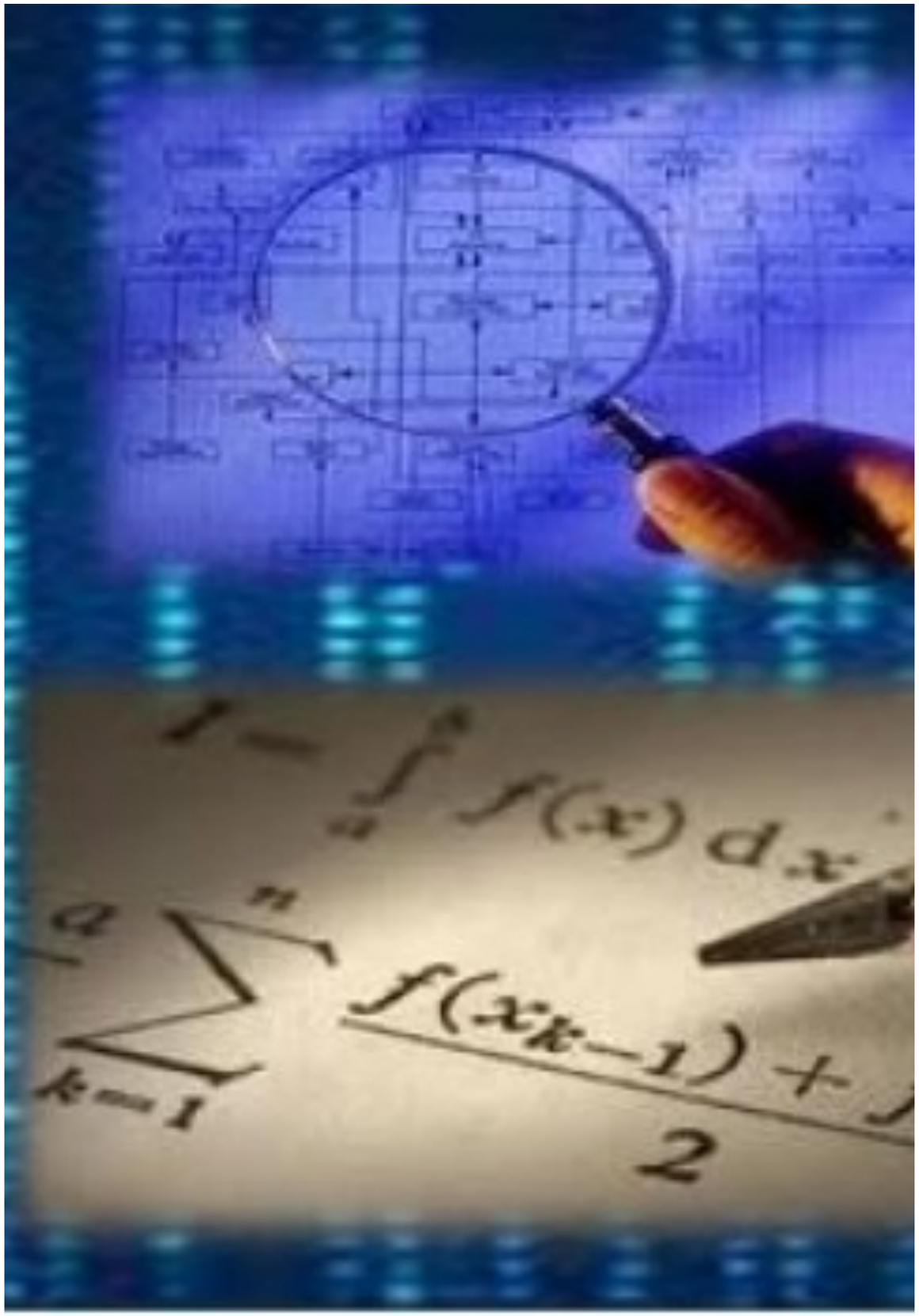
Programación básica

Alma González
Agosto 2021



Highlights de la historia de la programación

- 1843 - La matemática Ada Lovelace trabajó con Charles Babbage, quien desarrolló los planos de la primera computadora mecánica. Ada escribe un algoritmo, o programa, que puede ejecutarse en dicha máquina. Se le considera la primera programadora.
- 1938 - Konrad Zuse diseña y construye lo que consideraríamos una computadora completamente programable, utiliza código binario “ceros y unos”.
- 1945 – John Mauchly y J. Presper Eckert completan la primera computadora electrónica, ENIAC (Electronic Numerical Integrator Analyzer and Computer). Usada en cálculos de balística, ENIAC pesaba ~30 Toneladas y ocupaba un cuarto de 500m^2. Seis mujeres fueron las programadoras principales de ENIAC.
- 1959 - COBOL, es el primer lenguaje que utiliza palabras en lugar de solo números. Desarrollado con base en el trabajo de Grace Hooper.
- 1972 - Se desarrolla el lenguaje C por Dennis Ritchie – AT&T Bell Laboratories.
- 1973 - La cuarta edición del sistema UNIX fue reescrito en lenguaje C. Lo que permitió que fuera portable a diferentes computadoras.
- 1975 - Bill Gates y Paul Allen fundan Microsoft.
- 1976 - Steve Jobs and Steve Wozniak fundan Apple.
- 1981 - la computadora personal IBM debuta en el mercado a un precio accesible.
- 1983 - Se desarrolla el lenguaje C++ por Bjarne Stroustrup.
- 1989 - Se crea la World Wide Web por el cientifico inglés Tim Berners-Lee, quien ademas establece el lengunaje HTML, la ubicación URL y el protocolo de transferencia HTTP.
- 1991 - Se desarrolla el lenguaje de programación Python por Guido Van Russum
- 1991 - Linus Torvalds desarrolla y hace Publico el kernel Linux, que evolucionó en las diferentes distribuciones de Linux. Sistema Operativo de uso libre.
