Universidad Nacional
de Lomas de Zamora

Versión 1.03 – 15-Abril-2021

Cátedra de Fundamentos de programación Informática

Objetivo: Aprender a programar comprendiendo cómo funciona una computadora, desarrollando la capacidad de utilizar código desarrollado por otros.

Universidad Nacional de Lomas de Zamora

Tipos de Lenguaje

Lenguaje Máquina

El lenguaje máquina es el único que entiende la computadora, es su "lenguaje natural". En él sólo se pueden utilizar dos símbolos: el cero (0) y el uno (1) (Lógica Binaria). Estos lenguajes son dependientes del procesador, es decir no funcionarán en una pc con cpu distinto, es por esto que se necesitará una “traducción” de todas las instrucciones para lograr que funcione en otro equipo.

Lenguaje de Bajo Nivel

Los lenguaje de bajo nivel, también llamados lenguajes ensambladores, permiten al programador escribir instrucciones de un programa usando abreviaturas del inglés, también llamadas palabras nemotécnicas, tales como: ADD, DIV, SUB, etc. Un programa escrito en un lenguaje ensamblador tiene el inconveniente de que no es comprensible para la computadora, ya que, no está compuesto por ceros y unos. Para traducir las instrucciones de un programa escrito en un lenguaje ensamblador a instrucciones de un lenguaje máquina hay que utilizar un programa llamado ensamblador.

Lenguaje de Medio Nivel

Suelen ser clasificados muchas veces de alto nivel, pero permiten ciertos manejos de bajo nivel. Son precisos para ciertas aplicaciones como la creación de sistemas operativos, ya que permiten un manejo abstracto (independiente de la máquina, a diferencia delensamblador), pero sin perder mucho del poder y eficiencia que tienen los lenguajes de bajo nivel.

- C
- Basic

Lenguaje de Alto Nivel

El lenguaje de alto nivel (high-level language) es aquel que se aproxima más al lenguaje natural humano que al lenguaje binario de las computadoras, el que se conoce como lenguaje de bajo nivel. Existe la posibilidad de que se pueda utilizar el mismo programa en distintas máquinas, es decir que es independiente de un hardware determinado, pero suelen ser más lentos que los lenguajes de medio y bajo nivel.

- C++
- Lisp
- Fortran
- Cobol

Universidad Nacional de Lomas de Zamora

Compilación vs interpretación

Compilacion

Convertir de lenguaje fuente a lenguaje de máquina, generando así un programa o archivo ejecutable. Es decir la conversión se realiza toda completa de una vez.

- C
- C++
- Swift
- Rust

Interpretacion

Traduce de a partes, típicamente de a una instrucción del lenguaje fuente. Se traduce, se ejecuta lo traducido y se repite. La velocidad de ejecución es una de las principales desventajas de esta estrategia.

- Python
- Lisp
- PHP
- VBScript

Compilador

Código fuente , escrito por el programador. En el cual se describen las secuencias lógicas del programa. Contiene Comentarios, y directivas del preprocesador siempre precedidas por el simbolo #.

Toma como entrada los archivos fuentes que componen el programa y se encarga de eliminar comentarios, e interpretar y procesar directivas del preprocesamiento:
#include: Sustituye la línea por el contenido del fichero especificado.
#define: Define una constante (identificador) simbólico.

Analiza la sintaxis y la semántica del código fuente preprocesado y lo traduce, generando un fichero que contiene el código objeto.
En el proceso de compilación se interpreta el código fuente salvo las referencias a objetos externos (funciones o variables) en el módulo de código que se está compilando.

Toma los archivos objetos y los combina en una imagen (archivo) ejecutable.
Completa las llamadas de un objeto a otro, reemplazando las “anotaciones” dejadas por el compilador con la dirección o desplazamiento real para acceder al código requerido.

Universidad Nacional de Lomas de Zamora

Historia y estándares

DENNIS RITCHIE

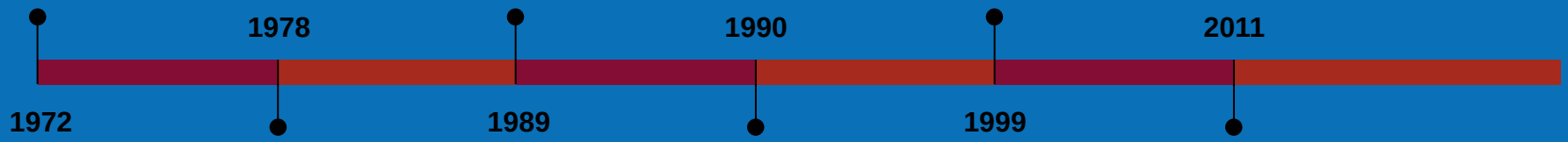
El desarrollo inicial de C se llevó a cabo en los Laboratorios Bell de AT&T entre 1969 y 1973

ANSI C

En 1983, el Instituto Nacional Estadounidense de Estándares (ANSI) organizó un comité, X3j11, para establecer una especificación estándar de C.

C99

Tras el proceso de estandarización de ANSI, la especificación del lenguaje C permaneció relativamente estable durante algún tiempo, mientras que C++ siguió evolucionando.



kERNIGHAN & RITCHIE

Ritchie y Brian Kernighan publicaron la primera edición de El lenguaje de programación C, también conocido como La biblia de C

ANSI/ISO C

En 1990, el estándar ANSI (con algunas modificaciones menores) fue adoptado por la Organización Internacional para la Estandarización (ISO) en el estándar ISO/IEC 9899:1990.

C11

C11 (antes conocido como C1X) es un nombre informal para ISO/IEC 9899:20118. El borrador final, N1570, fue publicado en abril de 2011.

Universidad Nacional
de Lomas de Zamora

Entorno de desarrollo (IDE)

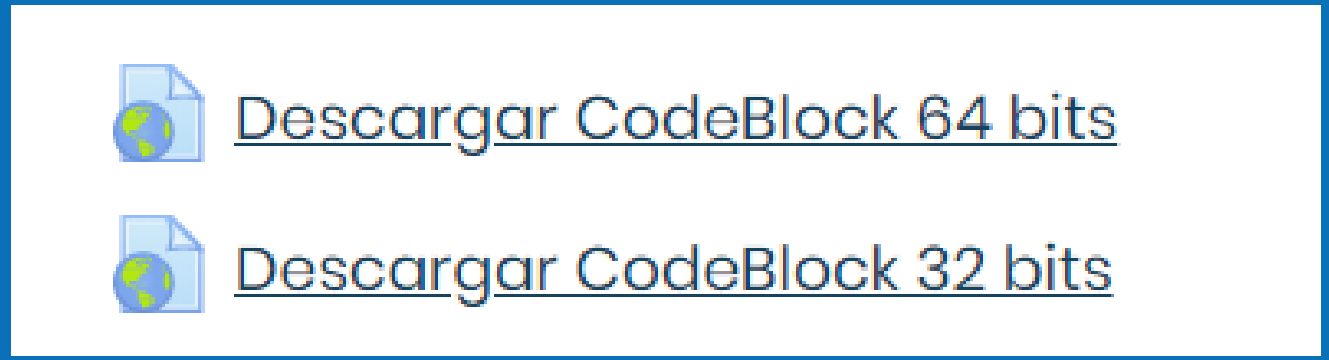
Para poder programar en C necesitamos un IDE (Entorno de desarrollo) que venga con su respectivo compilador.

