# Soluciones de Ejercicios de C++ (1-20)

## Ejercicio 1: Hola Mundo

**Problema**: Escribe un programa que imprima “¡Hola, C++!” en la pantalla.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "¡Hola, C++!" << endl;
    return 0;
}
```

## Ejercicio 2: Variables y Entrada de Datos

**Problema**: Solicita al usuario su nombre y edad, luego muestra un mensaje personalizado.

**Solución**:

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    string nombre;
    int edad;

    cout << "Ingresa tu nombre: ";
    getline(cin, nombre);
    cout << "Ingresa tu edad: ";
    cin >> edad;

    cout << "Hola " << nombre << ", tienes " << edad << " años." << endl;
    return 0;
}
```

## Ejercicio 3: Operaciones Aritméticas

**Problema**: Solicita dos números y muestra la suma, resta, multiplicación y división.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    double num1, num2;

    cout << "Ingresa el primer número: ";
    cin >> num1;
    cout << "Ingresa el segundo número: ";
    cin >> num2;

    cout << "Suma: " << num1 + num2 << endl;
    cout << "Resta: " << num1 - num2 << endl;
    cout << "Multiplicación: " << num1 * num2 << endl;

    if (num2 != 0) {
        cout << "División: " << num1 / num2 << endl;
    } else {
        cout << "No se puede dividir por cero" << endl;
    }

    return 0;
}
```

## Ejercicio 4: Par o Impar

**Problema**: Determina si un número ingresado por el usuario es par o impar.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    int numero;

    cout << "Ingresa un número: ";
    cin >> numero;

    if (numero % 2 == 0) {
        cout << numero << " es par." << endl;
    } else {
        cout << numero << " es impar." << endl;
    }

    return 0;
}
```

## Ejercicio 5: Mayor de Tres Números

**Problema**: Encuentra el mayor de tres números ingresados por el usuario.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    double num1, num2, num3, mayor;

    cout << "Ingresa tres números:" << endl;
    cout << "Primer número: ";
    cin >> num1;
    cout << "Segundo número: ";
    cin >> num2;
    cout << "Tercer número: ";
    cin >> num3;

    mayor = num1; // Asumimos que el primero es el mayor

    if (num2 > mayor) {
        mayor = num2;
    }
    if (num3 > mayor) {
        mayor = num3;
    }

    cout << "El número mayor es: " << mayor << endl;
    return 0;
}
```

## Ejercicio 6: Calificaciones

**Problema**: Convierte una calificación numérica (0-100) a letra (A, B, C, D, F).

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    int calificacion;

    cout << "Ingresa la calificación (0-100): ";
    cin >> calificacion;

    if (calificacion >= 90 && calificacion <= 100) {
        cout << "Calificación: A" << endl;
    } else if (calificacion >= 80) {
        cout << "Calificación: B" << endl;
    } else if (calificacion >= 70) {
        cout << "Calificación: C" << endl;
    } else if (calificacion >= 60) {
        cout << "Calificación: D" << endl;
    } else if (calificacion >= 0) {
        cout << "Calificación: F" << endl;
    } else {
        cout << "Calificación inválida" << endl;
    }

    return 0;
}
```

## Ejercicio 7: Tabla de Multiplicar

**Problema**: Genera la tabla de multiplicar de un número dado del 1 al 10.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    int numero;

    cout << "Ingresa un número para generar su tabla de multiplicar: ";
    cin >> numero;

    cout << "\nTabla de multiplicar del " << numero << ":" << endl;
    for (int i = 1; i <= 10; i++) {
        cout << numero << " x " << i << " = " << numero * i << endl;
    }

    return 0;
}
```

## Ejercicio 8: Suma de Números del 1 al N

**Problema**: Calcula la suma de todos los números enteros desde 1 hasta N.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    int n, suma = 0;

    cout << "Ingresa un número N: ";
    cin >> n;

    // Método 1: Usando bucle for
    for (int i = 1; i <= n; i++) {
        suma += i;
    }

    cout << "La suma de números del 1 al " << n << " es: " << suma << endl;

    // Método 2: Usando fórmula matemática
    int sumaFormula = n * (n + 1) / 2;
    cout << "Verificación con fórmula: " << sumaFormula << endl;

    return 0;
}
```

