

TEMA 3:

Fundamentos para la construcción de código a partir del algoritmo

TEMA 3:

3.1 Sintaxis básica y semántica.

Objetivo

- El alumno construirá programas utilizando el lenguaje de programación C a través de un análisis y modelado algorítmico previo.

Ingredientes para programar en C

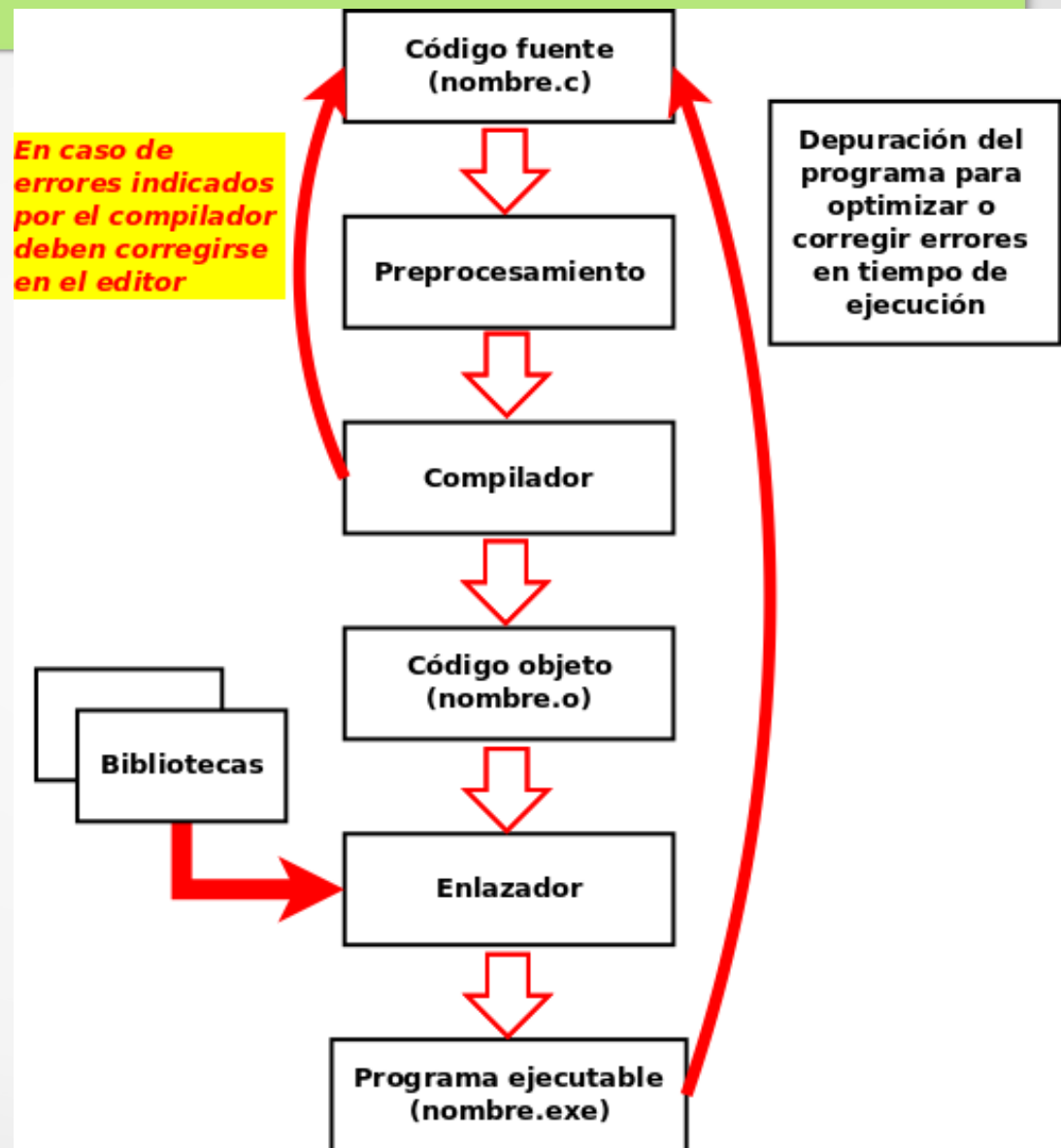
- Editor de texto plano guardando con extensión .c
- Compilador de lenguaje C como gcc (GNU Compiler Collection) o integrado en un IDE
- Una consola o terminal del Sistema Operativo para correr el archivo ejecutable