El IDE que utilizaremos en la materia es CODE:BLOCKS, con el compilador MINGW.

Universidad Nacional de Lomas de Zamora

Instalación y configuración

Para realizar la instalación buscaremos los bits de nuestro sistema operativo (32 o 64).
Teniendo ese dato vamos al aula virtual de Moodle y hacemos clic en el link correspondiente:



Universidad Nacional
de Lomas de Zamora

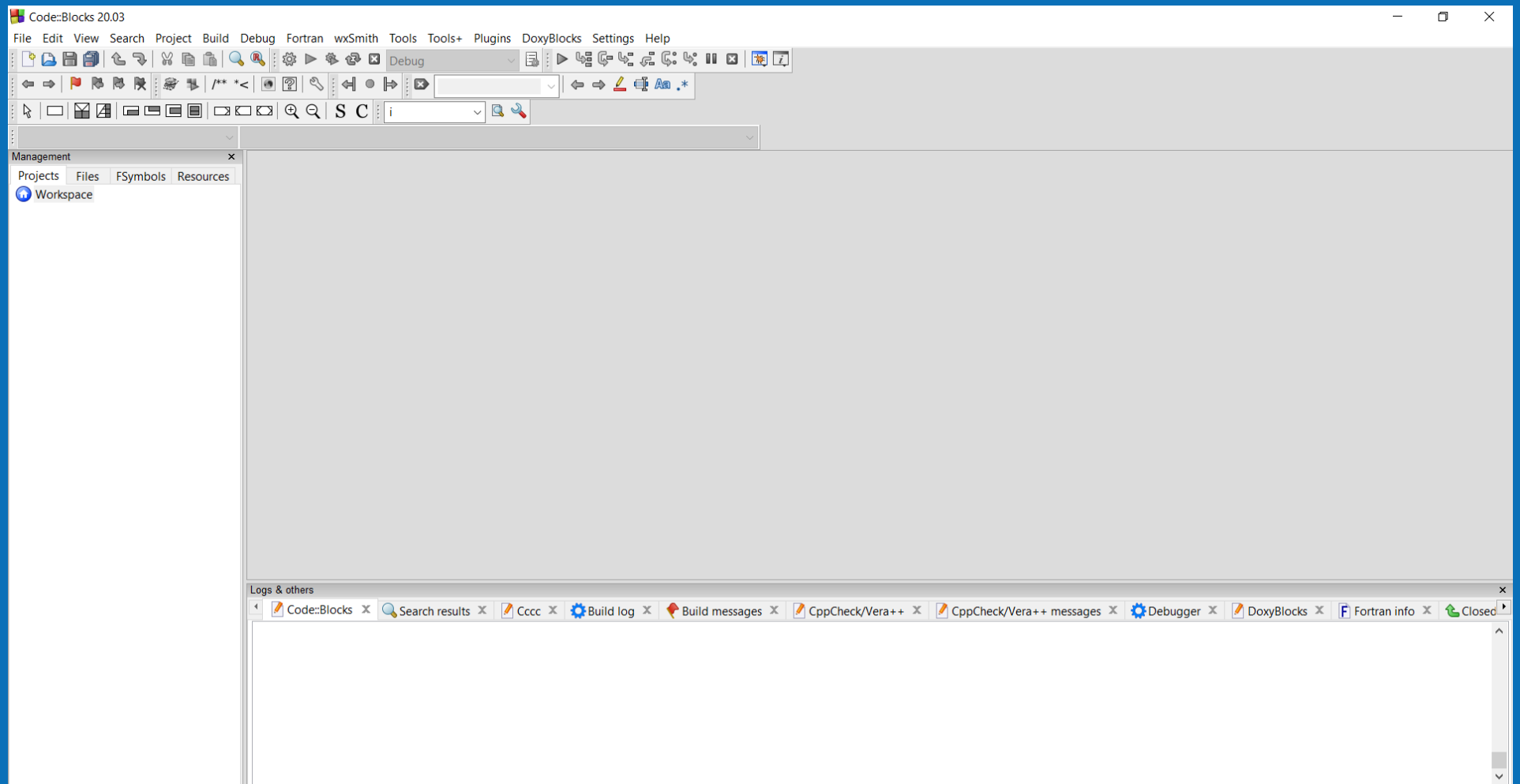
Instalación y configuración

Una vez descargado, le hacemos doble clic al archivo y lo instalamos dando “siguiente – siguiente – siguiente” hasta que esté el programa instalado.

Universidad Nacional de Lomas de Zamora

Instalación y configuración

Una vez instalado abrimos el programa.

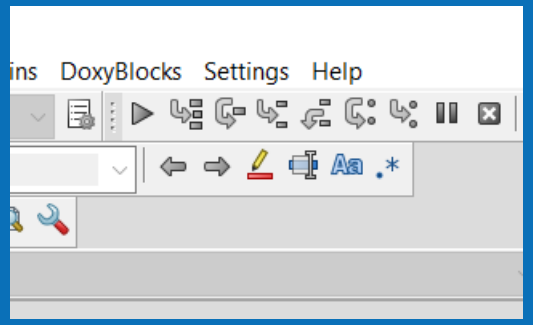


Universidad Nacional de Lomas de Zamora

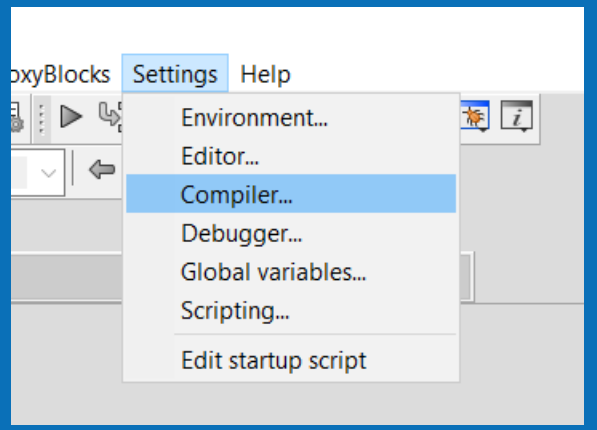
Instalación y configuración

Ahora procedemos a hacer el seteo inicial, hacemos clic en settings (arriba a la derecha), luego compiler.

