

(Versión de fecha 14 de noviembre de 2024)

La mayoría de los programas que desarrollamos a lo largo del curso constan de un solo fichero fuente .c. En la práctica de la programación es frecuente que los programas consten de varios ficheros fuente .c, algunos de los cuales se acompañan de sus correspondientes ficheros de cabecera .h. En este documento se va a explicar cómo compilar un programa compuesto de dos ficheros .c más un fichero .h, tanto desde el terminal, como desde el propio entorno de desarrollo VSCode.

1 CÓDIGO DEL EJEMPLO

Para desarrollar la explicación se va a utilizar un programa de ejemplo que consta de los siguientes ficheros fuente:

- **main.c**: programa principal de la aplicación. Básicamente tiene la función *main()* y declara en un *#include* el acceso a la librería de funciones *libreria.c*.
- **libreria.c**: código fuente de las funciones que ofrece la librería auxiliar utilizada por el programa principal. Contiene dos funciones:
 - *maximo()*: esta función tiene dos parámetros. El primero es una referencia a un array de números enteros y el segundo es un número entero que indica el número de elementos de los que consta el array del primer parámetro. La función devuelve un número entero con el valor del máximo elemento del array que recibe como argumento.
 - *suma()*: esta función también tiene la misma definición de parámetros que la función *maximo()*. Devuelve un número entero con la suma de los elementos del array que recibe como primer argumento.
- **libreria.h**: fichero de cabecera de *libreria.c*. Contiene los prototipos de las funciones de *libreria.c*.

Tienes que crear un proyecto en VSCode para albergar los ficheros del ejemplo. Para ello, utiliza la opción del menú *Fichero -> Abrir carpeta* y crea una nueva carpeta donde guardarás los archivos. En nuestro caso, la hemos llamado *CompilacionVariosFuente*.

A continuación, desde dentro del editor, ve creando los ficheros *libreria.h*, *libreria.c* y *main.c* y copia el código que reproducimos a continuación.

Ejemplo 1 Código del fichero *libreria.h*

```
#pragma once

// Calcula el valor máximo entre los elementos del
// array que recibe como argumento
int maximo(int array[], int num_elementos);

// Calcula la suma de los elementos del array
// que recibe como argumento
int suma(int array[], int num_elementos);
```

Observa el código anterior. En primer lugar se declara la directiva de precompilación *#pragma once* y, a continuación, se declaran los prototipos de las funciones cuyo código está en el fichero *libreria.c*.

La directiva *#pragma once* le indica al compilador que, aunque el *#include libreria.h* aparezca en varios ficheros .c, al compilar el programa solo se debe incluir una vez. Recuerda que la instrucción *#include* lo que le dice al compilador es que copie en ese lugar el código del fichero indicado.

Ejemplo 2 Código del fichero *libreria.c*

```
#include "libreria.h"

int maximo(int array[], int num_elementos) {
    int valor_max = array[0];
    for(int i=0; i<num_elementos; i++) {
        if(array[i]>valor_max) {
            valor_max = array[i];
        }
    }
    return valor_max;
}

int suma(int array[], int num_elementos) {
    int suma_valores = 0;
    for(int i=0; i<num_elementos; i++) {
        suma_valores += array[i];
    }
    return suma_valores;
}
```

La primera línea de código es *#include libreria.h*, que incluye la cabecera del propio fichero. Aunque en este ejemplo no sería necesario, es una precaución recomendable, pues pudiera suceder que algunas funciones de la librería hicieran llamadas a otras funciones de la misma librería. De esta forma, no hay que preocuparse del orden en el que se escriben las funciones en *libreria.c*, pues los prototipos aparecieran antes que el código de las mismas.

El resto del fichero *libreria.c* contiene el código de las dos funciones de la librería.

Acerca de *#include*

Observa que, cuando se incluye una librería del sistema, por ejemplo *stdio.h*, el nombre de la librería se pone entre corchetes:

```
#include <stdio.h>
```

Esto indica que el fichero correspondiente se encuentra en alguno de los directorios que tiene definidos el compilador para las librerías del sistema ^a.

En cambio, para la librería del programa, hemos puesto el nombre de la librería entre comillas:

```
#include "libreria.h"
```

Esto indica al compilador que debe buscar el fichero en el directorio actual. Si el fichero estuviera en otro directorio, deberías poner entre comillas la ruta completa hasta el fichero.

^aSi estás en un sistema Windows y has hecho la instalación del compilador *gcc* en el directorio *C:\msys64*, puedes visitar con el explorador de archivos de Windows la carpeta *C:\msys64\ucrt64\include* y comprobar que ahí está el fichero *stdio.h* y muchos otros ficheros .h que se suelen incluir en los programas.