Ingredientes para programar en C

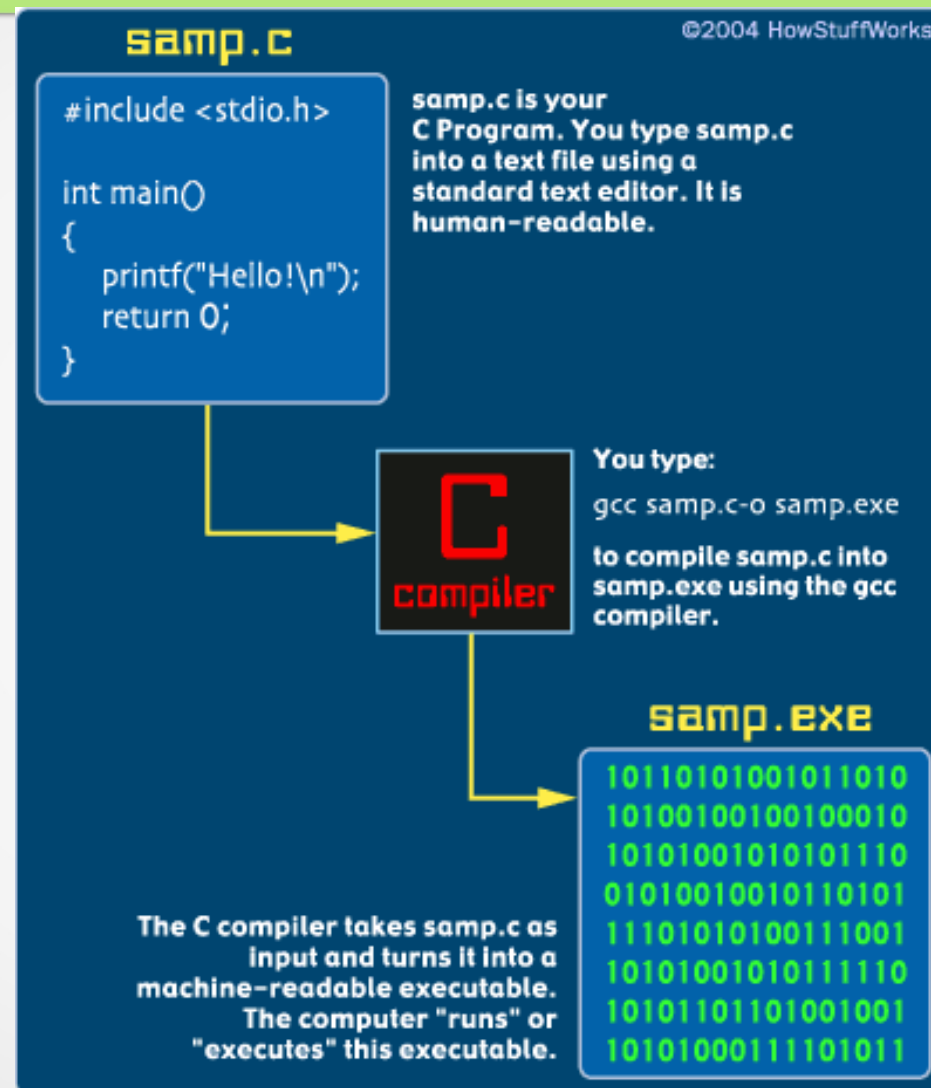
- Editores de texto plano:
 - Nativos del S.O. edit, notepad, **vi**
 - Resaltan sintáxis del lenguaje: **Gedit**, notepad++, atom, sublime, emacs
- IDE (Entorno de desarrollo integrado):
 - Turbo C, - Visual Studio
 - **Dev-C++** - **CodeBlocks**
 - **Geany** - Eclipse
 - Xcode - CppDroid

Compilación en C

- El código fuente se guarda como .c
- Es dependiente del sistema operativo y arquitectura de la máquina
- Se obtiene un ejecutable .exe compuesto de unos y ceros
- Se basa en funciones





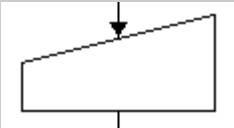
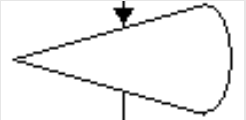


Compilación en C



Conceptos básicos

- **Variable o identificador:**
 - Letra o palabra que toma un dato inicializado, dado por el usuario o resultado de operación. `int uno, float x, int miVariable`
- **Función:**
 - Bloque de código incluido en el compilador listo para usar con solo pedirlo a la biblioteca que lo contenga. `printf()` , `scanf()`
- **Biblioteca o archivo de cabecera:**
 - Conjunto de funciones específicas ya programadas especializadas en manejo de cadenas, operaciones matemáticas, lectura del teclado e impresión en pantalla. `stdio.h, math.h, stdlib.h`

Pseudocódigo – DFD - C

ACCIÓN	DFD	LENGUAJE C
INICIO / ALGORITMO		<pre>int main(void) int main(int argc, char* argv[])</pre>
ASIGNAR / DECLARAR / DEFINIR		<pre>int x=0; float y=5.1;</pre>
LEER Y GUARDAR EN / LEER Y ALMACENAR EN		<pre>scanf("%d", &x); scanf("%f", &y); scanf("%d,%f", &x, &y);</pre>
IMPRIMIR / ESCRIBIR / MOSTRAR MENSAJE		<pre>printf("Hola Mundo"); printf("La suma es: %d", x);</pre>
CALCULAR / REALIZAR OPERACIÓN		<pre>z= x + y; suma= suma + conde;</pre>
FIN / FIN ALGORITMO		<pre>return 0;</pre>

Función main

- Se requiere para que el programa sea ejecutable.
- Al ejecutarse es la primer función que busca sin importar su ubicación y va realizando en secuencia las instrucciones o llamadas a funciones que encuentre.
- Hay diversas formas de construir la función principal o main:
 - `void main()`
 - `main()`
 - `main(void)`
 - `int main(void)`
 - `int main(int argc, char* argv[])`

