------------------------------------------------------------------------------------------------------------ **C** ----------------------------------------------------------------------------------------------------------------------

- Debe contener la función **main()**. Aquí es donde inicia el programa.

- Todas las palabras reservadas básicas en el header: **#include<stdio.h>**

- Dependiendo el programa podría necesitar: **stdlib.h, math.h**, entre otras.

- Se deben declarar las librerías al inicio de todo programa.

- Toda función ***int main*** deberá incluir un ***return 0*** una vez concluidas las instrucciones.

--------------------------------------------------------------------------------------- VARIABLES -----------------------------------------------------------------------------------------

**//Algunos tipos de datos en C. Declaración**

**char** caracter = ‘c’; // 1 byte de memoria, rango de -128 a 127.

**int** entero = 55; // 2-4 bytes

**short** corto = 123; // 2 bytes

**long** largo = 123456789; // 4 bytes

**float** real = 0.000001; //4 bytes de memoria

**double** otro\_real = 0.0000000001; //8 bytes

**long double** mas\_real = 0.00000000001; //10-12-16 bytes

**// *sizeof*** este operador obtiene el tamaño en bytes de su operando.

sizeof(char); // 1 sizeof(*corto*); // 2 // ejemplo

----------------------------------------------------------------------------- OPERADORES -----------------------------------------------------------------------------------------

**//Aritméticos: +, -, \*, /, %**

c = a + b; //Suma c = a - b; //Resta

c = a \* b; //Multiplicación c = a / b; //División

c = a % b; //Residuo

**//Asignación: +=, -=, \*=, /=, %=**

c -= a; //Equivalente a: ***c = c - a;*** c += a; //***c = c + a;***

c /= a; //Equivalente a: ***c = c / a;*** c \*= a; //***c = c \* a;***

c %= a; //Equivalente a: ***c = a % b***

c++; //Incrementa en 1 a c

c--; //Decrementa en 1 a c

**//Comparación: >, <, >=, <=, ==, != (1→Verdadero, 0→Falso)**

**>** // mayor que **>=** // mayor o igual que

**<** // menor que **<=** // menor o igual que

!= // diferente que **==** // igual que

**//Booleanos: and, not, or**

AND → **&&** //Ambos casos deben ser verdaderos

OR → **||** //Al menos un caso debe ser verdadero

NOT → **!** //Negación. El caso debe ser falso para cumplir

**//Operador ternario**

*condición* **?** *valor\_si\_verdadero* **:** *valor\_si\_falso*

----------------------------------------------------------ESTRUCTURAS DE CONTROL SELECTIVAS (if / switch)-----------------------------------------------------------------------------

// - En el caso de los condicionales **if**; *else* y *else if* pueden ser omitidos,

// dependiendo la situación del problema.

// - Se pueden incluir todos los *else if* que se deseen. En tal caso, se

// recomienda utilizar ***switch case***

**if** (*condicion1*) { // “**Si”**

//*Instrucciones*

} **else if**(*condicion2*) { // “**De lo contrario si”**

//*Instrucciones*

} **else** { // “**De lo contrario”**

//*Instrucciones*

}

//**Switch,** puede tener tantos *case* como sean necesarios

**switch** (n){

**case** *constante1*:

//*Instrucciones*

**break**;

**case** *constante2*:

//*Instrucciones*

**break**;

**default**:

//*Instrucciones*

**break**;

}

---------------------------------------------------- ESTRUCTURAS DE CONTROL REPETITIVAS (for / while / do-while)-------------------------------------------------------

**//For**

for(**inicialización*;* condiciónDeContinuaciónDelCiclo*;* incremento**) {

//*Instrucciones*

}

**//While**

while(**condiciónDeContinuaciónDelCiclo**) {

//*Instrucciones*

//*incremento*

}

**//Do… While**

do {

//*Instrucciones*

} while(**condiciónDeContinuaciónDelCiclo**)

// Las condiciones siempre deben ser operadores o constantes que regresen 1 o 0**(1→Verdadero, 0→Falso)**

----------------------------------------------------------------------------- FUNCIONES ESTÁNDAR -----------------------------------------------------------------------------------------

**//Funciones de librería estándar**

**// *printf*** lee entradas formateadas

intvariable = 1;

printf( **"**Envía **%d** salida formateada**",** **variable** );

**// *scanf*** lee entradas formateadas

scanf( "**%d**", **&variable** );

**// *getchar*** obtiene un carácter de entrada por teclado

char c;

c = getchar();

**// *putchar*** escribe un carácter especificado por el argumento

putchar(**c**); // variable de tipo char

**// gets** lee una línea desde la entrada y la almacena

char cadena[50];

gets( cadena );

**// *puts*** escribe una cadena pero sin incluir el carácter nulo.

// Se agrega un carácter de nueva línea a la salida.

char cadena[15] = "Hola mundo";

puts(**cadena**); // variable cadena

----------------------------------- ALGUNOS ESPECIFICADORES DE CONVERSIÓN (ESPECIFICADORES DE FORMATO) ------------------------------

**%c** Carácter ASCII correspondiente

**%d, %i** Decimal con signo de un entero

**%x, %X** Hexadecimal sin signo

**%p** Dirección de memoria (puntero)

**%e, %E** Coma flotante con signo en notación científica

**%f, %F** Flotante con signo // printf("%.2f", 3.1416); //3.14

**%g, %G** Coma flotante, usando la notación que requiera menor espacio

**%o** Octal sin signo de un entero

**%u** Decimal sin signo de un entero

**%s** Cadena de caracteres (terminada en '\0')

------------------------------------------------------------- IMPRESIÓN ---------------------------------------------------------------------------

**%%** Imprime el símbolo %

**\n** Nueva línea. Coloca el cursor al principio de la siguiente línea

**\t** Tabulador horizontal

**\a** Alerta. Suena la campana del sistema

**\\** Inserta una diagonal invertida en una cadena

**\”** Comillas. Inserta unas comillas en una cadena