



Instituto Politécnico Nacional

INSTITUTO POLITÉCNICO NACIONAL



Escuela Superior de Computo

Fundamentos De Programación

Pérez García Edgar Israel.


Control de flujo en C.

30 de septiembre de 2025.

Introducción.

Todo programa de computadora, para ser útil, necesita interactuar con el mundo exterior. Un programa que solo realiza cálculos internos sin una forma de recibir datos o mostrar resultados es un ejercicio puramente teórico. Esta comunicación fundamental se logra a través de las operaciones de **Entrada y Salida (E/S)**, que actúan como el puente entre la lógica abstracta del código y el usuario, los archivos o cualquier otro dispositivo. En el lenguaje de programación C, el manejo de la E/S no es parte del núcleo del lenguaje en sí, sino que se delega a una colección de funciones robustas y estandarizadas contenidas en la biblioteca **stdio.h** (Standard Input/Output).

El modelo de E/S de C se basa en el concepto de "flujos" o *streams*, que son secuencias de datos que fluyen desde una fuente hacia un destino. Por defecto, todo programa en C tiene acceso a tres flujos estándar:

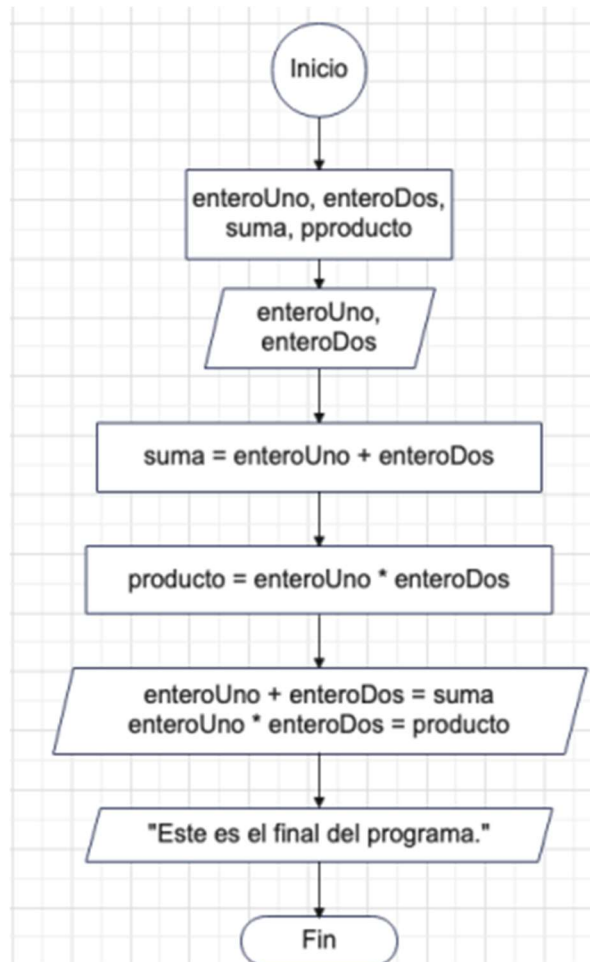
1. **stdin (Entrada Estándar)**: Es el flujo de entrada principal, usualmente conectado al **teclado** .
2. **stdout (Salida Estándar)**: El flujo de salida principal, conectado por lo general a la **pantalla** o terminal.
3. **stderr (Error Estándar)**: Un flujo de salida secundario, también conectado a la pantalla, reservado específicamente para mostrar mensajes de error.

Para interactuar con estos flujos, `stdio.h` provee una familia de funciones, siendo las más conocidas **printf()** y **scanf()**. La función `printf()` ("print formatted") es la herramienta principal para enviar datos a `stdout`, permitiendo construir cadenas de texto complejas insertando valores de variables con un formato específico mediante códigos como `%d` (para enteros) o `%s` (para cadenas). Por otro lado, **scanf()** ("scan formatted") es su contraparte para leer datos desde `stdin`, interpretando la entrada del usuario y convirtiéndola al tipo de dato adecuado para ser almacenada en una variable.

Más allá de estas funciones de alto nivel, C también ofrece mecanismos más simples como `getchar()` y `putchar()` para leer y escribir caracteres individuales. La belleza de este sistema es su consistencia: los mismos principios y funciones que se usan para la E/S estándar se extienden al manejo de archivos, permitiendo que un programa lea y escriba información en el almacenamiento persistente. Dominar las herramientas de la biblioteca `stdio.h` es, por lo tanto, un paso esencial para transformar un programa de una simple secuencia de instrucciones a una aplicación interactiva y funcional.