“Hola Mundo”

*int es el valor
devuelto por la
función*

*main es la
función
principal*

*argc es el
contador de
argumentos*

*argv[] es el arreglo de
caracteres que guarda
los argumentos*

INICIO

1. IMPRIMIR 'Hola Mundo'

FIN

```
int main(int argc, char* argv[])
{
    printf("Hola Mundo");
    return 0;
}
```

*Termina la
ejecución
devolviendo cero*

*Para imprimir una cadena
ahora se delimita por
comillas dobles "cadena"*

Hecho por Huicho :)

Líneas de código adicionales

- Incluir biblioteca de acuerdo a la función del sistema usada

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

```
#include <math.h>
```

- Función para mantener estática la ejecución del programa al finalizar o evitar el cierre de la ventana con la consola

```
getchar() ;
```

```
getch() ;
```

```
getche() ;
```

```
System(Pause) ;
```

Mantener en espera la ejecución

Función	Biblioteca o fuente	Uso
<code>getch () ;</code>	<code>conio.h</code>	Lee un carácter del teclado pero no lo muestra en pantalla.
<code>getche () ;</code>	<code>conio.h</code>	Lee un carácter del teclado pero si lo muestra en pantalla.
<code>System ("Pause") ;</code>	Sistema operativo Ms-DOS	Recibe como argumento cualquier comando que se use en la consola del S.O. o cualquier programa que le digamos
<code>getchar () ;</code>	<code>stdio.h</code>	<p>Lee un carácter del teclado.</p> <p>NOTA: dependiendo del s.o. en ocasiones un <code>getchar()</code> no atrapa el carácter a la primera, usar mas de uno en forma continua o limpiar el buffer del teclado con <code>fflush(stdin)</code> de <code>stdio.h</code></p>

Estructura de un código fuente en lenguaje C

```
/*comentarios generales del programa
quien lo hizo*/

//declaración de bibliotecas
#include <archivo.h>

//declaración de constantes
#define pi 3.14159

//prototipos de funciones
tipo nombre_funcion(parametros);

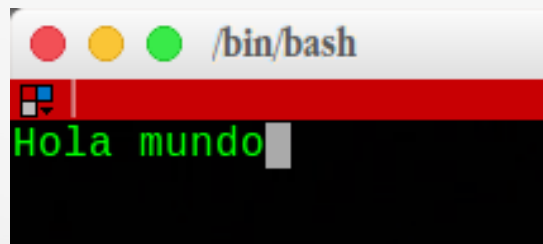
tipo nombre_variable; //variables globales

int main(int argc, char* argv[]) //función principal
{
    printf("Hola mundo"); //imprime mensaje en pantalla
    nombre_funcion(argumentos); //llamada a función
    getchar(); //pausar pantalla
    return 0; //termina ejecución
}

//declaración de funciones
tipo nombre_funcion(argumentos)
{
    //contenido de la función
}
```

Plantilla de “Hola Mundo”

```
1  /*programa hola mundo
2  | hecho por huicho*/
3
4  #include <stdio.h> //para printf y getchar
5
6  //int -> devuelve un entero
7  //argc -> contador de argumentos del programa
8  //argv -> arreglo de cadenas guarda los argumentos
9  int main(int argc, char* argv[]) //función principal
10 { //abre función principal
11     printf("Hola mundo"); //imprime msj
12     getchar(); //mantiene estática la ejecución
13     return 0; //termina ejecución devolviendo cero
14 } //cierra función principal
15
```



Notas y recomendaciones

- Sólo hay una función principal (main) en cada programa si se va a ejecutar.
- La función main puede recibir argumentos pasados desde la línea de comandos al ejecutar el programa.
- Incluir la biblioteca que contenga la función del sistema usada en el programa.
- La mayoría de las expresiones termina con " ; ".
- El programa debe ser amigable y sencillo para el usuario, no es para nosotros.
- Tratar de buscar el equivalente en lenguaje C para cada línea del diseño en diagrama de flujo o pseudocódigo.
- La palabra return indica que la función devuelve algo.
- Agregar comentarios que sirvan de guía por línea o bloque.
- Usar sentido común e inglés básico para corregir errores mostrados en la consola al compilar y no estar moliendo al profesor.



Compilación desde una terminal de Linux

- Escribir código en gedit con extensión **.c**

```
]$ gedit hola.c
```

- Compilar con gcc y nombra automáticamente **a.out** el ejecutable

```
]$ gcc hola.c
```

- Compilar con gcc y el **modificador -o** para asignar nombre al ejecutable que puede o no llevar extensión

```
]$ gcc hola.c -o hola.exe
```

