

Universidad tecnológica de Honduras

tercer parcial

del

Primer periodo

**Alumno**

Raúl Fernando Ramos Lara

**Carrea**

Ingeniería en computación

**Catedrático**

Ing. Jair Hernández

**Espacio formativo**

Análisis y diseño de algoritmos

Fecha de entrega

19/03/2025

**Investigación**

**¿Qué son los comentarios y estructura de C++?**

### R//Comentarios en C++

Los comentarios en C++ sirven para documentar el código y mejorar su legibilidad. No afectan la ejecución del programa. Hay dos tipos principales:

1. **Comentario de una sola línea:** Se usa // al inicio de la línea.
2. **Comentario de varias líneas:** Se encierra entre /\* y \*/.
3. **Comentarios con TODO, FIXME y NOTE:**  
   Son usados para marcar tareas pendientes o notas importantes en el código. (// TODO: Mejorar la eficiencia de este algoritmo.

// FIXME: Corregir el error de división por cero.

// NOTE: Esta función solo se usa en modo depuración.)

### ****Estructura Avanzada de un Programa en C++****

Un programa en C++ puede incluir más elementos como funciones, clases y estructuras de control.

1. **Directivas de preprocesador:** ( **#include <iostream>#include <cmath> // Se pueden incluir más bibliotecas según sea necesario)**
2. **Espacios de nombres:** (using namespace std; // Para evitar usar std:: en cada instrucción)
3. **Declaraciones globales (opcional): (const double PI = 3.1416; // Definición de una constante global, int globalVar = 100; // Variable global)**
4. **Funciones**: (void mostrarMensaje() { cout << "Hola desde una función!" << endl;})
5. **Clase y métodos (POO en C++):** (class Persona {public: string nombre;int edad;void mostrarInfo() { cout << "Nombre: " << nombre << ", Edad: " << edad << endl; }};)
6. **Función** main **con estructura:** **class Persona {public:string nombre;int edad;void mostrarInfo() {cout << "Nombre: " << nombre << ", Edad: " << edad << endl;}};**

**¿Qué son los encabezados de C++ (librerías) y de 4 ejemplos**

R//los **encabezados** (o **librerías de encabezado**) son archivos que contienen declaraciones de funciones, clases, macros y otras definiciones que pueden usarse en un programa sin necesidad de reescribirlas. Estos encabezados permiten reutilizar código y organizarlo de manera modular.

Los archivos de encabezado tienen la extensión **.h** o **sin extensión en las librerías estándar de C++** (por ejemplo, <iostream> en lugar de iostream.h).

### 🔹 Ejemplos de encabezados en C++

1. **<iostream>** → Permite la entrada y salida estándar (como cin y cout).
2. **<vector>** → Contiene la clase std::vector para manejar arreglos dinámicos.
3. **<cmath>** → Proporciona funciones matemáticas como sqrt(), pow(), sin(), etc.
4. **<string>** → Permite manipular cadenas de texto mediante la clase std::string.

**¿Cuáles son los tipos de datos que se usan en C++?**

## R//**1. Tipos de datos primitivos (básicos)**

Son los tipos de datos fundamentales proporcionados por el lenguaje:

| **Tipo** | **Descripción** | **Tamaño aproximado** |
| --- | --- | --- |
| int | Enteros (positivos y negativos) | 4 bytes |
| float | Números de punto flotante de precisión simple | 4 bytes |
| double | Números de punto flotante de doble precisión | 8 bytes |
| char | Un solo carácter (almacena un número ASCII) | 1 byte |
| bool | Valores booleanos (true o false) | 1 byte |
| void | No almacena datos, se usa en funciones | - |

**2. Modificadores de tipo:** algunos tipos pueden modificarse para ajustar su rango y tamaño:

| **Modificador** | **Ejemplo** | **Descripción** |
| --- | --- | --- |
| short | short int | Entero corto (generalmente 2 bytes) |
| long | long int | Entero largo (4 u 8 bytes) |
| unsigned | unsigned int | Solo valores positivos |
| signed | signed int | Valores positivos y negativos (por defecto) |

**3. Tipos de datos compuestos:** Son estructuras más complejas que agrupan múltiples valores:

| **Tipo** | **Descripción** |
| --- | --- |
| string | Almacena cadenas de texto (requiere <string>) |
| array | Arreglo de elementos del mismo tipo |
| struct | Agrupación de datos bajo una misma estructura |
| union | Similar a struct, pero los miembros comparten memoria |
| enum | Conjunto de valores constantes con nombre |

**4. Tipos de datos punteros:** permiten almacenar direcciones de memoria (int x = 10; int\* ptr = &x; // Puntero a la dirección de x)

**5. Tipos definidos por el usuario:** C++ permite crear nuevos tipos con typedef y using (typedef unsigned int Edad;using Altura = double;Edad e = 20;Altura h = 1.75;)

**¿Cuáles son las entradas y salidas de datos en C++ CIN y COUT-capítulo2?**

R//En C++, la entrada y salida de datos se manejan principalmente con los objetos cin y cout, que pertenecen a la biblioteca estándar <iostream>.