Solo queda mostrar el código del programa principal *main.c*. El código del fichero *main.c* es el siguiente:

Ejemplo 3 Código del programa principal *main.c*

```
#include <stdio.h>
#include "libreria.h"

int main() {

    int nums[] = {1, -1, 2, 4, 6, -3};

    int valor_max = maximo(nums, 6);
    int suma_valores = suma(nums, 6);

    printf("Máximo: %d\n", valor_max);
    printf("Suma  : %d\n", suma_valores);

    getchar();
}
```

El programa anterior declara en primer lugar los *include* de la librería del sistema *stdio.h* y del fichero *libreria.h*. Observa que, en este último caso, se pone el nombre de la librería entre comillas. Observa también que es la segunda vez que usamos *#include "libreria.h"* en los ficheros del programa, pues ya lo habíamos hecho en el fichero *libreria.c*. Como se ha explicado, solo se incluirá una vez, pues el fichero *libreria.h* lleva la directiva *#pragma once*.

Dentro de la función *main()*, se declara e inicializa un array de 6 números enteros. A continuación, se hacen sendas llamadas a las funciones *suma()* y *maximo()*, pasando dicho array y su número de elementos como argumentos y recogiendo el resultado en unas variables llamadas *suma_valores* y *valor_max*. Por último, se muestran en pantalla los resultados.

El aspecto final del proyecto en VSCode debería ser similar al de la siguiente figura:

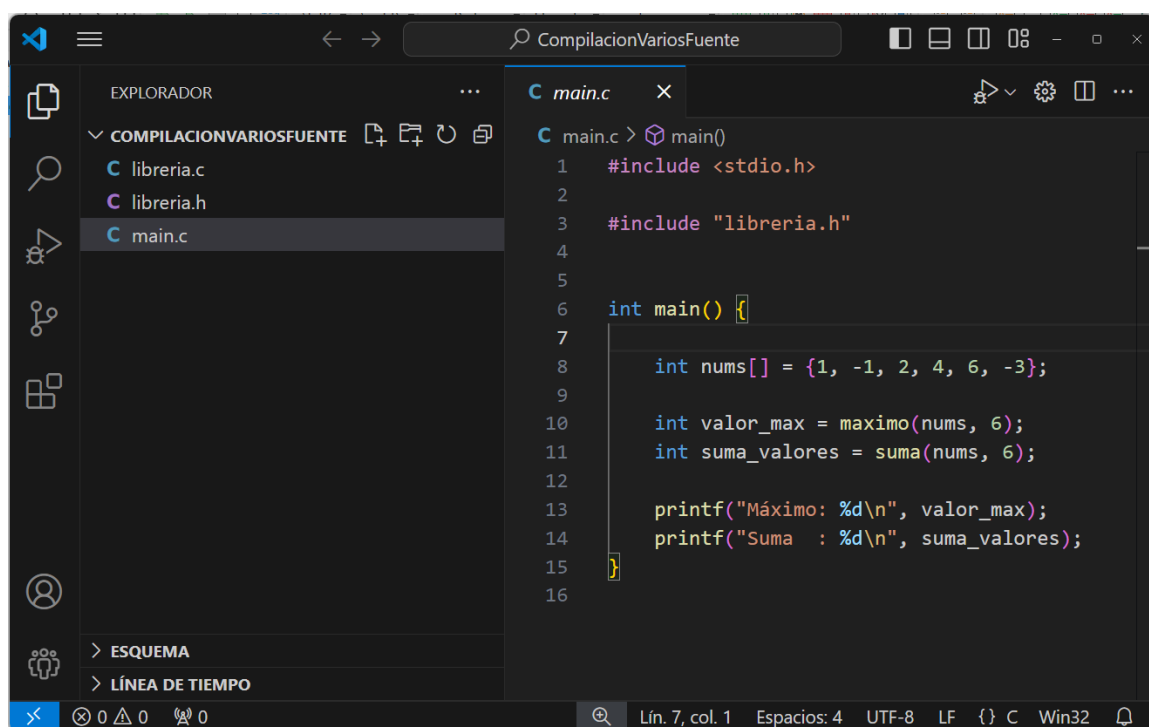


Figura 1: Escritorio de VSCode tras añadir el código de los fichero .c

2 COMPILACIÓN DEL PROGRAMA EN EL TERMINAL

Pincha en el editor de texto de VSCode, dentro del código de uno de los ficheros del proyecto y utiliza el atajo de teclado **CTRL+SHIFT+C** para abrir un terminal posicionado en el directorio raíz del proyecto.

Ahora, para compilar el proyecto, hay que teclear la siguiente instrucción en el terminal:

```
gcc main.c libreria.c -o main.exe
```

Esta instrucción ejecuta el compilador *gcc*, al que le pasa el nombre de los ficheros que se quieren compilar y el parámetro *-o*, que le indica el nombre del fichero ejecutable que queremos generar (la *o* se refiere a *output*).