1



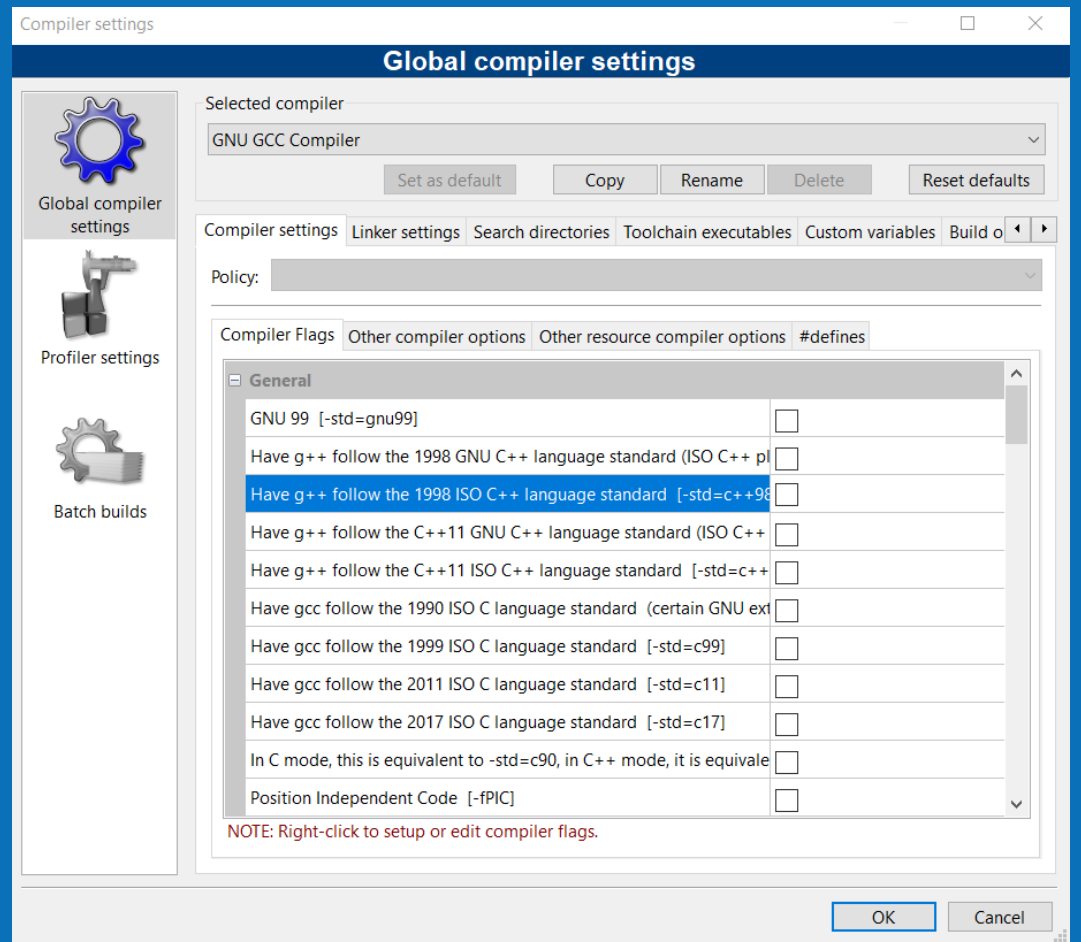
2



Universidad Nacional de Lomas de Zamora

Instalación y configuración

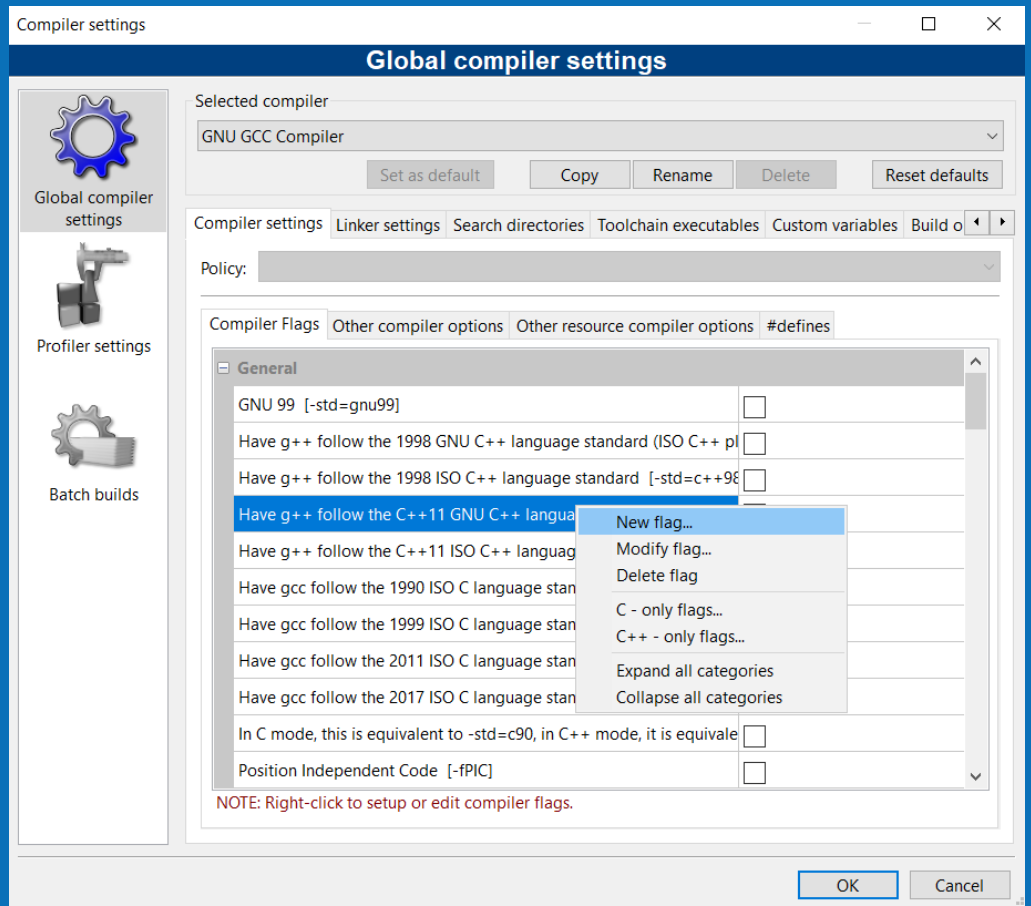
Aparecerá la ventana siguiente:



