valor de x, que es el número al cual obtendremos su factorial, una vez que se hace el proceso dentro de ese método, este va a devolver un valor que se guardara en la variable factorial, enseguida de ese proceso mostramos en pantalla el resultado.

Declaración de las variables x de tipo entero y factorial de tipo float, con ayuda de la función puts() mandamos una impresión en pantalla, y con scanf() obtenemos la entrada por teclado.

Método principal

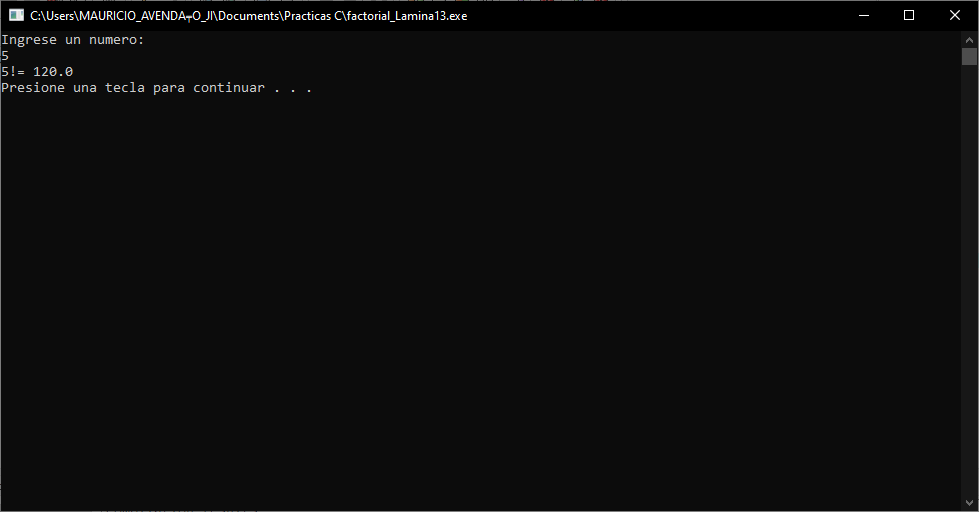
Valor que devuelve el método.

Bibliotecas

Declaración de la función recursividad como prototipo.

Cuando se llama al método recursividad le estamos diciendo que debe ejecutar todo el código que tiene, para ello nos pide un valor como parámetro, este valor, es el que pedimos por teclado, una vez que tenemos el valor compara si dicho número es igual a 0, porque recordemos que la factorial de 0 es 1, por ello es que devolvemos el valor de 1, pero si la comparación resulta falsa, entonces estaremos llamando al método dentro de si mismo, pero con la diferencia es que cada vez que se invoque le estaremos restando un valor al número original, después como los resultados se van a ir apilando los tiene que liberar y es ahí cuando estaremos multiplicando por el resultado anterior.

Por ejemplo, tenemos a 5, entra en el método y compara 5<1=falso, entonces hace el otro proceso, 5-1=4, guarda este valor y se lo vuelve a pasar como parámetro 4-1=3 y así sucesivamente hasta el 1. Digamos que 4 fue el primero entonces es como si estuviera en la parte trasera, y 1 hasta el principio, entonces el valor de x se ira multiplicando por cada valor hasta llegar al 4, entonces es: 5\*1=5, 5\*2=10, 10\*3=30, 30\*4=120, por lo que 120 es la factorial de 5.



**V. Conclusiones:**

La recursividad es muy importante, ya que en muchas ocasiones puede suplir a estructuras repetitivas, que en varias ocasiones ocupan o requieren de mas espacio en memoria, además este es eficaz, y es un poco más eficiente, ya que funciona como método y ciclo, además al dejar de cumplir con una condición este termina su ejecución.