Desarrollo.

1. Escriba un programa que lea dos enteros y luego genere para ambos su suma y su producto. La última instrucción de salida podría ser el siguiente:

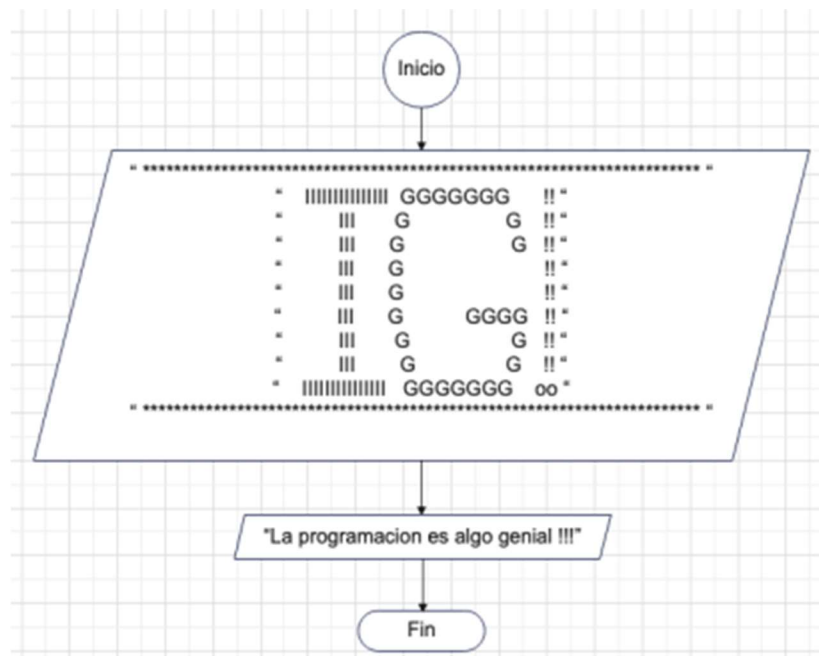


```
printf( "Este es el final del programa." );
```

Se desarrolló el código fuente en PracticaTresNumeros.c contenido en la misma carpeta que este documento

2. Escriba un programa que imprima “I G !” en letras mayúsculas grandes dentro de un contorno de * (asteriscos) seguido de dos líneas en blanco y luego el mensaje La programación es algo genial. La salida debe tener el siguiente aspecto:

```
*****
||||||| GGGGGG !!
||| G G !!
||| G G !!
||| G !!
||| G !!
||| G GGGG !!
||| G G !!
||| G G !!
||||||| GGGGGG oo
*****
La programacion es algo genial !!!
```



Se desarrolló el código fuente en PracticaTresTextolG.c contenido en la misma carpeta que este documento

3. Escriba un programa que le permita al usuario ingresar una cantidad de cuartos, diez centavos, y monedas de cinco centavos y luego generar el valor monetario de las monedas en centavos.

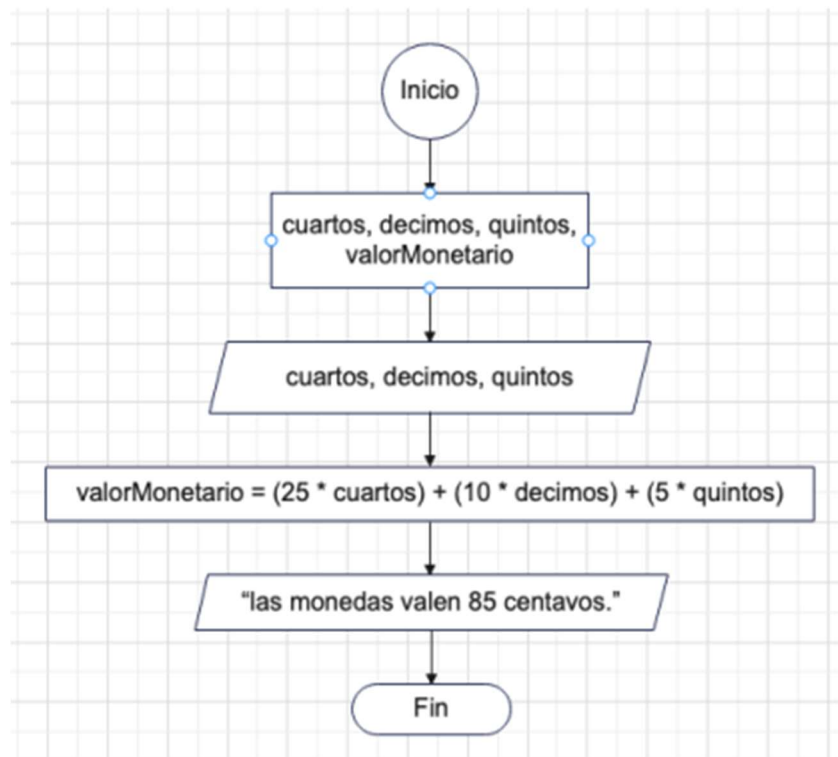
Por ejemplo, si el usuario ingresa:

2 para el número de cuartos,

3 para el número de monedas de diez centavos y

1 para el número de monedas de cinco centavos

Entonces el programa debería tener como salida que las monedas valen 85 centavos.

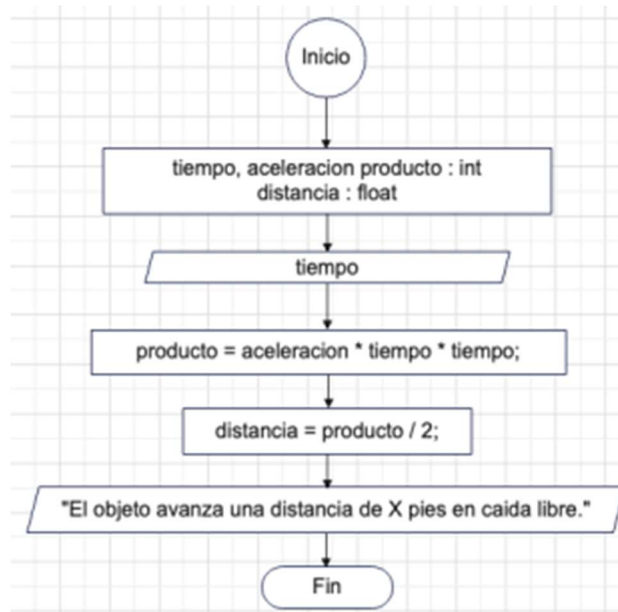


Se desarrolló el código fuente en PracticaTresTextolG.c contenido en la misma carpeta que este documento

4. Escriba un programa que le permita al usuario ingresar un tiempo en segundos y luego muestra qué tan lejos caería un objeto si está en caída libre durante esa longitud de hora. Suponga que el objeto comienza en reposo, no hay fricción ni resistencia del aire, y hay una aceleración constante de 32 pies por segundo debido a la gravedad. Use la ecuación:

$$distancia = \frac{aceleracion \times tiempo^2}{2}$$

Primero debe calcular el producto y luego dividir el resultado por 2.



Se desarrolló el código fuente en PracticaTresTextolG.c contenido en la misma carpeta que este documento

Conclusión.

Las operaciones de entrada y salida son la base de la programación funcional en C. Sin ellas, un programa es una entidad aislada incapaz de recibir datos, interactuar con el usuario o almacenar resultados de forma persistente. A través de la librería **stdio.h**, C proporciona un mecanismo estandarizado y potente que transforma el código de una simple lógica abstracta a una herramienta práctica, permitiendo la comunicación esencial entre el programa y el mundo exterior, ya sea un usuario, un archivo o cualquier otro dispositivo. Las operaciones de entrada y salida son la base de la programación funcional en C. Sin ellas, un programa es una entidad aislada incapaz de recibir datos, interactuar con el usuario o almacenar resultados de forma persistente. A través de la librería **stdio.h**, C proporciona un mecanismo estandarizado y potente que transforma el código de una

simple lógica abstracta a una herramienta práctica, permitiendo la comunicación esencial entre el programa y el mundo exterior, ya sea un usuario, un archivo o cualquier otro dispositivo.

Referencias.

Kernighan, B. W., & Ritchie, D. M. (2006). *The C programming language* (2nd ed.). Prentice Hall.

King, K. N. (2008). *C programming: A modern approach* (2nd ed.). W. W. Norton & Company.

Prata, S. (2013). *C primer plus* (6th ed.). Addison-Wesley Professional.