Universidad Nacional de Lomas de Zamora

Instalación y configuración

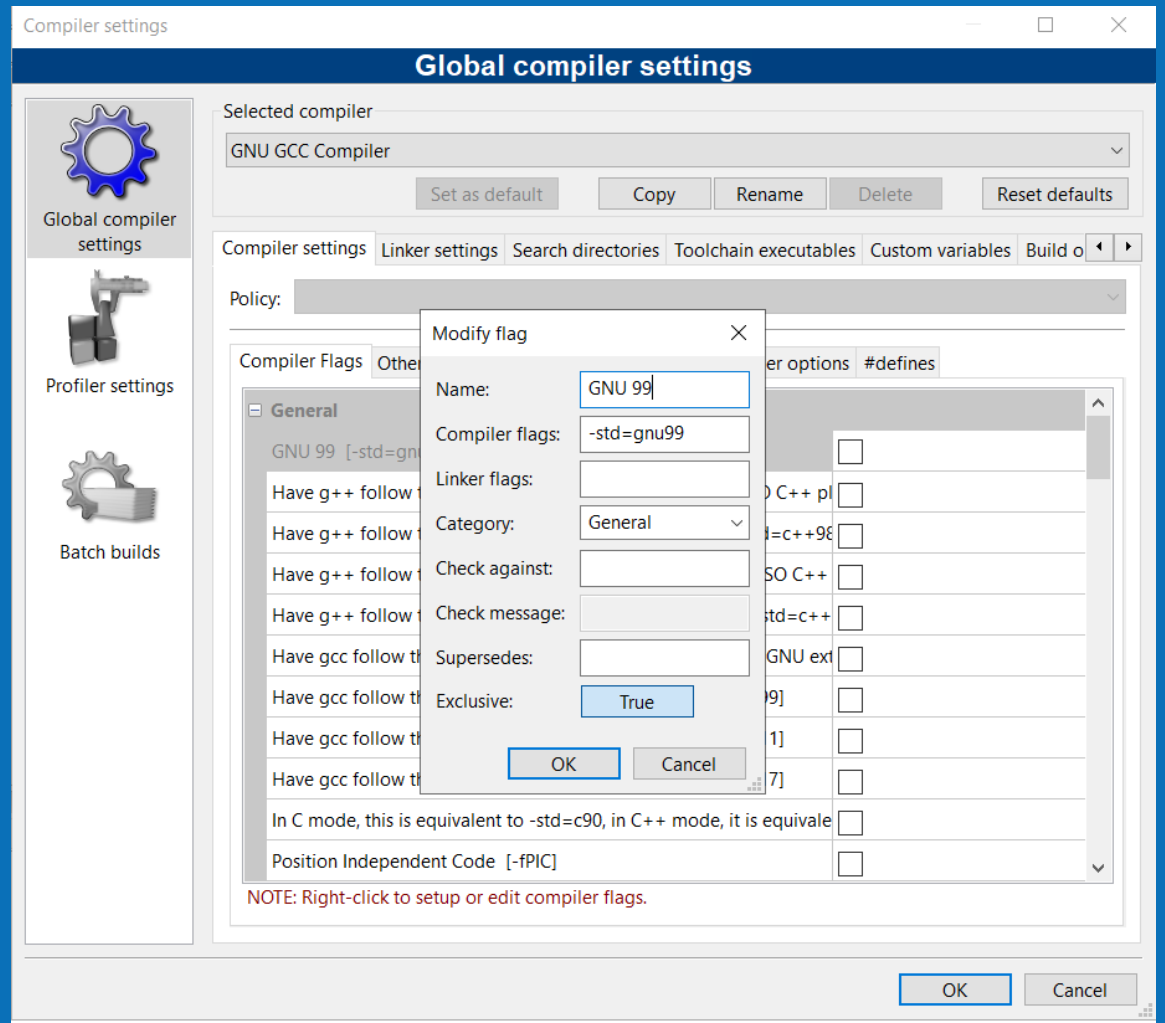
Hacemos botón derecho en alguno de estos elementos que dicen “Have…..” y hacemos clic en “New flag…”:



Universidad Nacional de Lomas de Zamora

Instalación y configuración

En la ventana que aparece llenaremos los datos como en la imagen:



En el cuadro “Name:” Pondremos:

GNU 99

En el cuadro “Compiler flags:” Pondremos:

-std=gnu99

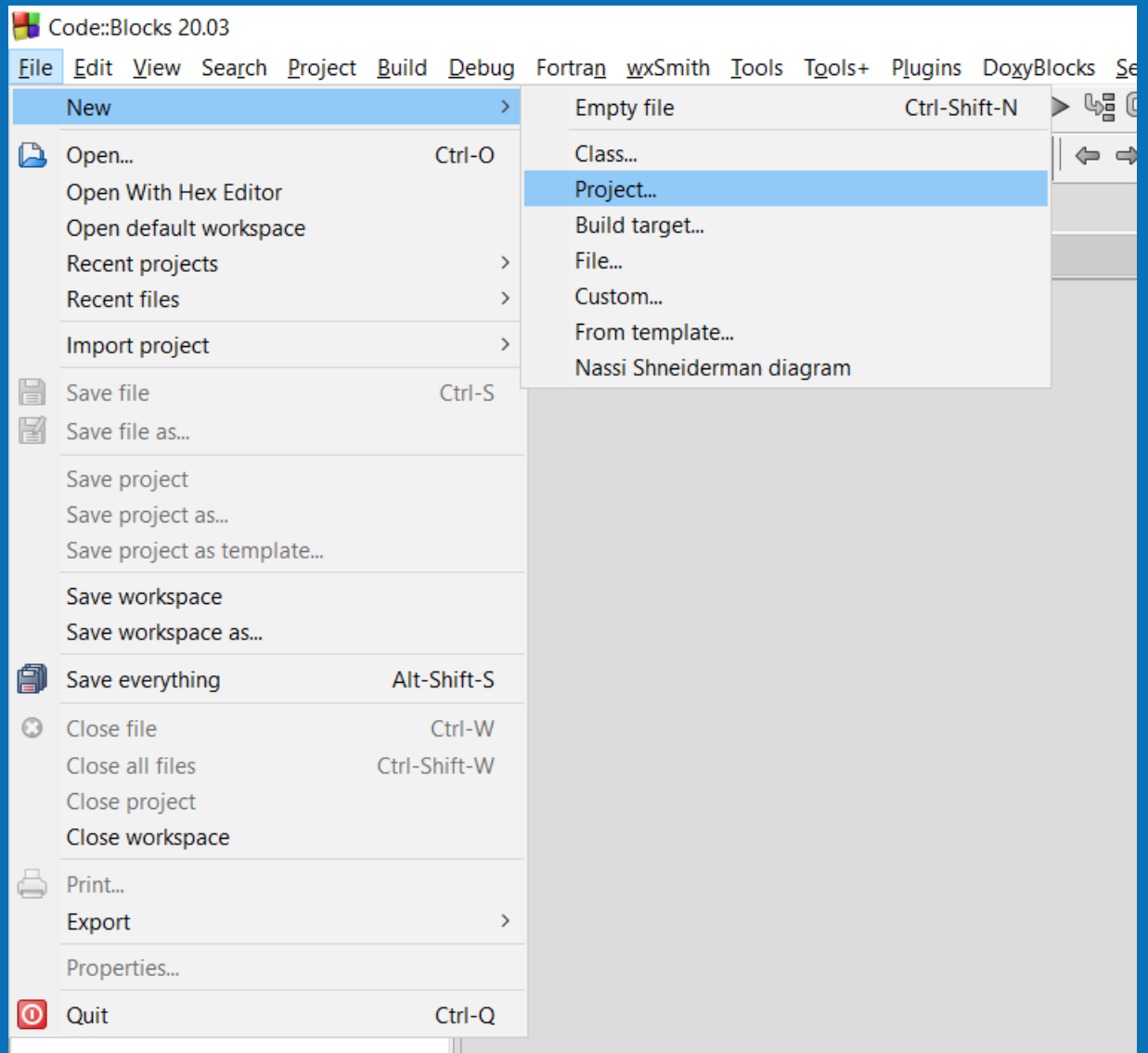
Y luego haremos clic en el botón “OK” de ambas ventanas

Instalación y configuración

Listo!