Compilación en Mac y Linux

Si estás trabajando en Mac o en Linux, la instrucción para compilar es la siguiente:

```
gcc main.c libreria.c -o main
```

Solo se distingue de la instrucción Windows en que el fichero de salida no lleva la extensión *.exe*

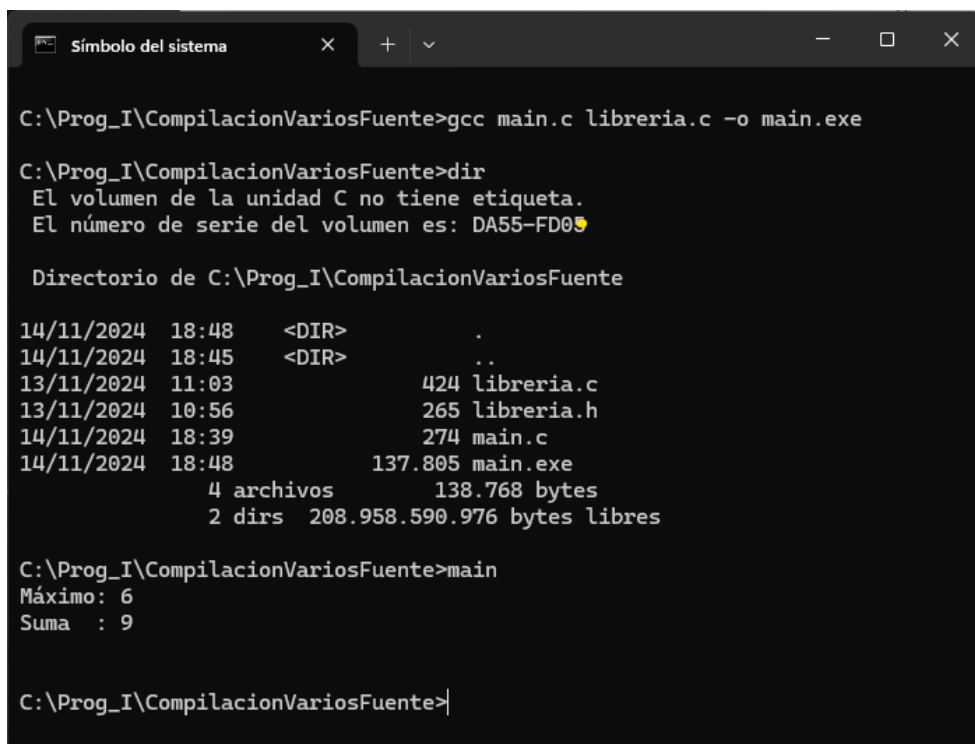
Si se produce algún error durante la compilación, se mostrarán los mensajes correspondientes. Si la compilación finaliza con éxito, el terminal quedará preparado para ejecutar el programa. Para ejecutarlo, debes teclear la siguiente instrucción si estás en un sistema Windows:

```
main
```

Si estás en Mac o en Linux, para ejecutar el programa debes teclear:

```
./main
```

En Windows, la salida por pantalla debería ser similar a la siguiente (en la figura, tras compilar y antes de ejecutar el programa, se ha tecleado la orden *dir*, para ver el contenido del directorio y comprobar que el fichero *main.exe* se había generado):



```
Símbolo del sistema
C:\Prog_I\CompilacionVariosFuente>gcc main.c libreria.c -o main.exe

C:\Prog_I\CompilacionVariosFuente>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: DA55-FD09

Directorio de C:\Prog_I\CompilacionVariosFuente

14/11/2024  18:48    <DIR>          .
14/11/2024  18:45    <DIR>          ..
13/11/2024  11:03                424 libreria.c
13/11/2024  10:56                265 libreria.h
14/11/2024  18:39                274 main.c
14/11/2024  18:48           137.805 main.exe
               4 archivos           138.768 bytes
               2 dirs  208.958.590.976 bytes libres

C:\Prog_I\CompilacionVariosFuente>main
Máximo: 6
Suma : 9

C:\Prog_I\CompilacionVariosFuente>
```

Figura 2: Compilación y ejecución del programa de pruebas

3 COMPILAR Y EJECUTAR DENTRO DE VSCODE

Para poder compilar el programa dentro de VSCode, lo primero que debemos hacer es generar la carpeta `.vscode` del proyecto con los ficheros `tasks.json` y `launch.json`. Para ello, pincha con el botón derecho del ratón dentro del código de uno de los ficheros del programa. Se te abrirá un menú flotante en el que debes elegir la opción *Agregar configuración de depuración* y, a continuación, seleccionar el compilador. Como resultado, verás que se ha creado una carpeta llamada `.vscode` con los ficheros `tasks.json` y `launch.json` dentro de ella.