- Ejecutar por nombre asignado

```
]$ ./hola.exe
```

Compilación desde una terminal de Linux

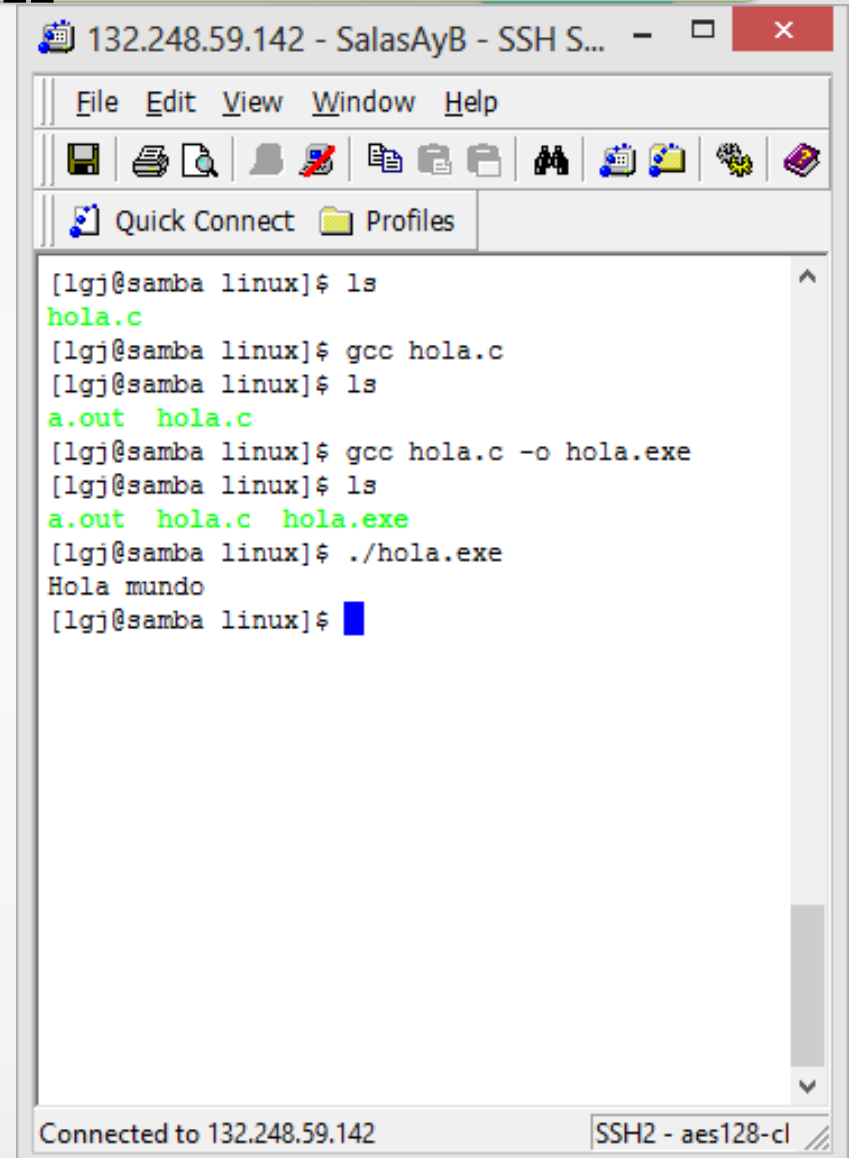
```
hola.c (~/Documentos) - gedit
Abrir ▾ [icon]
1 /*programa hola mundo
2 hecho por huicho*/
3
4 #include <stdio.h> //para printf y getchar
5
6 //int -> devuelve un entero
7 //argc -> contador de argumentos del programa
8 //argv -> arreglo de cadenas guarda los argumentos
9 int main(int argc, char* argv[]) //función principal
10 { //abre función principal
11     printf("Hola mundo"); //imprime msj
12     getchar(); //mantiene estática la ejecución
13     return 0; //termina ejecución devolviendo cero
14 } //cierra función principal
```

```
huicho@NX-01: ~/Documentos
huicho@NX-01:~/Documentos$ gedit hola.c
huicho@NX-01:~/Documentos$ gcc hola.c
huicho@NX-01:~/Documentos$ ls
a.out  hola.c
huicho@NX-01:~/Documentos$ gcc hola.c -o hola.exe
huicho@NX-01:~/Documentos$ ls
a.out  hola.c  hola.exe
huicho@NX-01:~/Documentos$ ./hola.exe
Hola mundo
huicho@NX-01:~/Documentos$
```

Hecho por Huicho :)

Compilación desde una terminal de Linux de SSH Secure Shell

- Escribir código en gedit con extensión .c
-]\$ **gedit hola.c**
- Compilar con gcc y nombra automáticamente a.out el ejecutable
-]\$ **gcc hola.c**
- Compilar con gcc y el modificador -o para asignar nombre al ejecutable que puede o no llevar extensión
-]\$ **gcc hola.c -o hola.exe**
- Ejecutar por nombre asignado
-]\$ **./hola.exe**



The screenshot shows an SSH terminal window titled "132.248.59.142 - SalasAyB - SSH S...". The terminal displays the following commands and output:

```
[lgj@samba linux]$ ls
hola.c
[lgj@samba linux]$ gcc hola.c
[lgj@samba linux]$ ls
a.out hola.c
[lgj@samba linux]$ gcc hola.c -o hola.exe
[lgj@samba linux]$ ls
a.out hola.c hola.exe
[lgj@samba linux]$ ./hola.exe
Hola mundo
[lgj@samba linux]$
```

The terminal window includes a menu bar (File, Edit, View, Window, Help) and a toolbar with various icons. The status bar at the bottom indicates "Connected to 132.248.59.142" and "SSH2 - aes128-cl".

