PROGRAMACIÓN I
C++

UCM

Grado en Estadística Aplicada. EUE.

Tema 3.- Programación estructurada

Isabel Riomoros

Programación estructurada

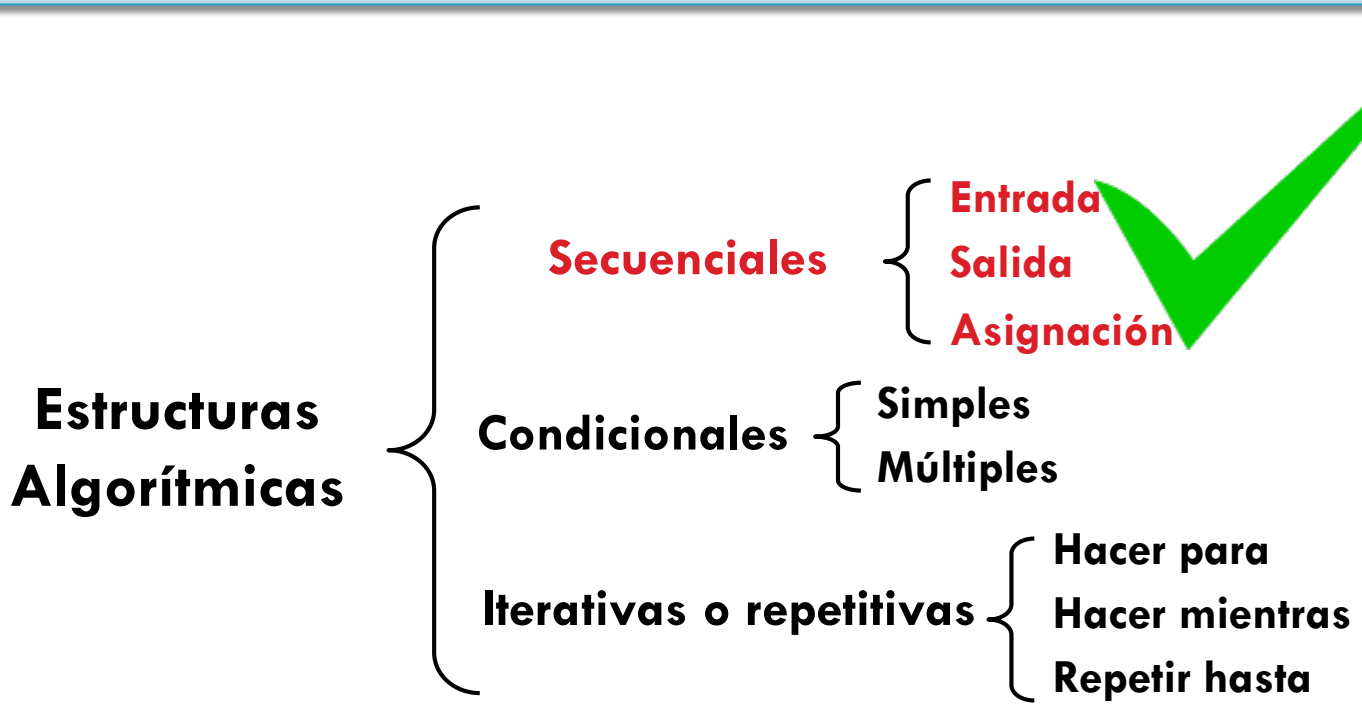
TEMA 3

- Sentencias condicionales
- Sentencias iterativas o repetitivas

C++

Introducción

programas lineales - comienzan por la primera Sentencia y acababan por la última, ejecutándose todas una sola vez.



C++

Introducción

cout ,cin, asignación

- Sólo con estas sentencias podemos resolver problemas sencillos.

Sentencias IF y SWITCH

- Cuando nos interesa que dependiendo de los valores de los datos, se ejecuten unas sentencias u otras →**sentencias condicionales**.

Sentencias WHILE, DO-WHILE y FOR

- A veces nos interesará repetir una sentencia ó sentencias un número determinado de veces→ **sentencias de control iterativas ó repetitivas** (ciclos o bucles).

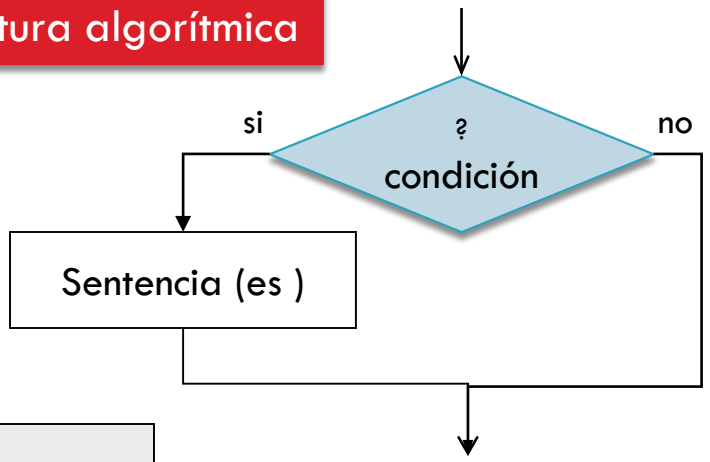
C++

Sentencias condicionales

IF, IF-ELSE, SWITCH

Sentencia condicional simple: IF

estructura algorítmica



El formato general de una sentencia **if** es la siguiente:

```
if (condición)
   {
      Sentencia 1;
      …
      Sentencia n;
   }
```