## Ejercicio 9: Factorial

**Problema**: Calcula el factorial de un número.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    int numero;
    long long factorial = 1;

    cout << "Ingresa un número para calcular su factorial: ";
    cin >> numero;

    if (numero < 0) {
        cout << "El factorial no está definido para números negativos." << endl;
    } else if (numero == 0 || numero == 1) {
        cout << "El factorial de " << numero << " es: 1" << endl;
    } else {
        for (int i = 2; i <= numero; i++) {
            factorial *= i;
        }
        cout << "El factorial de " << numero << " es: " << factorial << endl;
    }

    return 0;
}
```

## Ejercicio 10: Números Primos

**Problema**: Determina si un número es primo.

**Solución**:

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    int numero;
    bool esPrimo = true;

    cout << "Ingresa un número para verificar si es primo: ";
    cin >> numero;

    if (numero <= 1) {
        esPrimo = false;
    } else if (numero == 2) {
        esPrimo = true;
    } else if (numero % 2 == 0) {
        esPrimo = false;
    } else {
        // Verificar divisibilidad desde 3 hasta la raíz cuadrada del número
        for (int i = 3; i <= sqrt(numero); i += 2) {
            if (numero % i == 0) {
                esPrimo = false;
                break;
            }
        }
    }

    if (esPrimo) {
        cout << numero << " es un número primo." << endl;
    } else {
        cout << numero << " no es un número primo." << endl;
    }

    return 0;
}
```

## Ejercicio 11: Adivinar el Número

**Problema**: Crea un juego donde el programa genera un número aleatorio y el usuario debe adivinarlo.

**Solución**:

```cpp
#include <iostream>
#include <random>
#include <ctime>
using namespace std;

int main() {
    // Inicializar generador de números aleatorios
    srand(time(0));
    int numeroSecreto = rand() % 100 + 1; // Número entre 1 y 100
    int intento;
    int intentos = 0;

    cout << "¡Bienvenido al juego de adivinar el número!" << endl;
    cout << "He pensado un número entre 1 y 100. ¡Intenta adivinarlo!" << endl;

    do {
        cout << "Ingresa tu número: ";
        cin >> intento;
        intentos++;

        if (intento > numeroSecreto) {
            cout << "Demasiado alto. Intenta con un número menor." << endl;
        } else if (intento < numeroSecreto) {
            cout << "Demasiado bajo. Intenta con un número mayor." << endl;
        } else {
            cout << "¡Felicidades! Adivinaste el número " << numeroSecreto;
            cout << " en " << intentos << " intentos." << endl;
        }

    } while (intento != numeroSecreto);

    return 0;
}
```

## Ejercicio 12: Función para Calcular Área

**Problema**: Crea funciones para calcular el área de un círculo, rectángulo y triángulo.

**Solución**:

```cpp
#include <iostream>
#include <cmath>
using namespace std;

// Función para calcular área del círculo
double areaCirculo(double radio) {
    return M_PI * radio * radio;
}

// Función para calcular área del rectángulo
double areaRectangulo(double base, double altura) {
    return base * altura;
}

// Función para calcular área del triángulo
double areaTriangulo(double base, double altura) {
    return (base * altura) / 2.0;
}

int main() {
    int opcion;

    cout << "Calculadora de Áreas" << endl;
    cout << "1. Círculo" << endl;
    cout << "2. Rectángulo" << endl;
    cout << "3. Triángulo" << endl;
    cout << "Selecciona una opción: ";
    cin >> opcion;

    switch (opcion) {
        case 1: {
            double radio;
            cout << "Ingresa el radio del círculo: ";
            cin >> radio;
            cout << "Área del círculo: " << areaCirculo(radio) << endl;
            break;
        }
        case 2: {
            double base, altura;
            cout << "Ingresa la base del rectángulo: ";
            cin >> base;
            cout << "Ingresa la altura del rectángulo: ";
            cin >> altura;
            cout << "Área del rectángulo: " << areaRectangulo(base, altura) << endl;
            break;
        }
        case 3: {
            double base, altura;
            cout << "Ingresa la base del triángulo: ";
            cin >> base;
            cout << "Ingresa la altura del triángulo: ";
            cin >> altura;
            cout << "Área del triángulo: " << areaTriangulo(base, altura) << endl;
            break;
        }
        default:
            cout << "Opción no válida." << endl;
    }

    return 0;
}
```