Compilación en DEV-C++

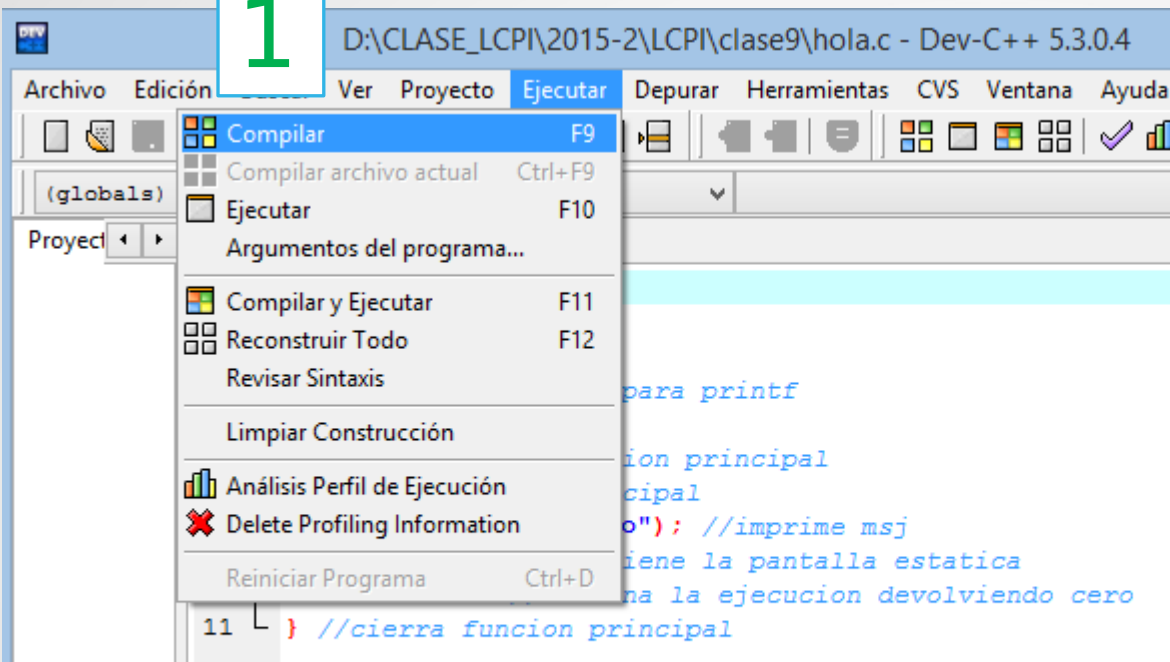
- Plataforma: **Windows**
- Liga de **descarga del instalador**:

<http://sourceforge.net/projects/orwellddevcpp/files/Setup%20Releases/Dev-Cpp%205.3.0.4%20MinGW%204.7.0%20Setup.exe/download>

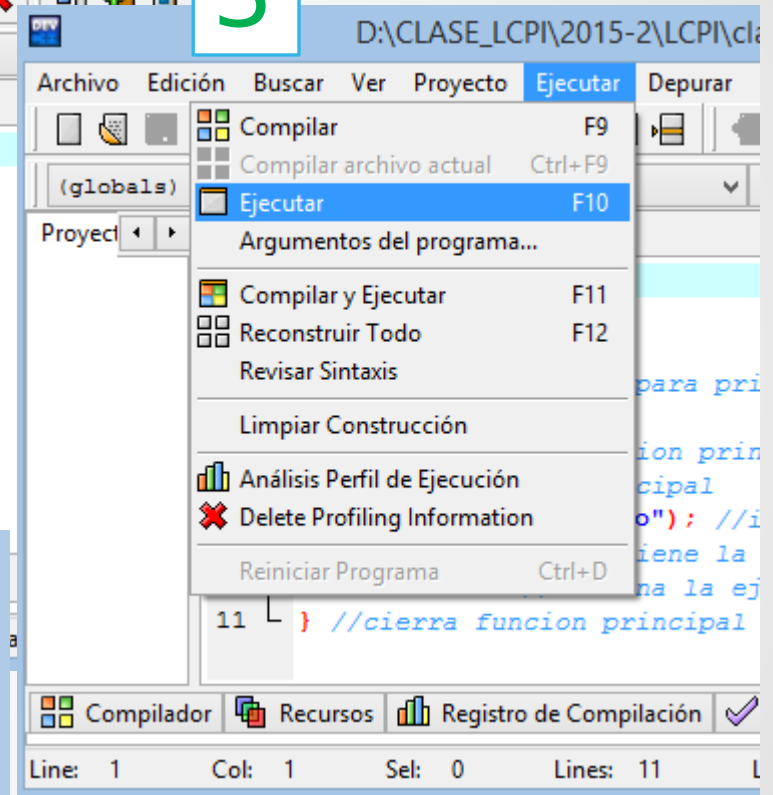
- Escribir código fuente en el editor integrado del IDE
- Menú **Archivo** → **Guardar Como...** Guardar con **extensión .c**
- Menú **Ejecutar** → **Compilar**
- Menú **Ejecutar** → **Ejecutar**