Universidad Nacional de Lomas de Zamora

Creación de un programa:



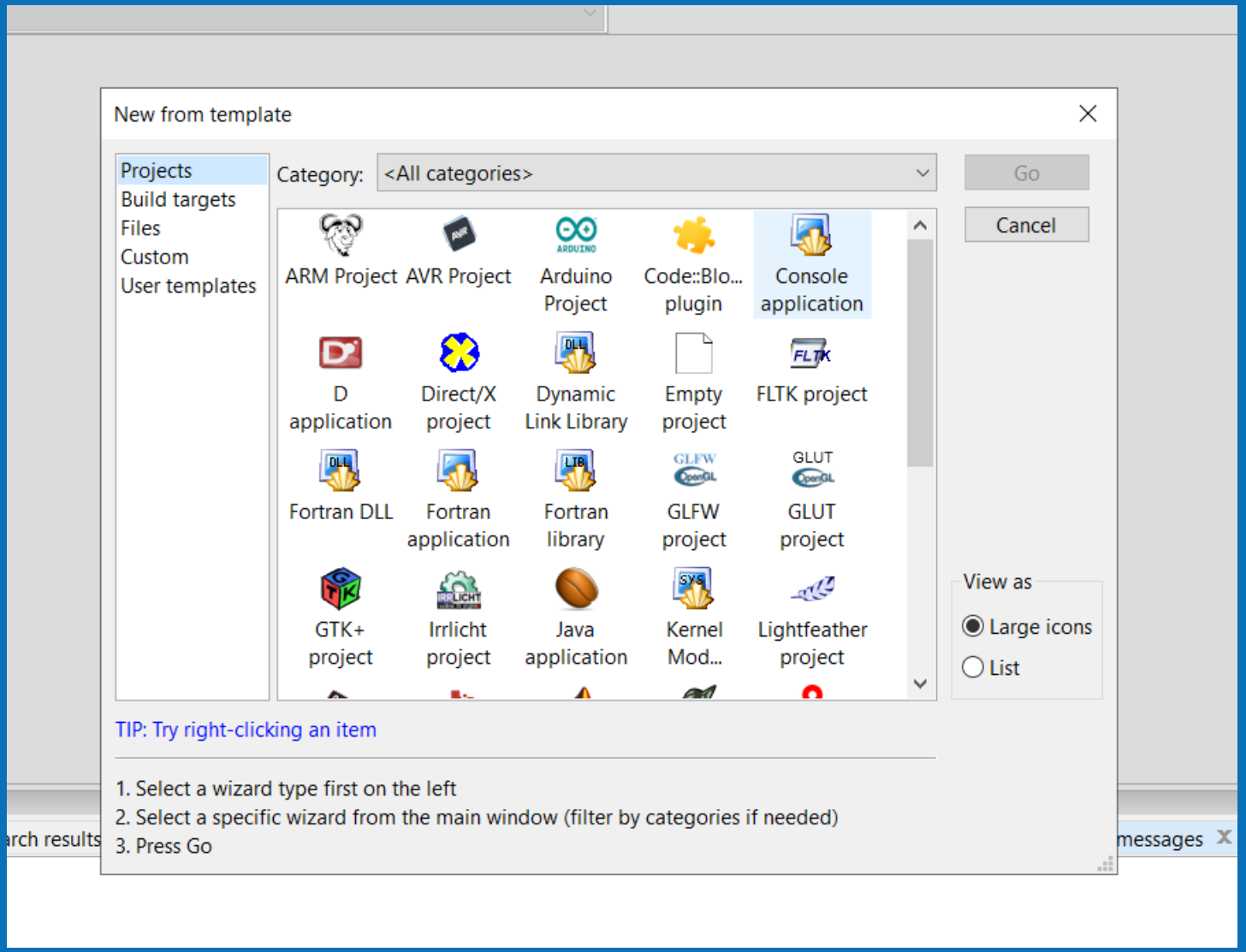
Para crear un nuevo programa hacemos clic en las pestañas:

File -> New -> Project…

Universidad Nacional de Lomas de Zamora

Creación de un programa:

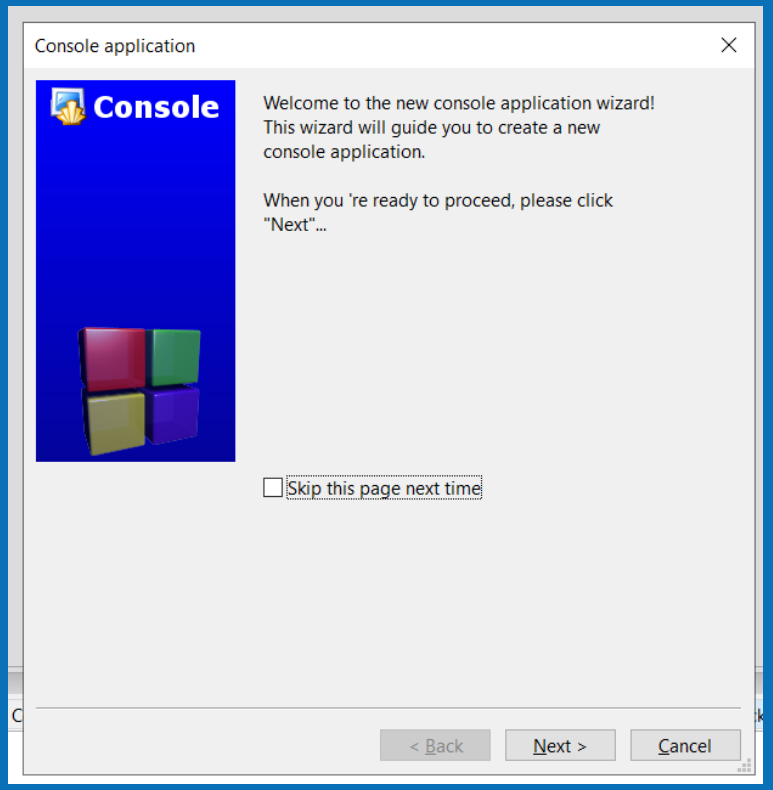
Luego clic en “Console application” y luego en “Go”



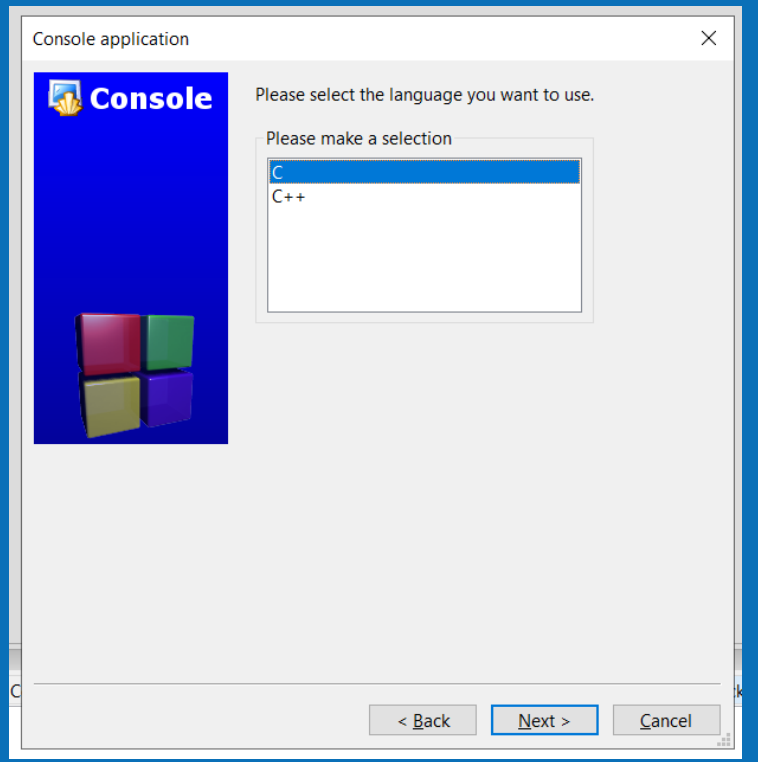
Universidad Nacional de Lomas de Zamora

Creación de un programa:

Clic en “Next”