- 1999 - Las fechas seguían un formato MM/DD/YY lo que provocó un poco de temor de que todo fallará al pasar del año 1999 al 2000. Se gastaron millones de dólares en arreglar el “bug” Y2K” antes de que iniciará el nuevo año.

Lenguaje de programación C

- Las ideas del lenguaje provienen de un lenguaje llamado BCPL desarrollado por Martin Richards. Primero se desarrolla B por Ken Thompson en 1970 en los laboratorios Bell, y posteriormente nace C. Dennis Ritchie – AT&T Bell Laboratories – 1972

- 1978 Publicación de “The C Programing lenguage” de Kerningan y Ritchie revoluciona el computo. Especificación del lenguaje

- Ampliamente usado en

    - Diferentes sistemas operativos y arquitecturas.
    - Micro-controladores: Autos, Aviones, etc.
    - Procesadores integrados: Telefonos, tablets, electrónicos en gral.
    - Procesadores Digitales: Sistemas de Audio, TV, etc.
    - Eficiencia/Rendimiento

- Es un lenguaje compilado, i.e. Las instrucciones se escriben en un archivo de texto que es interpretado para generar un ejecutable.

- Programación nivel bajo. Los comandos o funciones en el lenguaje están íntimamente ligadas a las instrucciones del procesador.

Estructura básica de un programa en C

```c
/* Inicia con comentarios.
*/

//Esta es otra forma de poner un comentario
#include <nombre_de_libreria.h>

int main( ){
    exit(0);
}

//Otras funciones definidas.
```

Instrucción/Encabezado #INCLUDE

- Indica que en el programa se incluyan las definiciones de constantes, funciones y otras declaraciones que se encuentran en los archivos especificados.

- #include <nombre_libreria.h> //lee e incluye el contenido de la libreria stdio.h, que se encuentra en el archivo con extensión *.h*

- Se añaden solo aquellas librerías que se van a usar.

- Podemos usar librerías propias (más adelante en el curso) :
  - #include “libreria.h”

  - El compilador buscará primero en el directorio donde se este intentando compilar el programa. Si está en otro directorio se podrá especificar.