Compilación en DEV-C++

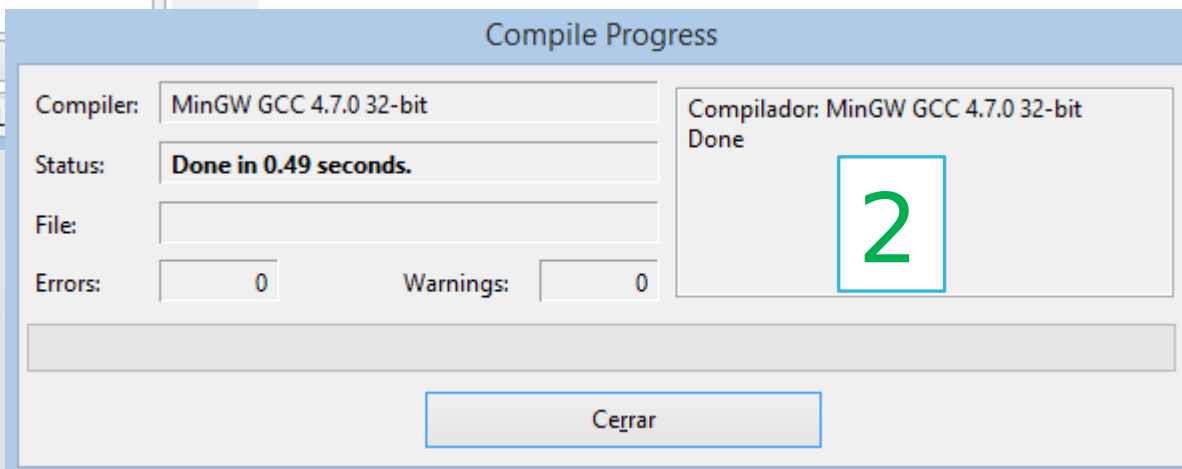
1



3



2

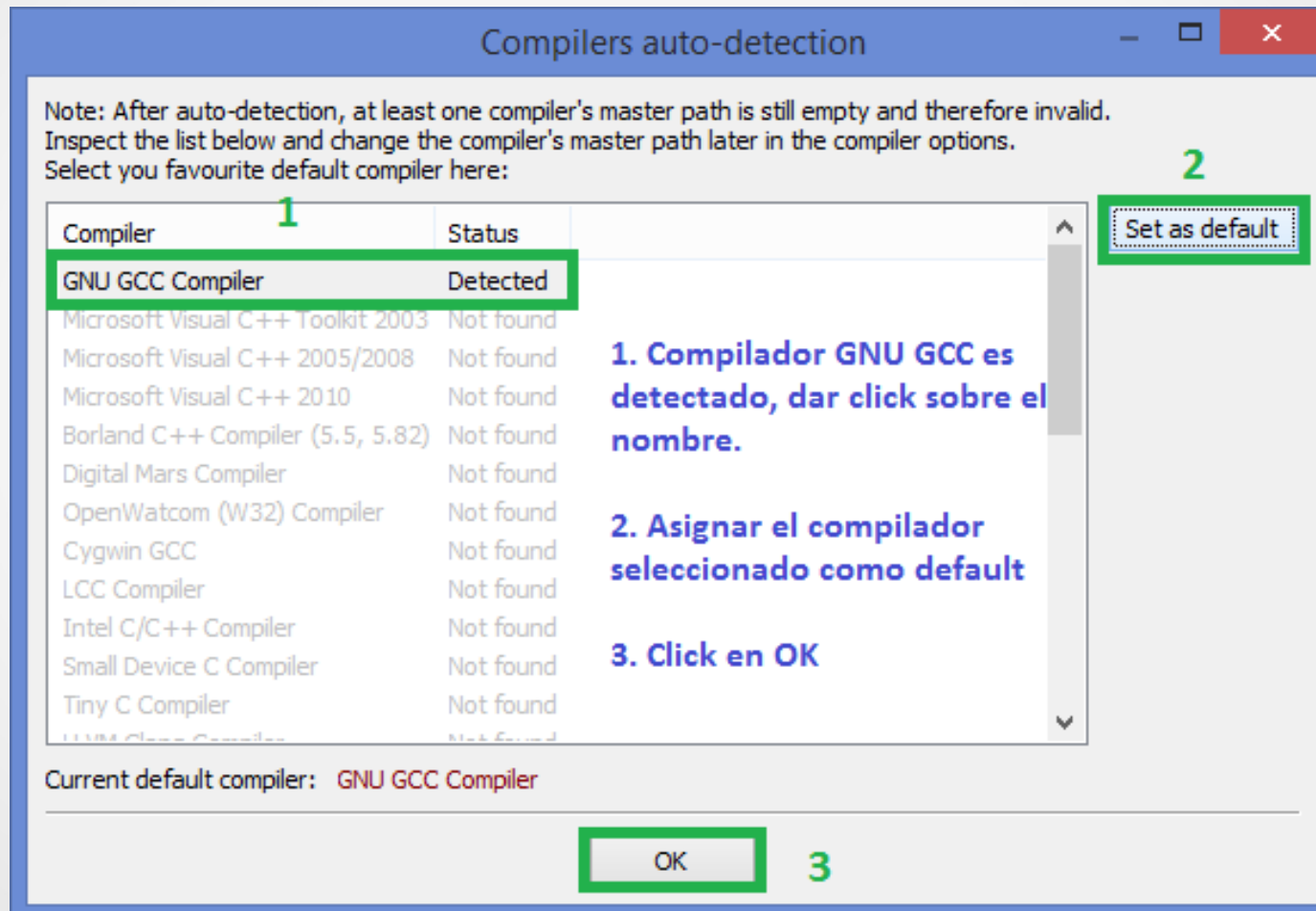


Hecho por Huicho :)

