Programar en C++

Cabecera de un programa

Librerías Importantes

```
#include <iostream> // Para usar cin, cout
#include <math> // Para usar operadores como sqrt, pow
#include <string> // Para usar strings
#include <vector> // Para usar vectores
#include <algorithm> // Para usar sort
...

#include <bits/stdc++.h> // Para usar todas las librerías
```

Main

```
using namespace std;

int main()
{
    /*Tu código*/
    return 0;
}
```

Entrada/Salida

```
cout << "¿Cómo te llamas?";
string nombre; // Puede ser cualquier tipo de dato
cin >> nombre; // Lee hasta el siguiente espacio
```

Tipos de datos

Nombre	Bytes	Rango
int	4	−2,147,483,648 to 2,147,483,647
unsigned int	4	0 to 4,294,967,295
bool	1	true o false
char	1	−128 to 127
unsigned char	1	0 to 255

Nombre	Bytes	Rango
float	4	3.4E +/- 38 (7 digits)
double	8	1.7E +/- 308 (15 digits)
long	4	−2,147,483,648 to 2,147,483,647
unsigned long	4	0 to 4,294,967,295

IMPORTANTE: El operador de división retorna diferentes valores de acuerdo al tipo de dato

Para tipo int:

```
int n = 9, k =4;
int res = n / k; // res = 2
```

Para tipo float:

```
float n = 9, k=4;
float res = n / k; // res = 2.2
```

Operadores Aritméticos y Lógicos

Operadores Aritméticos

Operación	En C++	Descripción	Asumiendo	Ejemplo

Módulo	`%`	Calcula el residuo de una división de enteros	`// Libreria:`
			`#include <math>`
			
			`// Variables:`
			`int n = 9`
			`int k = 4;`
			`float m = 2.5;`
				`int resto = n % k; // 1`

Potencia	`pow(n,k)`	Calcula n^k		`int potencia = pow(n, k);` `//6561`
Raíz cuadrada	`sqrt(n)`	Calcula $\sqrt{n}$		`int raiz = sqrt(n); // 3`
Incremento	`++`	Suma 1 a la variable		`n++; // n es igual a 10`
Decremento	`--`	Resta 1 a la variable		`k--; // k es igual a 3`
Redondeo	`ceil(m)`	Redondea el valor real al entero siguiente		`int sup = ceil(m); // sup es 3`
	`floor(m)`	Redondea el valor real al entero anterior		`int inf = floor(m); // inf es 2`

Operadores Lógicos

Operación	En C++	Descripción	Asumiendo	Ejemplo

Y	`&&`	true si todas las condiciones son true	`//Variables:`
			
			`bool a = true;`
			`bool b = false;`
			`bool c = true;`
			`bool d = false;`
			
			`int n1 = 51;`
			`int n2 = 36;`
			`int n3 = 51;`
				`bool res1= a && b; // false`
				`bool res1= a && c; // true`

O	`||`	true si alguna condición es true		`bool res1= a || b; // true`
				`bool res1= d || b; // false`

Igualdad	`==`
	`!=`
		Compara igual/distinto		`bool res1= b == d; // true`
				`bool res2= n1 == n3; //false`

Comparación	`<, >, >=, <=`	Comparación aritmética		`bool res1= n1 <= n2;//false`
				`bool res2= n1 >= n3;//true`

Negación	`!`	Invierte el valor de valor		`bool res1= !(n1 >= n3);//false`
				`bool res2 = !a; // false`

Expresiones lógicas	Útiles para describir expresiones complejas. Se usan paréntesis para prioridad de evaluación.		`bool res1 = (n2 < 50)&&(n2 % 6 == 0 || n2 % 5 == 0);`
			`// es n2 menor que 50 y múltiplo de 6 o 5?`
			`bool res2 = a || b || c;`	

Condicionales

Sintaxis

```
if (condición)
{
     /* Código que se ejecuta si se cumple la condición*/
}
```

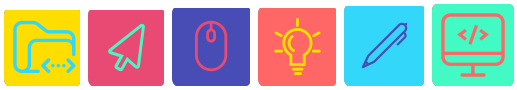
```
if (condición)
{
     /* Código que se ejecuta si se cumple la condición*/
}
else
{
     /* Código que se ejecuta si NO se cumple la condición*/
}
```

```
if (condición1)
{
    /* Código que se ejecuta si se cumple la condición 1*/
}
else if (condición2)
{
    /* Código que se ejecuta si no se cumple la condición 1 pero sí la condición 2*/
}
else
{
    /* Código que se ejecuta si no se cumple ninguna condición*/
}
```

Ejemplo: Determinar si el número ingresado es múltiplo de 7

```
// fragmento de codigo
int num;
cin >> num;
if (num % 7 == 0){
    cout << "Es multiplo de 7";
}
else{
    cout << "No es múltiplo de 7";
}
```

Notas Personales



Iteraciones

For

```
for (int i=(valor inicial); i<=(valor final); i=i+paso)
{
 ... bloque de instrucciones ...
}
```