## Ejercicio 13: Arrays - Promedio de Calificaciones

**Problema**: Almacena 5 calificaciones en un array y calcula su promedio.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    const int TAMANIO = 5;
    double calificaciones[TAMANIO];
    double suma = 0.0;

    cout << "Ingresa 5 calificaciones:" << endl;

    // Leer calificaciones
    for (int i = 0; i < TAMANIO; i++) {
        cout << "Calificación " << (i + 1) << ": ";
        cin >> calificaciones[i];
        suma += calificaciones[i];
    }

    // Calcular promedio
    double promedio = suma / TAMANIO;

    cout << "\nCalificaciones ingresadas: ";
    for (int i = 0; i < TAMANIO; i++) {
        cout << calificaciones[i] << " ";
    }

    cout << "\nPromedio: " << promedio << endl;

    // Determinar si está aprobado (promedio >= 70)
    if (promedio >= 70) {
        cout << "¡Felicidades! Estás aprobado." << endl;
    } else {
        cout << "Necesitas mejorar tus calificaciones." << endl;
    }

    return 0;
}
```

## Ejercicio 14: Búsqueda en Array

**Problema**: Busca un número específico en un array y muestra su posición.

**Solución**:

```cpp
#include <iostream>
using namespace std;

int main() {
    const int TAMANIO = 10;
    int numeros[TAMANIO] = {12, 45, 23, 67, 89, 34, 56, 78, 90, 11};
    int numeroBuscado;
    bool encontrado = false;
    int posicion = -1;

    cout << "Array: ";
    for (int i = 0; i < TAMANIO; i++) {
        cout << numeros[i] << " ";
    }
    cout << endl;

    cout << "Ingresa el número a buscar: ";
    cin >> numeroBuscado;

    // Búsqueda lineal
    for (int i = 0; i < TAMANIO; i++) {
        if (numeros[i] == numeroBuscado) {
            encontrado = true;
            posicion = i;
            break; // Salir del bucle cuando encontremos el número
        }
    }

    if (encontrado) {
        cout << "El número " << numeroBuscado << " se encontró en la posición " << posi
cion << endl;
    } else {
        cout << "El número " << numeroBuscado << " no se encontró en el array." << endl
;
    }

    return 0;
}
```

## Ejercicio 15: Ordenamiento Burbuja

**Problema**: Implementa el algoritmo de ordenamiento burbuja para ordenar un array.

**Solución**:

```cpp
#include <iostream>
using namespace std;

// Función para intercambiar dos elementos
void intercambiar(int& a, int& b) {
    int temp = a;
    a = b;
    b = temp;
}

// Función para mostrar el array
void mostrarArray(int arr[], int tamanio) {
    for (int i = 0; i < tamanio; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;
}

// Algoritmo de ordenamiento burbuja
void ordenamientoBurbuja(int arr[], int tamanio) {
    for (int i = 0; i < tamanio - 1; i++) {
        for (int j = 0; j < tamanio - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                intercambiar(arr[j], arr[j + 1]);
            }
        }
    }
}

int main() {
    const int TAMANIO = 8;
    int numeros[TAMANIO] = {64, 34, 25, 12, 22, 11, 90, 88};

    cout << "Array original: ";
    mostrarArray(numeros, TAMANIO);

    ordenamientoBurbuja(numeros, TAMANIO);

    cout << "Array ordenado: ";
    mostrarArray(numeros, TAMANIO);

    return 0;
}
```

## Ejercicio 16: Manipulación de Strings

**Problema**: Crea un programa que cuente vocales y consonantes en una palabra.

**Solución**:

```cpp
#include <iostream>
#include <string>
#include <cctype>
using namespace std;

int main() {
    string palabra;
    int vocales = 0, consonantes = 0;

    cout << "Ingresa una palabra: ";
    getline(cin, palabra);

    for (char c : palabra) {
        if (isalpha(c)) { // Verificar si es una letra
            c = tolower(c); // Convertir a minúscula

            if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
                vocales++;
            } else {
                consonantes++;
            }
        }
    }

    cout << "Palabra: " << palabra << endl;
    cout << "Vocales: " << vocales << endl;
    cout << "Consonantes: " << consonantes << endl;
    cout << "Total de letras: " << vocales + consonantes << endl;

    return 0;
}
```

## Ejercicio 17: Palíndromo

**Problema**: Determina si una palabra es un palíndromo.

**Solución**:

```cpp
#include <iostream>
#include <string>
#include <algorithm>
#include <cctype>
using namespace std;

bool esPalindromo(string palabra) {
    // Convertir a minúsculas y eliminar espacios
    string procesada = "";
    for (char c : palabra) {
        if (isalnum(c)) {
            procesada += tolower(c);
        }
    }

    // Comparar la palabra con su reverso
    string reverso = procesada;
    reverse(reverso.begin(), reverso.end());

    return procesada == reverso;
}

int main() {
    string palabra;

    cout << "Ingresa una palabra o frase: ";
    getline(cin, palabra);

    if (esPalindromo(palabra)) {
        cout << "\"" << palabra << "\" es un palíndromo." << endl;
    } else {
        cout << "\"" << palabra << "\" no es un palíndromo." << endl;
    }

    return 0;
}
```

## Ejercicio 18: Clase Rectángulo

**Problema**: Crea una clase Rectángulo con métodos para calcular área y perímetro.

**Solución**:

```cpp
#include <iostream>
using namespace std;

class Rectangulo {
private:
    double base;
    double altura;

public:
    // Constructor
    Rectangulo(double b, double a) {
        if (b > 0 && a > 0) {
            base = b;
            altura = a;
        } else {
            base = altura = 1; // Valores por defecto
        }
    }

    // Método para calcular el área
    double calcularArea() const {
        return base * altura;
    }

    // Método para calcular el perímetro
    double calcularPerimetro() const {
        return 2 * (base + altura);
    }

    // Getters
    double getBase() const { return base; }
    double getAltura() const { return altura; }

    // Setters
    void setBase(double b) {
        if (b > 0) base = b;
    }

    void setAltura(double a) {
        if (a > 0) altura = a;
    }

    // Método para mostrar información
    void mostrarInfo() const {
        cout << "Rectángulo:" << endl;
        cout << "Base: " << base << endl;
        cout << "Altura: " << altura << endl;
        cout << "Área: " << calcularArea() << endl;
        cout << "Perímetro: " << calcularPerimetro() << endl;
    }
};

int main() {
    double base, altura;

    cout << "Ingresa la base del rectángulo: ";
    cin >> base;
    cout << "Ingresa la altura del rectángulo: ";
    cin >> altura;

    Rectangulo rect(base, altura);
    rect.mostrarInfo();
```

```
    // Cambiar dimensiones
    cout << "\nCambiando las dimensiones..." << endl;
    rect.setBase(10);
    rect.setAltura(5);
    rect.mostrarInfo();

    return 0;
}
```

## Ejercicio 19: Clase Cuenta Bancaria

**Problema**: Crea una clase CuentaBancaria con métodos para depositar, retirar y consultar saldo.

**Solución**:

```cpp
#include <iostream>
#include <string>
using namespace std;

class CuentaBancaria {
private:
    string titular;
    string numeroCuenta;
    double saldo;

public:
    // Constructor
    CuentaBancaria(string tit, string numCuenta, double saldoInicial = 0.0) {
        titular = tit;
        numeroCuenta = numCuenta;
        saldo = (saldoInicial >= 0) ? saldoInicial : 0.0;
    }

    // Método para depositar dinero
    bool depositar(double cantidad) {
        if (cantidad > 0) {
            saldo += cantidad;
            cout << "Depósito exitoso de $" << cantidad << endl;
            return true;
        } else {
            cout << "La cantidad a depositar debe ser positiva." << endl;
            return false;
        }
    }

    // Método para retirar dinero
    bool retirar(double cantidad) {
        if (cantidad > 0 && cantidad <= saldo) {
            saldo -= cantidad;
            cout << "Retiro exitoso de $" << cantidad << endl;
            return true;
        } else if (cantidad > saldo) {
            cout << "Fondos insuficientes. Saldo actual: $" << saldo << endl;
            return false;
        } else {
            cout << "La cantidad a retirar debe ser positiva." << endl;
            return false;
        }
    }