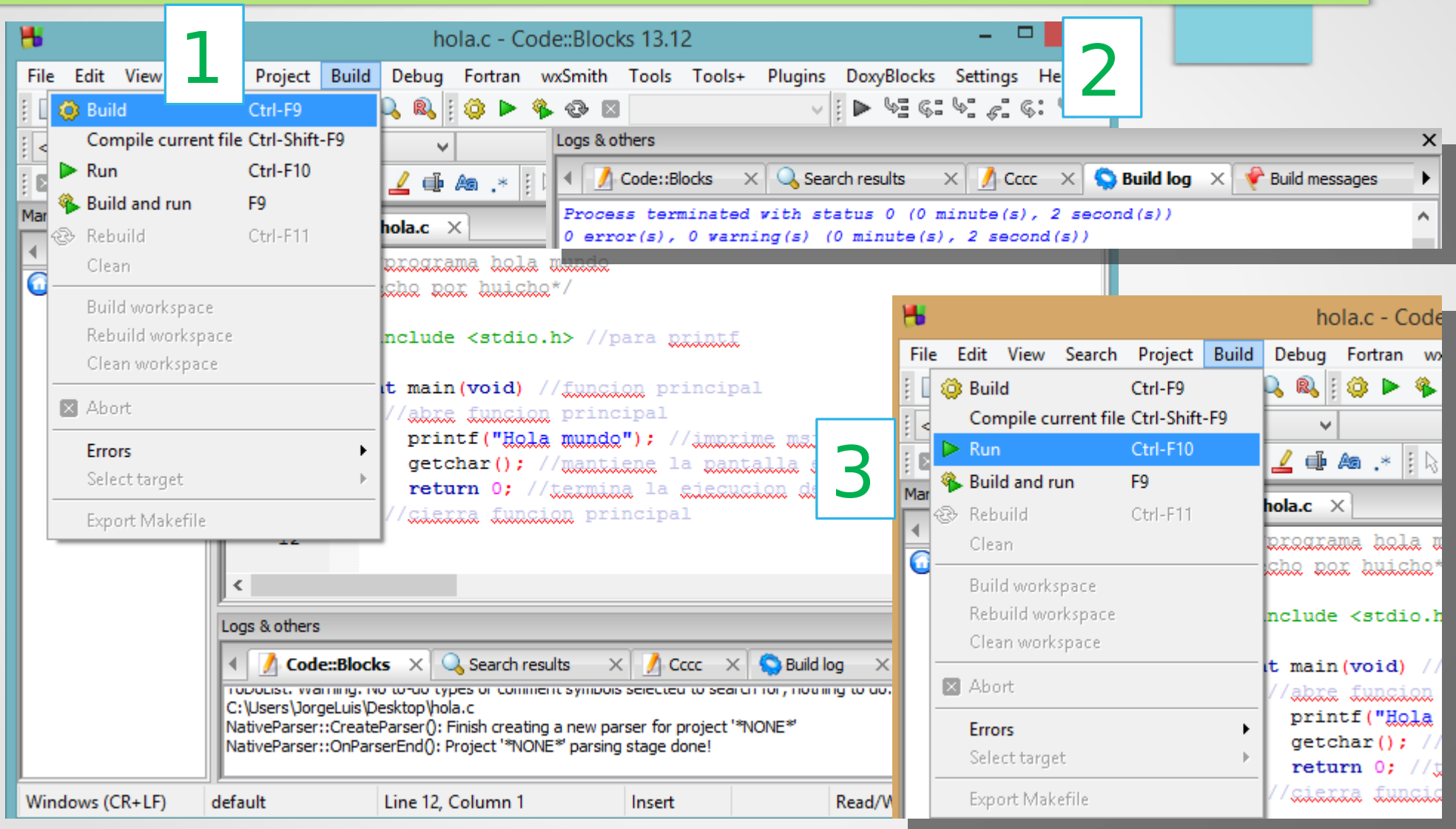
Compilación en Code::Blocks

- Plataforma: **Windows, Linux y Mac OS**
- Liga de **descarga**:
<http://www.codeblocks.org/downloads/26>
- Usuarios de Windows sin gcc ni MinGW descargar:
codeblocks-17.12mingw-setup.exe
- Después de instalar seleccionar el compilador GCC como default. (*Ver siguiente diapositiva*)
- Escribir código fuente en el editor integrado del IDE
- Menú **File** → **Save file as...** Guardar con **extensión .c**
- Menú **Build** → **Build**
- Menú **Build** → **Run**