* **Salida de datos con COUT**

El objeto cout se usa para mostrar información en la pantalla. Se acompaña del operador << (operador de inserción).

* std::cout → Se usa para imprimir en la consola.
* << → Inserta el contenido en la salida estándar.
* std::endl → Salto de línea (también se puede usar "\n").

**Ejemplo:** **#include <iostream>**

**int main() {**

**std::cout << "Hola, mundo!" << std::endl;**

**return 0;**

**}**

* **Entrada de datos con CIN**

El objeto cin se usa para recibir datos del usuario. Se acompaña del operador >> (operador de extracción).

* std::cin → Captura la entrada del usuario.
* >> → Extrae el valor ingresado y lo guarda en la variable edad.

**Ejemplo:** **#include <iostream>**

**int main() {**

**int edad;**

**std::cout << "Ingresa tu edad: ";**

**std::cin >> edad;**

**std::cout << "Tienes " << edad << " años." << std::endl;**

**return 0;**

**}**

**¿Cuáles son los operadores lógicos y aritméticos en C++?**

R//En **C++**, los operadores se dividen en varias categorías, incluyendo **operadores aritméticos** y **operadores lógicos**.

### Operadores aritméticos

Estos operadores realizan operaciones matemáticas sobre valores numéricos.

* + Suma (Ejemplo: 10 + 3 da 13)
* - Resta (Ejemplo: 10 - 3 da 7)
* \* Multiplicación (Ejemplo: 10 \* 3 da 30)
* / División (Ejemplo: 10 / 3 da 3 en división entera)
* % Módulo o residuo (Ejemplo: 10 % 3 da 1)
* ++ Incremento (Ejemplo: a++ o ++a aumenta a en 1)
* -- Decremento (Ejemplo: b-- o --b disminuye b en 1)

### Operadores lógicos

Se utilizan en expresiones booleanas (true o false) y son clave en estructuras de control como if, while, etc.

* && AND lógico (Ejemplo: true && false da false)
* || OR lógico (Ejemplo: true || false da true)
* ! NOT lógico (Ejemplo: !true da false)

**¿Cuáles son las estructuras de selección?**

R//En **C++**, las estructuras de selección permiten tomar decisiones en función de condiciones, y ejecutar diferentes bloques de código dependiendo de si esas condiciones son verdaderas o falsas. Las principales estructuras de selección en C++ son:

**1. Estructura if**

La estructura if ejecuta un bloque de código si una condición es verdadera.

**Sintaxis:** **if (condición) {**

**// Código a ejecutar si la condición es verdadera**

**}**

### 2. ****Estructura**** if-else

La estructura if-else ejecuta un bloque de código si la condición es verdadera, y otro bloque diferente si la condición es falsa.

**Sintaxis:** **#include <iostream>**

**int main() {**

**int x = 3;**

**if (x > 5) {**

**std::cout << "x es mayor que 5" << std::endl;**

**} else {**

**std::cout << "x no es mayor que 5" << std::endl;**

**}**

**return 0;**

**}**

**3. Estructura if-else if-else**

Esta estructura permite evaluar múltiples condiciones de manera secuencial. Si una condición es verdadera, se ejecutará el bloque correspondiente; si no, se evaluará la siguiente condición.

**Sintaxis:** **#include <iostream>**

**int main() {**

**int x = 8;**

**if (x > 10) {**

**std::cout << "x es mayor que 10" << std::endl;**

**} else if (x == 8) {**

**std::cout << "x es igual a 8" << std::endl;**

**} else {**

**std::cout << "x es menor que 8" << std::endl;**

**}**

**return 0;**

**}**

**4. Estructura switch**

La estructura switch evalúa una expresión de manera más eficiente cuando hay múltiples posibles valores para una sola variable, en comparación con el uso de múltiples if y else.

**Sintaxis:** **#include <iostream>**

**int main() {**

**int x = 2;**

**switch (x) {**

**case 1:**

**std::cout << "x es 1" << std::endl;**

**break;**

**case 2:**

**std::cout << "x es 2" << std::endl;**

**break;**

**default:**

**std::cout << "x es otro valor" << std::endl;**

**break;**

**}**

**return 0;**

**}**