    // Método para consultar saldo
    double consultarSaldo() const {
        return saldo;
    }

    // Método para mostrar información de la cuenta
    void mostrarInfo() const {
        cout << "\n=== INFORMACIÓN DE LA CUENTA ===" << endl;
        cout << "Titular: " << titular << endl;
        cout << "Número de cuenta: " << numeroCuenta << endl;
        cout << "Saldo actual: $" << saldo << endl;
        cout << "================================" << endl;
    }

    // Getters
    string getTitular() const { return titular; }
    string getNumeroCuenta() const { return numeroCuenta; }
```

```
};

int main() {
    string nombre, numeroCuenta;
    double saldoInicial;

    cout << "=== CREACIÓN DE CUENTA BANCARIA ===" << endl;
    cout << "Nombre del titular: ";
    getline(cin, nombre);
    cout << "Número de cuenta: ";
    getline(cin, numeroCuenta);
    cout << "Saldo inicial: $";
    cin >> saldoInicial;

    CuentaBancaria cuenta(nombre, numeroCuenta, saldoInicial);
    cuenta.mostrarInfo();

    int opcion;
    double cantidad;

    do {
        cout << "\n=== MENÚ BANCARIO ===" << endl;
        cout << "1. Depositar" << endl;
        cout << "2. Retirar" << endl;
        cout << "3. Consultar saldo" << endl;
        cout << "4. Mostrar información" << endl;
        cout << "5. Salir" << endl;
        cout << "Opción: ";
        cin >> opcion;

        switch (opcion) {
            case 1:
                cout << "Cantidad a depositar: $";
                cin >> cantidad;
                cuenta.depositar(cantidad);
                break;
            case 2:
                cout << "Cantidad a retirar: $";
                cin >> cantidad;
                cuenta.retirar(cantidad);
                break;
            case 3:
                cout << "Saldo actual: $" << cuenta.consultarSaldo() << endl;
                break;
            case 4:
                cuenta.mostrarInfo();
                break;
            case 5:
                cout << "¡Gracias por usar nuestros servicios!" << endl;
                break;
            default:
                cout << "Opción no válida." << endl;
        }
    } while (opcion != 5);

    return 0;
}
```

## Ejercicio 20: Sistema de Inventario

**Problema**: Crea un sistema básico de inventario con productos.

**Solución**:

```cpp
#include <iostream>
#include <vector>
#include <string>
using namespace std;

class Producto {
private:
    int id;
    string nombre;
    double precio;
    int cantidad;

public:
    // Constructor
    Producto(int identificador, string nom, double pre, int cant) {
        id = identificador;
        nombre = nom;
        precio = (pre >= 0) ? pre : 0.0;
        cantidad = (cant >= 0) ? cant : 0;
    }

    // Métodos para modificar cantidad
    bool agregarStock(int cant) {
        if (cant > 0) {
            cantidad += cant;
            return true;
        }
        return false;
    }

    bool vender(int cant) {
        if (cant > 0 && cant <= cantidad) {
            cantidad -= cant;
            return true;
        }
        return false;
    }

    // Getters
    int getId() const { return id; }
    string getNombre() const { return nombre; }
    double getPrecio() const { return precio; }
    int getCantidad() const { return cantidad; }
    double getValorTotal() const { return precio * cantidad; }

    // Setter para precio
    void setPrecio(double nuevoPrecio) {
        if (nuevoPrecio >= 0) {
            precio = nuevoPrecio;
        }
    }

    // Método para mostrar información
    void mostrarInfo() const {
        cout << "ID: " << id
             << " | Nombre: " << nombre
             << " | Precio: $" << precio
             << " | Cantidad: " << cantidad
             << " | Valor total: $" << getValorTotal() << endl;
    }
};
```

```
class Inventario {
private:
    vector<Producto> productos;

public:
    // Agregar producto
    void agregarProducto(const Producto& producto) {
        productos.push_back(producto);
        cout << "Producto agregado exitosamente." << endl;
    }