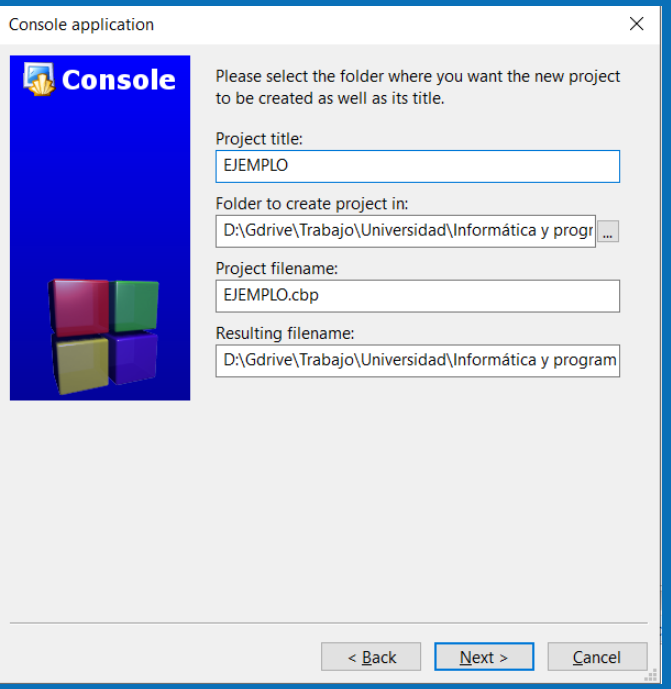


Clic en C y luego “Next”

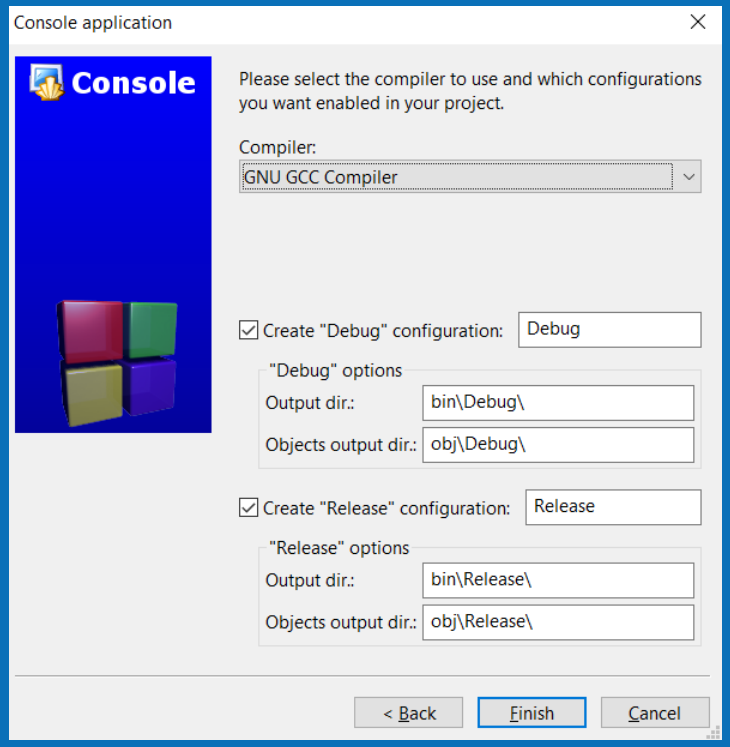


Universidad Nacional de Lomas de Zamora

Creación de un programa:



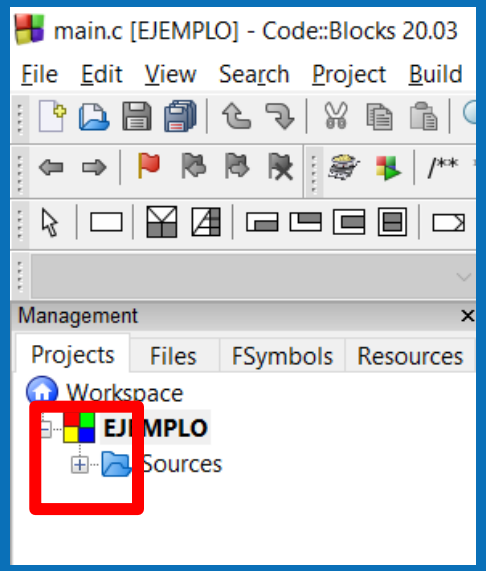
En Project title: le ponemos un nombre al programa y nos aseguramos de que el segundo cuadro NO esté en la carpeta “Program files” o “Archivos de programa” y luego clic en “Next”



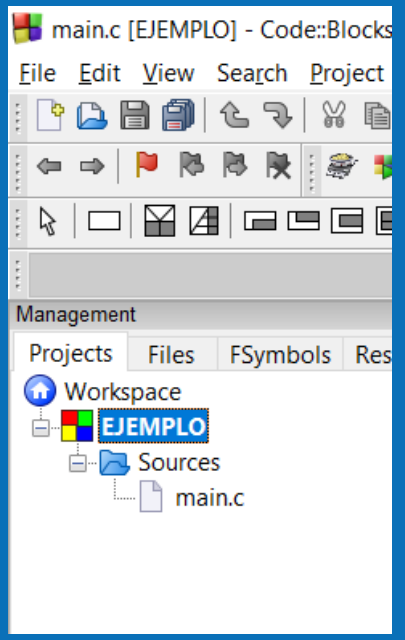
Nos aseguramos de que estén ambas opciones con clic “Debug” y “Releas” y luego clic en “Finish”

Universidad Nacional de Lomas de Zamora

Creación de un programa:



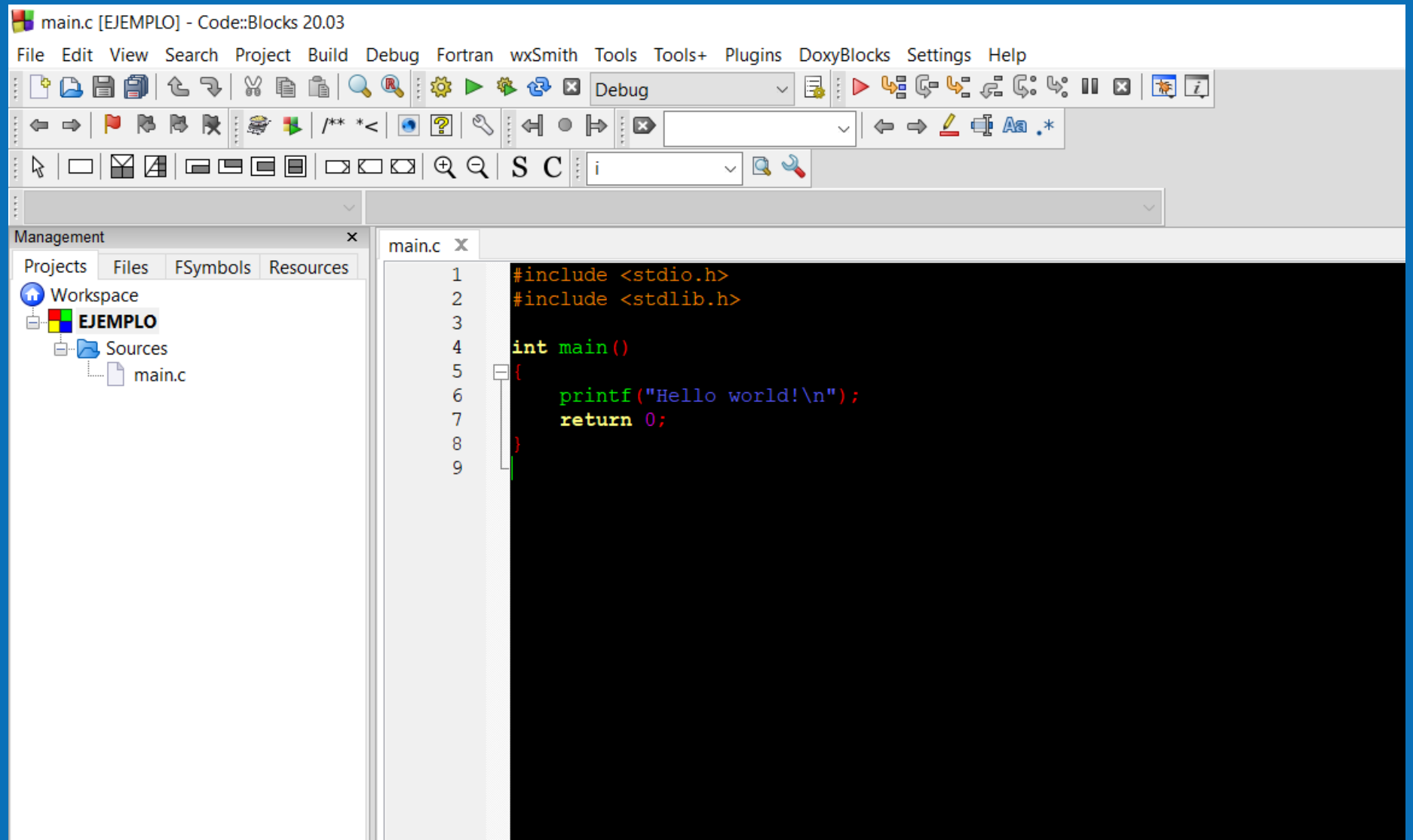
Miramos a la izquierda y debajo de nuestro proyecto vemos una carpeta llamada “Sources” junto a un símbolo de “ + “, le hacemos clic a ese símbolo.