¡¡¡OJO!!!

```
if (condición)
      Sentencia;
```

Si se cumple la condición, entonces se ejecuta la *Sentencia* ó el *bloque de Sentencias*; en caso contrario, no se ejecutan.

Sentencia condicional simple: IF


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

int main()
{
        int a;
        cin >> a;

        if (a<0)
            a = -a;

        cout << a;
        return 0;
}
```

Valor absoluto

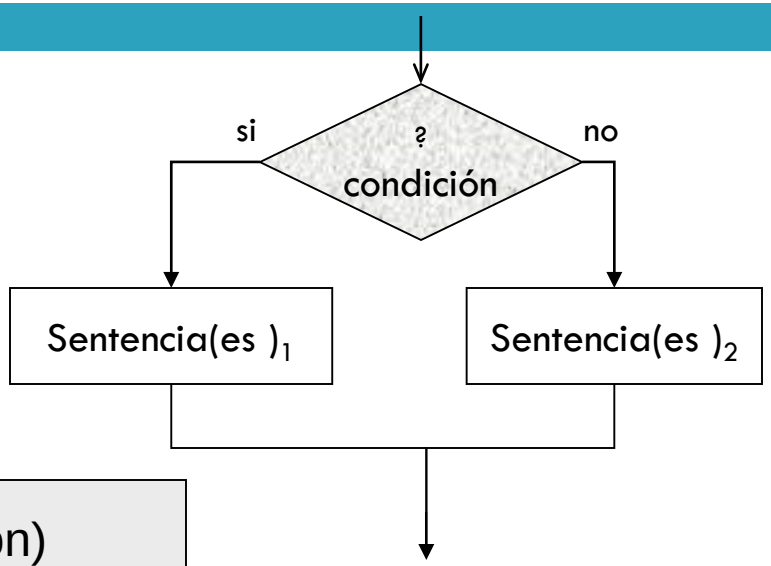
```
#include <iostream>
using namespace std;

int main()
{
        ...

        if (cantidad > 3)
        {
            descuento  = 0.2;
            precio = n*descuento;
         }

        ...
        return 0;
}
```

Descuento en un producto si compras más de tres unidades

C++

Sentencia condicional doble : IF-ELSE

Estructura algorítmica asociada:



sentencia **if-else**

```
if  (condición)
    {
       Sentencias 1;
    }
else
    {
        Sentencias 2;
    }
```