Librerías más comunes en C

`#include "stdio.h"` (Librerias estándar de lectura y escritura)
`#include "math.h"` (Librerias estándar de funciones matemáticas como exp, sqrt, cos, sin, log, etc.). Para compilar un programa que incluye esta librería se debe añadir la opción lm. Ejemplo: gcc -lm programa.c -o programa

`#include "stdlib.h"` (Librerias estándar que incluye funciones para manejo de la memoria, generador de números aleatorios, aritmética de números enteros, buscar y ordenar datos, convertir tipos de variables, etc.)

`#include "time.h"` (Contiene definiciones y funciones para obtener y manipular información de fechas y tiempo)

Veremos otras librerías conforme las vayamos utilizando. Para una lista mas extensa ver https://es.wikipedia.org/wiki/Biblioteca_estándar_de_C.

FUNCIÓN main()

- Veremos más detalles durante el curso. Por el momento nos basta saber que contendrá toda la secuencia de instrucciones a realizar por nuestro programa. La función más sencilla es una que no hace nada. Solo se inicia y ejecuta correctamente.

```
int main( ){
    exit(0);
}
```

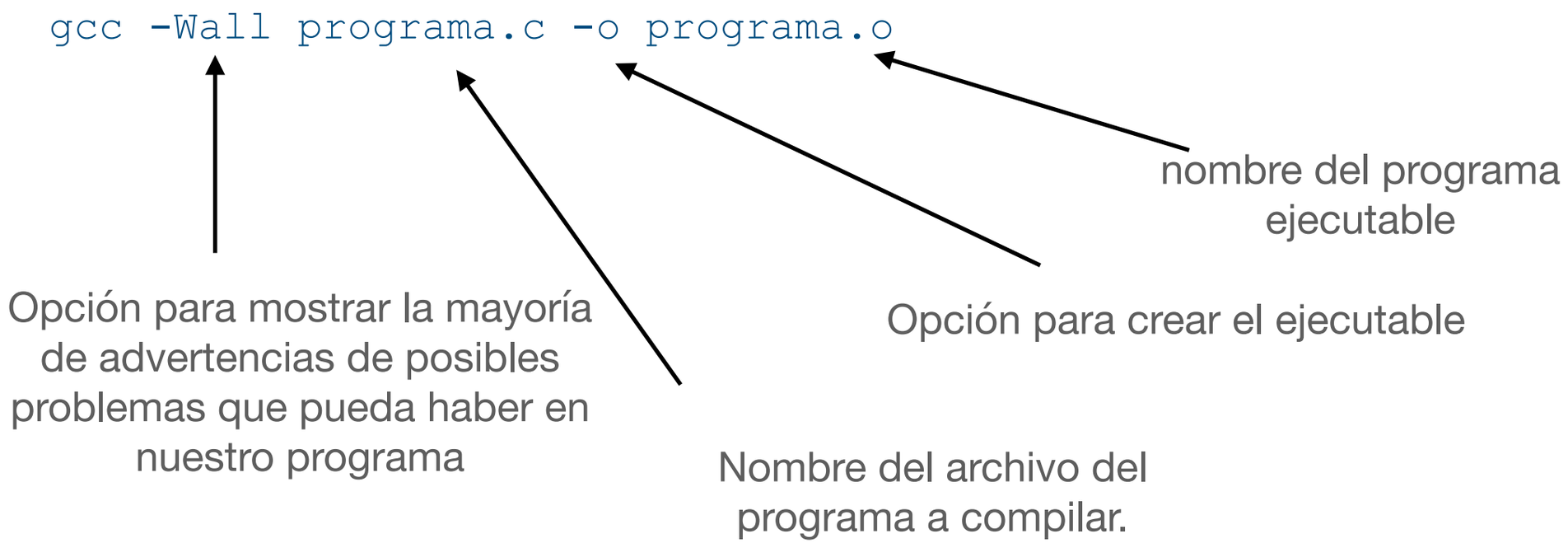
- Las llaves { } nos indican el inicio y final de la función.
- exit(0); indica que la secuencia de instrucciones sucedió correctamente, si no es así entonces enviará un error.
- El cuerpo del programa incluye la declaración de variables y las acciones a realizar con dichas variables. Así como la interacción con el usuario.
- Todas las instrucciones terminan con ; (Es muy importante y una de las fuentes principales de errores!)