While

```
while (condición)
{
 ... bloque de instrucciones ...
}
```

Ejemplo: Determinar si el número ingresado es primo

```
// fragmento de codigo
int num;
cin >> num;
int divisores = 0;
for (int i=2; i<=num/2; i=i+1){
    if (num % i == 0){
        divisores++;
    }
}
if (divisores == 0){
    cout << "Es primo";
}
else{
    cout << "No es primo";
}
```

Strings

En C++	Descripción	Asumiendo	Ejemplo

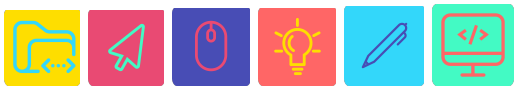
+	Concatenar	`// Biblioteca:`
		`#include <string>`
		
		`// Variables:`
		`string var1 = "alpha";`
		`string var2 = "beta";`
			`string var3 = var1 + var2;`
			`//alphabeta`

==, !=	Comparar		`bool valor = (var1==var2); // false`
			`bool valor = (var1!=var2); // true`

size	Nos entrega el tamaño		`int tamano = var1.size(); // 5`

substr	Genera un substring		`string var3 = var1.substr(1,3);`
			`// var3 = "lph", porque pedimos 3`
			`// caracteres desde la posición 1`

find	Entrega la posición donde se encuentra un contenido		`int pos = var1.find("l");`
			`// 1, dado que es el caracter de esa`
			`// posición`



Arreglos, Vectores y Matrices

Arreglos

```
int edades [4]; // Si conoces el tamaño, pero no sabes exactamente qué datos contiene
int edades [] = {45, 21, 33, 1}; // Si conoces tus datos, puedes ponerlos directamente
int elemento_pos_3 = numeros[3]; // Para acceder al elemento 3, en este caso es 1
edades[0] = 3; // ahora el valor en la posición 0 del array edades es 3
```

Vectores

```
// crear un vector
vector <int> edades;
// También puedes poner datos directamente
vector <int> edades = {45, 21, 33, 1};
int elemento_pos_0 = numeros[0]; // Para acceder al elemento 0, en este caso es 45
edades[0] = 3; // ahora el valor en la posición 0 del vector edades es 3
```

NOTA: La principal diferencia entre vectores y arreglos es que podemos crear vectores sin un tamaño definido y va a ir creciendo según le vamos agregando valores. Además posee funciones útiles.

Funciones útiles con vectores

En C++	Descripción	Asumiendo	Ejemplo

front	Nos entrega el primer elemento	`// Libreria:`
		`#include <vector>`
		
		`// Variables:`
		`vector notas = {43, 42, 67, 55};`
			`cout<< notas.front(); // 43`

back	Nos entrega el último elemento		`cout<< notas.back(); // 55`
clear	Borra el contenido del vector, lo deja de tamaño cero		`notas.clear();`
size	Entrega el tamaño del vector		`cout<< notas.size(); // 4`
empty	Retorna true si el vector está vacío		`if (notas.empty()) { ... }`
push_back	Agrega un elemento al final		`notas.push_back(66);`
pop_back	Elimina el último elemento		`notas.pop_back();`

Ejemplo con vectores:

Para ordenar un vector, de menor a mayor:

```
#include <algorithm>

// fragmento de código
vector <int> edades = {45, 21, 33, 1};
sort(edades.begin(), edades.end());

// El vector edades queda
// {1, 21, 33, 45}
```

Recibir n números decimales y guardarlos en un vector

```cpp
#include <bits/stdc++.h>
int main(){
    int n; // variable para cantidad de decimales
    std::cin >> n; // recibir valor para n
    // variable para el vector de decimales
    std::vector<float> miVector;
    float x; // variable para dejar un decimal
    for(int i = 0; i < n; i++){
        std::cin >> x; // Recibir un decimal
        miVector.push_back(x); // agregar x al final
    }
}
```

Matrices

```cpp
// Si conoces el tamaño, pero no sabes exactamente qué datos contiene
float precios [3][3]; // con un vector sería vector<vector<double>> precios
int calendario [5][7][12];

// Si conoces tus datos, puedes ponerlos directamente
float precios[3][3] = {
{1.5, 1.7, 1.2},
{0.7, 2.3, 3.6},
{1.0, 0.5, 2.1}
};
```

Tips para resolver problemas

Mientras leemos el problema planteado debemos interiorizar:

¿Cuál es la pregunta que quiero resolver?
¡Evitar distracciones dentro del enunciado!

Cuando empezamos a analizar el problema debemos plantearnos:

¿Qué datos son útiles para poder responder la pregunta?
¿Qué tipo de datos podríamos usar? (son valores int? float? string?)

Antes de empezar a programar debemos responder:

¿Cuáles son los pasos principales que debiera tener la solución?
¿Necesitamos cumplir con condiciones especiales? (podemos usar if, if...else?)
¿Alguna parte del problema requiere repetir algunos pasos? (podemos usar for, while?)
¿Necesito guardar los datos en un vector o matriz o los puedo procesar a medida que el usuario los va ingresando?

Notas Personales