Compilación en Code::Blocks

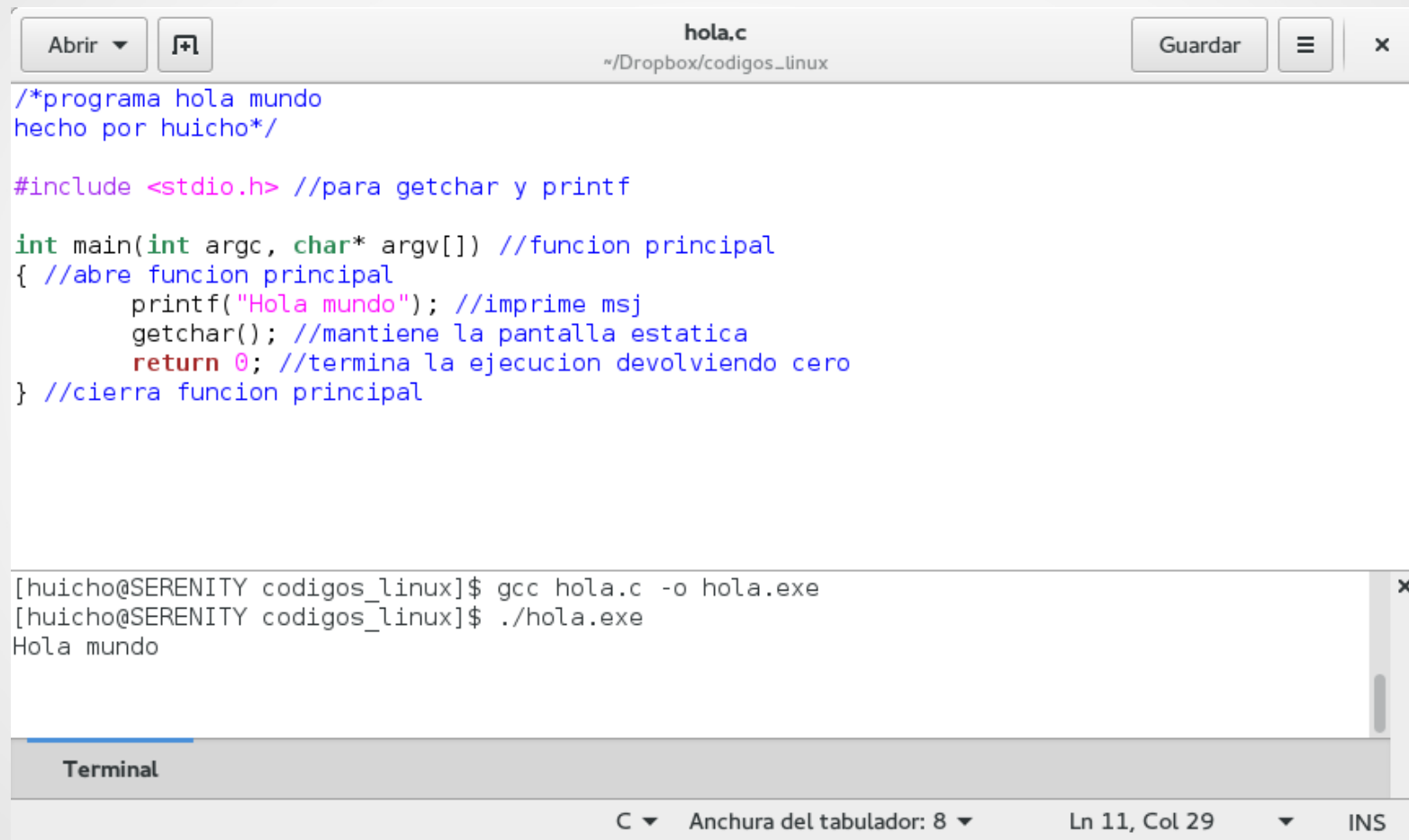


Compilación en Code::Blocks



Hecho por Huicho :)

Compilación desde una terminal empotrada en gedit en Linux



The screenshot shows the gedit text editor with a window titled 'hola.c' at the path '~/Dropbox/codigos_linux'. The editor contains a C program that prints 'Hola mundo'. Below the code, an embedded terminal window shows the compilation and execution of the program. The terminal output is 'Hola mundo'. The status bar at the bottom indicates 'C', 'Anchura del tabulador: 8', 'Ln 11, Col 29', and 'INS'.

```
hola.c
~/Dropbox/codigos_linux

/*programa hola mundo
hecho por huicho*/

#include <stdio.h> //para getchar y printf

int main(int argc, char* argv[]) //funcion principal
{ //abre funcion principal
    printf("Hola mundo"); //imprime msj
    getchar(); //mantiene la pantalla estatica
    return 0; //termina la ejecucion devolviendo cero
} //cierra funcion principal

[huicho@SERENITY codigos_linux]$ gcc hola.c -o hola.exe
[huicho@SERENITY codigos_linux]$ ./hola.exe
Hola mundo

Terminal

C  Anchura del tabulador: 8  Ln 11, Col 29  INS
```