El fichero `tasks.json` es el que le dice a VSCode cómo se debe compilar el proyecto. Haz doble click con el ratón sobre él para que se abra en el editor. Tenemos que modificar las dos líneas que se destacan en la siguiente figura:

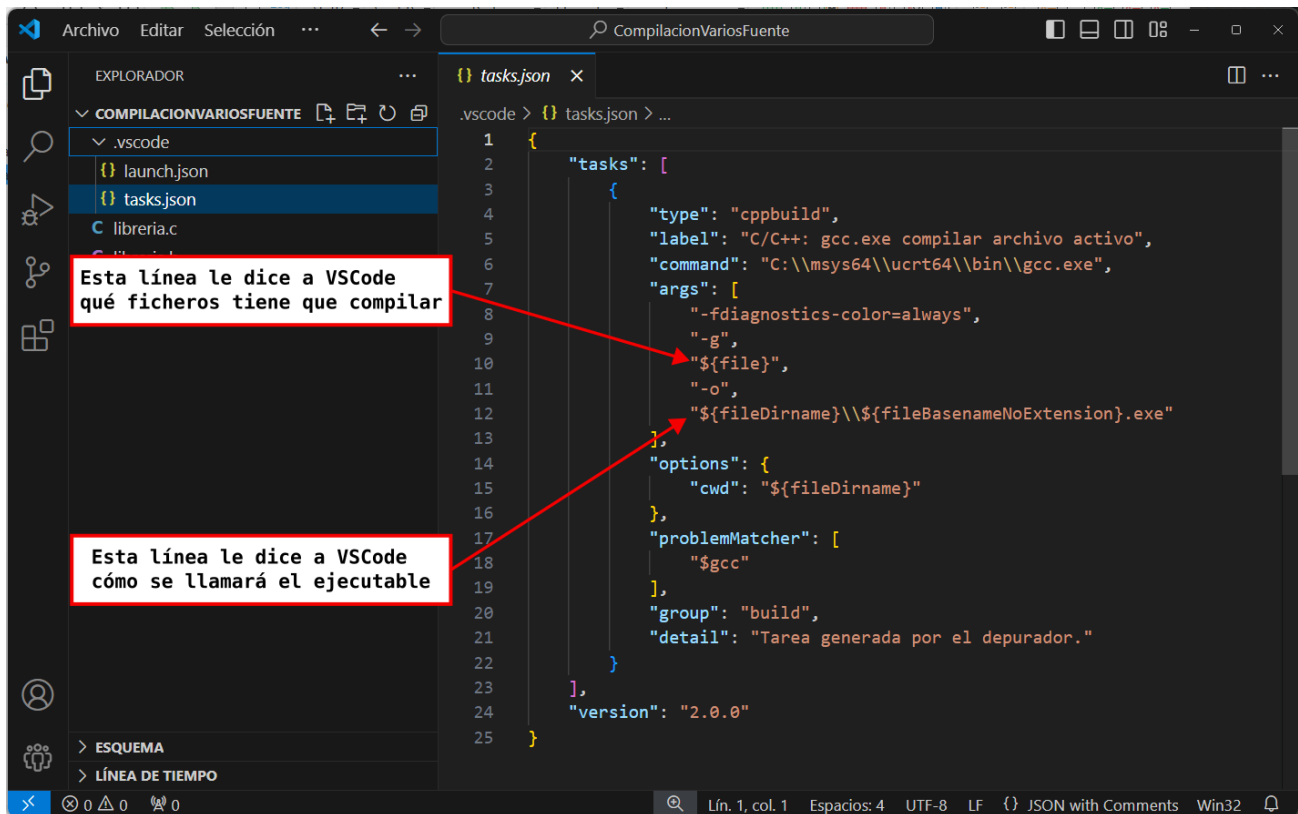


Figura 3: Líneas que hay que modificar en el archivo de configuración `tasks.json` para compilar varios ficheros

Para decirle a VSCode qué ficheros debe compilar, sustituye el contenido `"${file}"`, de la línea correspondiente por lo siguiente (no olvides las comas):

```
"main.c", "libreria.c",
```

Para decirle a VSCode el nombre del fichero ejecutable, sustituye el contenido de la línea correspondiente por `"main.exe"`, si estás en Windows, o `"main"` si estás en Mac o Linux (en ambos casos, entre comillas).

Ahora ya podemos compilar el programa. Para ello, abre en el editor el fichero `main.c` y selecciona la opción de menú *Terminal -> Ejecutar tarea de compilación* o bien utiliza el atajo de teclado `CTRL+SHIFT+B`. Como resultado, deberías ver en la parte inferior de la pantalla el resultado de la compilación.

Para ejecutar el programa, tienes que tener en el editor el fichero `main.c` y seleccionar la opción de menú *Ejecutar -> Ejecutar sin depuración (CTRL+F5)* o bien *Ejecutar -> Iniciar depuración (F5)*.