    // Buscar producto por ID
    Producto* buscarProducto(int id) {
        for (auto& producto : productos) {
            if (producto.getId() == id) {
                return &producto;
            }
        }
        return nullptr;
    }

    // Mostrar todos los productos
    void mostrarInventario() const {
        if (productos.empty()) {
            cout << "El inventario está vacío." << endl;
            return;
        }

        cout << "\n=== INVENTARIO COMPLETO ===" << endl;
        double valorTotal = 0;

        for (const auto& producto : productos) {
            producto.mostrarInfo();
            valorTotal += producto.getValorTotal();
        }

        cout << "Valor total del inventario: $" << valorTotal << endl;
        cout << "============================" << endl;
    }

    // Realizar venta
    bool realizarVenta(int id, int cantidad) {
        Producto* producto = buscarProducto(id);
        if (producto != nullptr) {
            if (producto->vender(cantidad)) {
                cout << "Venta realizada: " << cantidad << " unidades de "
                     << producto->getNombre() << endl;
                return true;
            } else {
                cout << "No hay suficiente stock. Disponible: "
                     << producto->getCantidad() << endl;
                return false;
            }
        } else {
            cout << "Producto no encontrado." << endl;
            return false;
        }
    }

    // Agregar stock
    bool agregarStock(int id, int cantidad) {
        Producto* producto = buscarProducto(id);
        if (producto != nullptr) {
```

```
            producto->agregarStock(cantidad);
            cout << "Stock agregado exitosamente." << endl;
            return true;
        } else {
            cout << "Producto no encontrado." << endl;
            return false;
        }
    }
};

int main() {
    Inventario inventario;
    int opcion;

    // Agregar algunos productos de ejemplo
    inventario.agregarProducto(Producto(1, "Laptop", 999.99, 10));
    inventario.agregarProducto(Producto(2, "Mouse", 25.50, 50));
    inventario.agregarProducto(Producto(3, "Teclado", 75.00, 30));

    do {
        cout << "\n=== SISTEMA DE INVENTARIO ===" << endl;
        cout << "1. Mostrar inventario" << endl;
        cout << "2. Agregar producto" << endl;
        cout << "3. Realizar venta" << endl;
        cout << "4. Agregar stock" << endl;
        cout << "5. Salir" << endl;
        cout << "Opción: ";
        cin >> opcion;

        switch (opcion) {
            case 1:
                inventario.mostrarInventario();
                break;

            case 2: {
                int id, cantidad;
                string nombre;
                double precio;

                cout << "ID del producto: ";
                cin >> id;
                cout << "Nombre del producto: ";
                cin.ignore();
                getline(cin, nombre);
                cout << "Precio: $";
                cin >> precio;
                cout << "Cantidad inicial: ";
                cin >> cantidad;

                inventario.agregarProducto(Producto(id, nombre, precio, cantidad));
                break;
            }

            case 3: {
                int id, cantidad;
                cout << "ID del producto a vender: ";
                cin >> id;
                cout << "Cantidad a vender: ";
                cin >> cantidad;

                inventario.realizarVenta(id, cantidad);
                break;
            }
```

```
            case 4: {
                int id, cantidad;
                cout << "ID del producto: ";
                cin >> id;
                cout << "Cantidad a agregar: ";
                cin >> cantidad;

                inventario.agregarStock(id, cantidad);
                break;
            }

            case 5:
                cout << "¡Gracias por usar el sistema de inventario!" << endl;
                break;

            default:
                cout << "Opción no válida." << endl;
        }
    } while (opcion != 5);

    return 0;
}
```

## Consejos para la Resolución de Ejercicios

1. **Lee cuidadosamente el problema**: Asegúrate de entender qué se pide antes de comenzar a codificar.
2. **Planifica tu solución**: Divide el problema en pasos más pequeños y manejables.
3. **Usa comentarios**: Explica tu lógica con comentarios en el código.
4. **Prueba con diferentes casos**: No te limites a un solo caso de prueba.
5. **Maneja errores**: Considera casos especiales y valida las entradas del usuario.
6. **Código limpio**: Mantén tu código ordenado y fácil de leer.
7. **Optimización gradual**: Primero haz que funcione, luego optimiza si es necesario.

¡Sigue practicando y mejorando tus habilidades de programación en C++!