Aparece un archivo llamado main.c
Le hacemos clic a ese archivo y nos abre ahora el editor con el programa “hello world”

Universidad Nacional de Lomas de Zamora

Creación de un programa:



Creación de un programa

Listo!

Universidad Nacional de Lomas de Zamora

Lógica Decimal y Binaria

Las computadoras funcionan con lógica binaria (2 estados 0 y 1) y para crear distintos valores nos valemos de la posición de ese número.

Universidad Nacional de Lomas de Zamora

Codificación Decimal

$17.350 =$

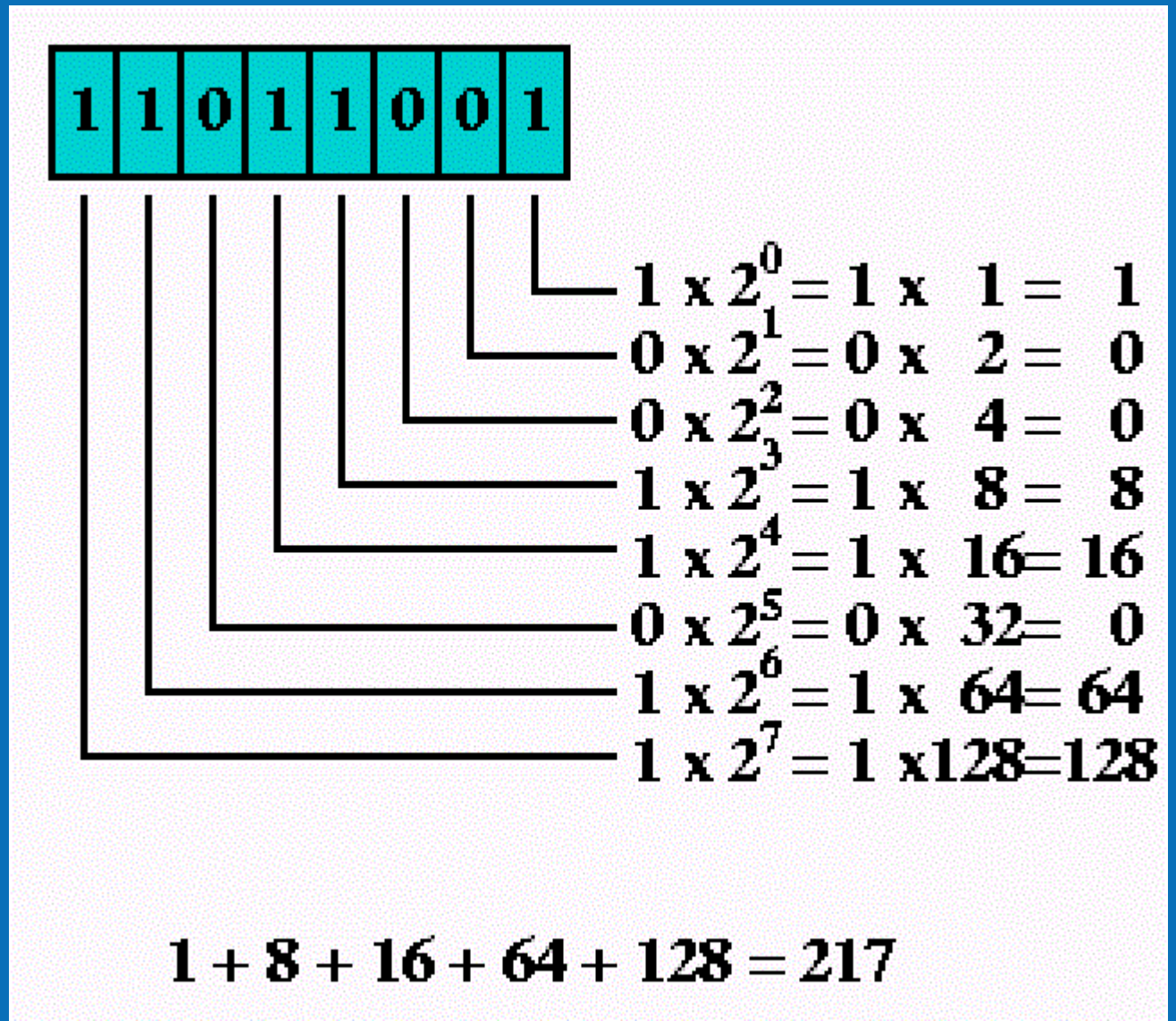
10000	1000	100	10	1
1	7	3	5	0

Luego multiplicas cada unidad por el valor posicional correspondiente y al sumarlo da el número decimal.

$$
\begin{array}{rcrcr}
10.000 & \times & 1 & = & 10.000 \\
1.000 & \times & 7 & = & 7.000 \\
100 & \times & 3 & = & 300 \\
10 & \times & 5 & = & 50 \\
1 & \times & 0 & = & 0 \\
\hline
 & & & & 17.350
\end{array}
$$

Universidad Nacional de Lomas de Zamora

Codificación Binaria



Cada posición determina la potencia de 2 a la que se multiplica el número binario.
El total de las sumas es el número codificado.

Universidad Nacional
de Lomas de Zamora

Bit y Byte

Al carácter que toma valores 0 o 1, se lo llama bit, el cual es la abreviación de Binary Digit (digito binario).
Un bit es la menor unidad de información de una computadora. Un bit tiene solamente un valor (que puede ser 0 o 1). 8 bits combinados entre sí dan origen a un byte, que es la unidad de información más utilizada junto con sus modificadores, kilobyte, megabyte, gigabyte, terabyte, etc.