Estructura básica de un programa en C

```c
/* Inicia con comentarios. Típicamente:
Fecha de creación:
Autor:
Propósito o funcionalidad del programa.
Historial de versiones, o cambios importantes.
*/

//Esta es otra forma de poner un comentario, generalmente corto, e.g describiendo que hace una línea en particular.
//Los comentarios no afectan la compilación/ejecución del programa.
//Son informativos y muy muy útiles tanto para quien escribe el programa, como para un futuro usuario.

#include <stdlib.h> //librería estándar de C.
#include <stdio.h>  //Librería para interacción con la pantalla.

int main( ){
   //Cuerpo del programa;
   //Se indican todas las instrucciones a realizar. El ejemplo más simple es imprimir un mensaje de saludo a la pantalla.
   printf("Hola, este es nuestro primer programa del curso PB2021\n"); //el carácter "\n" indica un salto de línea.
   exit(0);  // La mayoría de las veces se omite. Regresar un número entero 0 indica que el programa terminó exitosamente.
}

/*
Es posible definir otras funciones, además de la principal.
Por ahora al estar dentro de un comentario, es como si esta función no existiera.
int funcion1(){

   Aquí estarían las instrucciones a realizar por una función secundaria.
   Veremos los diferentes tipos de funciones durante el curso, incluidas diferentes formas de definir la función principal.

}

*/
```

Compilación (básica) de un programa C

- gcc (Disponible en casi cualquier distribución de linux y/o mac en versión ~4).
- https://gcc.gnu.org. La versión mas reciente



- `gcc -Wall -lm programa.c -o programa`

Opción que se debe incluir si se usa la librería math

Cuerpo del programa

- **Declaración de Variables**
- **Instrucciones.**

Iniciar variables

- Se especifica el nombre de la variable y su tipo. Ej.

```
int edad;
float temperatura;
char a;
char nombre[20];
```

- Podemos iniciar el valor desde la declaración (conveniente en algunos casos). Ej:

```
int n=3;
float phi=1.6180339;
```

- Podemos declarar varias variables del mismo tipo a la vez, Ej.

```
int a,b,c=0,d=4;
float T1=12.5, T2;
```

- Aquellas variables a las que que no se le asigne un valor, tendrán uno predeterminado por el compilador, que a priori no sabemos cuál es. Por ello es preferible asignar un valor a las variables antes de utilizarlas.

Variables

- En C todas las variables a usar deben ser declaradas ANTES de usarse.
- La declaración de variables incluye el nombre y tipo de variable.
- Tipos de variable:
    - char : caracter
    - int : Variable del tipo entero.
    - short : entero de
    - signed
    - unsigned …..
- Casi siempre usaremos int, pero en programas donde se requiere el uso óptimo de la memoria es mejor elegir el tipo adecuado.
- float : Variable del tipo punto flotante. ~6 cifras decimales.
- double : Variable del tipo punto flotante de doble precisión, ~15 cifras decimales.

Instrucciones (O SENTENCIAS)

- Se refiere a la secuencia de instrucciones que realizará nuestro programa.
- Es una secuencia por lo que el orden importa.
- Instrucciones de interacción con la terminal. Ej. imprimir a la pantalla ó captar información proporcionada por el usuario :

```
printf (“El texto a mostrar en pantalla \n”);
scanf (“%i”, &edad);   // %i indica el tipo de variable (entero en este caso).
scanf(“%f”, &temperatura);
printf (%i”, edad);
printf (%f”, temperatura);
```

El símbolo & lo veremos más adelante, pero es necesario para poder almacenar el valor tecleado en la terminal a la variable en el programa. Nota que solo está presente cuando la instrucción es **`scanf`**, pero no es necesario en **`printf`**.