Estructura básica de un programa en C

```c
#include <stdio.h>
/* Esto es un comentario
   en varias líneas*/

int main()
{
        printf("Hello world!\n");
        // otro comentario: printf es una función (de stdio)
        return 0;
}
```

Estructura general (simplificada) de un programa C:

- Directivas del preprocesador
- Definiciones de tipos de datos
- Implementación de funciones

Universidad Nacional
de Lomas de Zamora

Variables

En programación tenemos variables, que son espacios de memoria donde se guardan datos, podemos pensarlos como cajas con un nombre donde guardamos números, letras, palabras, direcciones de memoria, etc.

Universidad Nacional de Lomas de Zamora

Tipos de datos y tamaño

Existen diferentes tipos de datos en C, las variables, punteros, funciones, etc, pueden tomar esos tipos, para guardar por ejemplo variables de números naturales, variables de letras, variables de números con coma, etc.

Cada una de estas variables tiene y ocupa diferentes tamaños en la memoria del procesador.

Universidad Nacional de Lomas de Zamora

Tipos de datos y tamaño

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned int	4 bytes	0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	8 bytes	-9223372036854775808 to 9223372036854775807
unsigned long	8 bytes	0 to 18446744073709551615

Tipos de datos y tamaño

- char : Caracteres (en realidad números que representan caracteres)
- int : Números enteros, de distintos tamaños (short, long, long long)
- float : Punto flotante de 32 bits
- double : Punto flotante de 64 bits
- long double : Punto flotante de mayor precisión 80 a 128 bits
- bool: agregado recién en C99 es en realidad 0 o 1

Universidad Nacional
de Lomas de Zamora

Declaración de variables

La declaración de variables en C tiene el siguiente formato:

TIPO_DE_DATO NOMBRE_VARIABLE;

char c;

unsigned long cantidad;

int i;

Universidad Nacional
de Lomas de Zamora

Declaración de variables

También se puede declarar la variable y asignarle un valor a la vez:

TIPO_DE_DATO NOMBRE_VARIABLE = VALOR_INICIAL;

```
char letra = 'A';
unsigned long total = 0xA01D7F; // 10493311UL
int dato = 2;
bool flag = true;
```

Universidad Nacional
de Lomas de Zamora

Constantes

Constantes, toman valores como variables pero NO se pueden modificar:

#define PI 3.14159

Sería igual a
float PI = 3.14159
Con la diferencia que si usamos DEFINE, es una constante y NO se puede modificar.

Universidad Nacional
de Lomas de Zamora

Literales

Literales, son valores interpretados como de un tipo específico, ejemplo el 3 es un numero entero y se interpreta como int:

3 //entero
true, false //booleanos
'a' //caracter
“Hola” //String → en realidad “arreglo (vector) de char”
3.5 // punto flotante
'\n' // line feed (10 o 0xA)

Universidad Nacional de Lomas de Zamora

Secuencias de escape

Son secuencias especiales que nos permiten escribir caracteres especiales como “enter”, “return”, “tab”, etc.

Código	Significado	Valor ASCII (Decimal)	Valor ASCII (Hexadecimal)
'\n'	Nueva línea (dependiente SO)	10	0x0A
'\r'	Retorno de carro	13	0x0D
'\t'	Tabulador (horizontal)	09	0x09
'\f'	Nueva página	12	0x0C
'\a'	Alerta (campana)	07	0x07
'\b'	Retroceder un caracter	08	0x08
'\v'	Tabulador (vertical)	11	0xB
'\\'	Barra invertida	92	0x5C
'\''	Comilla simple	39	0x27
'\"'	Comilla doble	34	0x22
'\ddd'	El caracter ASCII cuyo código sea ddd en octal		
'\xhh'	El caracter ASCII cuyo código sea nn en hexadecimal		
Nota: Este listado no es completo			

Operadores

Universidad Nacional de Lomas de Zamora

Para nuestra lógica de programación necesitaremos hacer sumas, restas, comparaciones, asignaciones, etc.

Por esto nos valdremos de los siguientes operadores:

- Aritméticos
  - Suma, resta → + , -
  - Multiplicación, división → * , /
  - Módulo → %
- Relacionales
  - Menor, mayor → < , >
  - Menor o igual, mayor o igual → <= , >=
  - Igual, distinto → == , !=
- Lógicos
  - Not (Negación) → !
  - y → &&
  - o → | |

Operadores

- Incremento y decremento → ++ , - -

–Post: i++ (uso, luego incremento)

```
b = 2;
c = 3;
a = b++ * c;
//a vale 6 y b vale 3
```

–Pre: ++i (incremento, luego uso)

```
b = 2;
c = 3;
a = ++b * c;
//a vale 9 y b vale 3
```

Operadores

- Asignación → =

–Asocia de derecha a izquierda:

- a = b = c = 8;

- Operar y asignar

–Los operadores aritméticos y de manejo de bits pueden combinarse con la asignación

–Ejemplo de un sumador

- total = total + dato;
- total += dato;

–Genéricamente si el operador es X entonces

a = a X b;

a X= b;

Universidad Nacional de Lomas de Zamora

Secuencias de escape en E/S

Cuando usamos entrada y salida de datos podemos formatear nuestras cadenas para que integren diferentes tipos de datos

(Ver página siguiente)

Formateador	Salida
%d ó %i	entero en base 10 con signo (int) printf ("el numero enteronen base 10 es: %d" , -10);
%u	entero en base 10 sin signo (int)
%o	entero en base 8 sin signo (int)
%x	entero en base 16, letras en minúscula (int)
%X	entero en base 16, letras en mayúscula (int)
%f	Coma flotante decimal de precisión simple (float)
%lf	Coma flotante decimal de precisión doble (double)
%ld	Entero de 32 bits (long)
%lu	Entero sin signo de 32 bits (unsigned long)
%e	La notación científica (mantisa / exponente), minúsculas (decimal precisión simple ó doble)
%E	La notación científica (mantisa / exponente), mayúsculas (decimal precisión simple ó doble)
%c	carácter (char)
%s	cadena de caracteres (string)

Universidad Nacional de Lomas de Zamora

Ingreso y egreso de datos

- Funciones printf y scanf

–Pertenecen a la biblioteca estándar “stdio” (standard input output)
–Se utilizan secuencias de escape para indicar donde van los datos. Estas secuencias comienzan con %

–Supongamos que x es una variable int con valor 3
printf("Valor inicial: %d \t valor final: %d\n", x * 2, 7);

–Genera como salida
Valor inicial: 6 valor final: 7

–Para ingresar un valor en la variable dato
scanf("%d", &dato);
/*El operador & indica “la dirección de memoria de la variable dato”, ya que scanf siempre guarda lo ingresado en una dirección de memoria*/

Universidad Nacional de Lomas de Zamora

Ingreso y egreso de datos

- Más ejemplos printf y scanf

```
printf("Valor de variable: %d \n", numero_ingresado);
printf("Valor de variable multiplicado: %d \n", numero_ingresado * 2);
printf("Valor de variable multiplicado: %d \t numero literal: %d \n",
numero_ingresado * 5, 7);
```

- Para ingresar un valor en la variable dato

```
int dato ;
scanf("%d", &dato) ;
printf("Valor ingresado en variable dato: %d \n", dato);
```