- Instrucciones de operaciones entre variables. Ej. suma, resta, división o multiplicación, (u combinaciones de ellas)
- Uso de funciones definidas en otras librerías.

Operaciones aritméticas

- Supongamos que x, y , son dos variables. Podemos hacer las siguientes operaciones con ellas:

  - x+y, x-y, x*y, x/y, x%y

    (% la función modulo calcula el residuo de la división, ej. num%2 será igual a cero si el numero es par, y diferente de cero si es impar)

- Podemos asignar el resultado de una operación entre variables a otra variable (previamente declarada). Incluso podemos sustituir una misma variable repetidas veces, siendo conscientes que perdemos la información anterior. Ej.

  - y = x+3∗x/(y−4) (Sobre-escribimos el valor de y a su nuevo valor)
  - z = x+3∗x/(y−4) (asignamos el resultado de la operación a una variable diferente.)

Cuando sabemos que no volveremos a usar la información previa de la variable podemos sobre escribirla, esto es util para no definir muchas variables de forma innecesaria.

Orden de las operaciones

Jerarquia

Operador	Dirección de evaluación
()	
*,/,%	Izq-Der
+,-	Izq-Der
=,+=,-=,*=,/=,%=	DER-IZQ

No es lo mismo 2+5*10 que (2+5)*10. El uso de los paréntesis es muy importante para especificar la operación que se quiere realizar

No es lo mismo x=5 que 5=x. Lo segundo no tiene mucho sentido pues 5 es un número, no una variable.

Ejemplo.

```c
/*
Agosto 25, 2021
Alma González
Primer programa, mostrando estructura, definición de variables, imprimir y leer valores de variables en la pantalla.
*/


#include <stdlib.h> //librería estándar de C.
#include <stdio.h>  //Librería para interacción con la pantalla.

int main( ){
    char nombre[10];
    int edad;
    float temperatura;
    //Cuerpo del programa;
    printf ("¿Cuál es tu nombre? \n");
    scanf("%s", nombre);
    printf("Hola, %s este es el segundo programa del curso PB2021\n",nombre); //el carácter "\n" indica un salto de línea.

    printf ("Introduce tu edad \n");
    scanf ("%i", &edad);   // %i indica el tipo de variable (entero en este caso).
    printf ("¿Qué temperatura marcó el termomentro la última vez que fuiste al supermercado?  \n");
    scanf("%f", &temperatura);

    printf("Tu edad es: %i\n", edad);
    printf("Tu último registro de temperatura fué: %f\n", temperatura);
    exit(0);
}

```

Compilación y ejecución

```
almagonzalez@Almas-Mac-Pro-2 Pruebas % gcc primer_programa.c -o primer_programa.o
almagonzalez@Almas-Mac-Pro-2 Pruebas % ls
primer_programa.c       primer_programa.o
almagonzalez@Almas-Mac-Pro-2 Pruebas % ./primer_programa.o
¿Cuál es tu nombre?
Alma
Hola, Alma este es el segundo programa del curso PB2021
Introduce tu edad
37
¿Qué temperatura marcó el termomentro la última vez que fuiste al supermercado?
36.5
Tu edad es: 37
Tu último registro de temperatura fué: 36.500000
almagonzalez@Almas-Mac-Pro-2 Pruebas %
```

Ejercicios

- Escribe un programa que te solicite introducir 4 números enteros. El programa debe imprimir a la pantalla las siguientes operaciones. El nombre del archivo C debe ser operaciones_int.c

$$e = (a + b) * c / d$$

$$e = ((a + b) * c) / d$$

$$e = (a + b) * c / d$$

$$e = a + (b * c) / d$$

  - Recuerda añadir comentarios a tu código.

- Crea otro programa igual que el anterior, pero donde la entrada sean números reales (float). El nombre del archivo C debe ser operaciones_int.c