```
if  (condición)
       Sentencia 1;
else
       Sentencia 2;
```

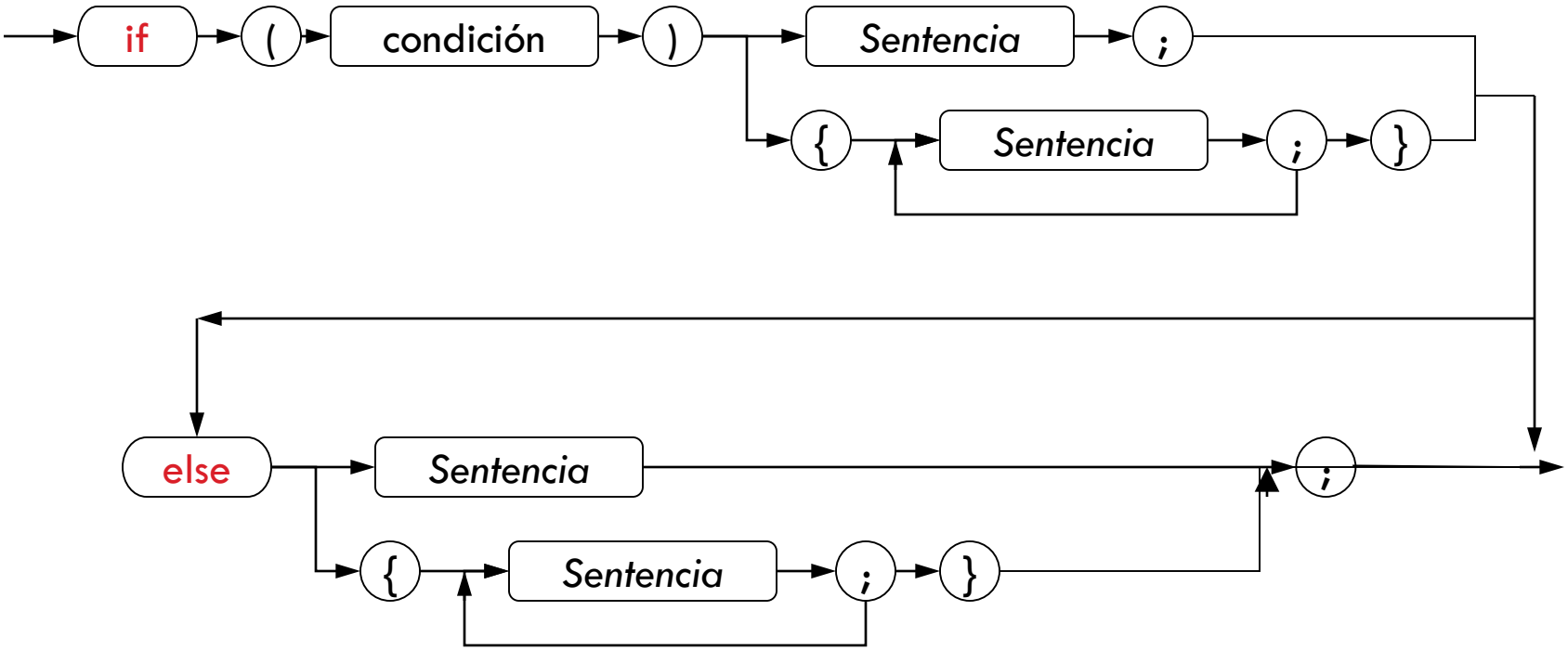
Si se cumple la condición, se ejecutan las Sentencias del primer bloque;

si no, se ejecutan las Sentencias del segundo bloque.

C++

Sentencia condicional IF, IF-ELSE

Diagrama sintáctico:



C++

Sentencia condicional doble : IF-ELSE

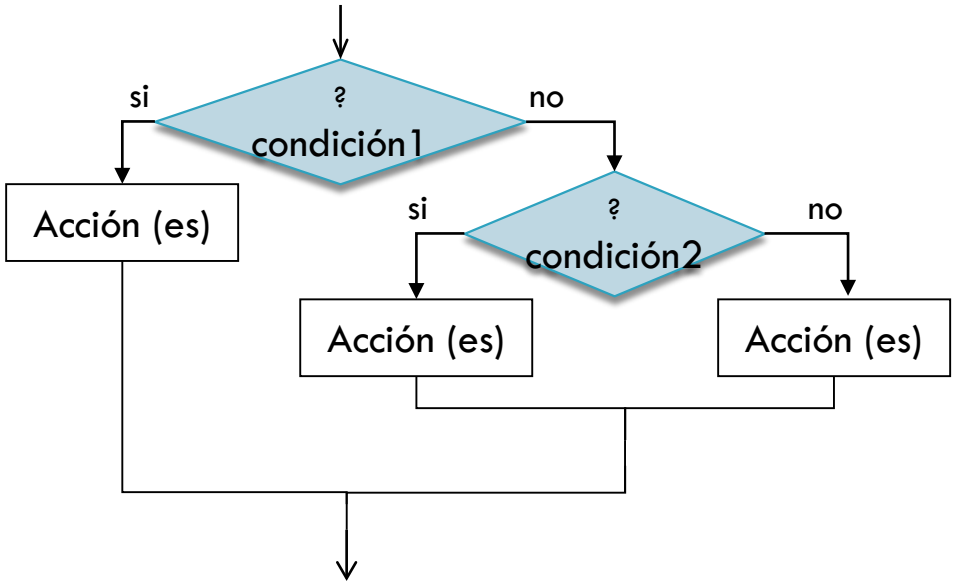

```
#include <iostream>
int main()
{
        int a, x;
        cin >> a;

        if (a==0)
                x = 1;
        else
                x= 0;

        cout << x;
        return 0;
}
```

```
#include <iostream>
int main()
{
        ...

        if (cantidad > 3)
        {
           descuento  = 0.2;
           precio = n*descuento;
         }
        else
           precio = n;

        ...
        return 0;
}
```

C++

Sentencia condicional doble : IF-ELSE anidadas

- Podemos utilizar las sentencias IF-ELSE anidadas, es decir, que alguna de las ramas sea a su vez otra Sentencia IF-ELSE.



C++

Sentencia condicional doble : IF-ELSE anidadas

Sentencias IF-ELSE anidadas

```
if  (condición1)
    Sentencia 1;
else
    if  (condición2)
            Sentencia 2;
    else
            Sentencia 3;
```

```
if  (condición1)
     Sentencia 1;
else
    if  (condición2)
            Sentencia 2;
    else
          if  (condición3)
               Sentencia 3;
         else
               Sentencia 4;
```

C++

Ejercicios

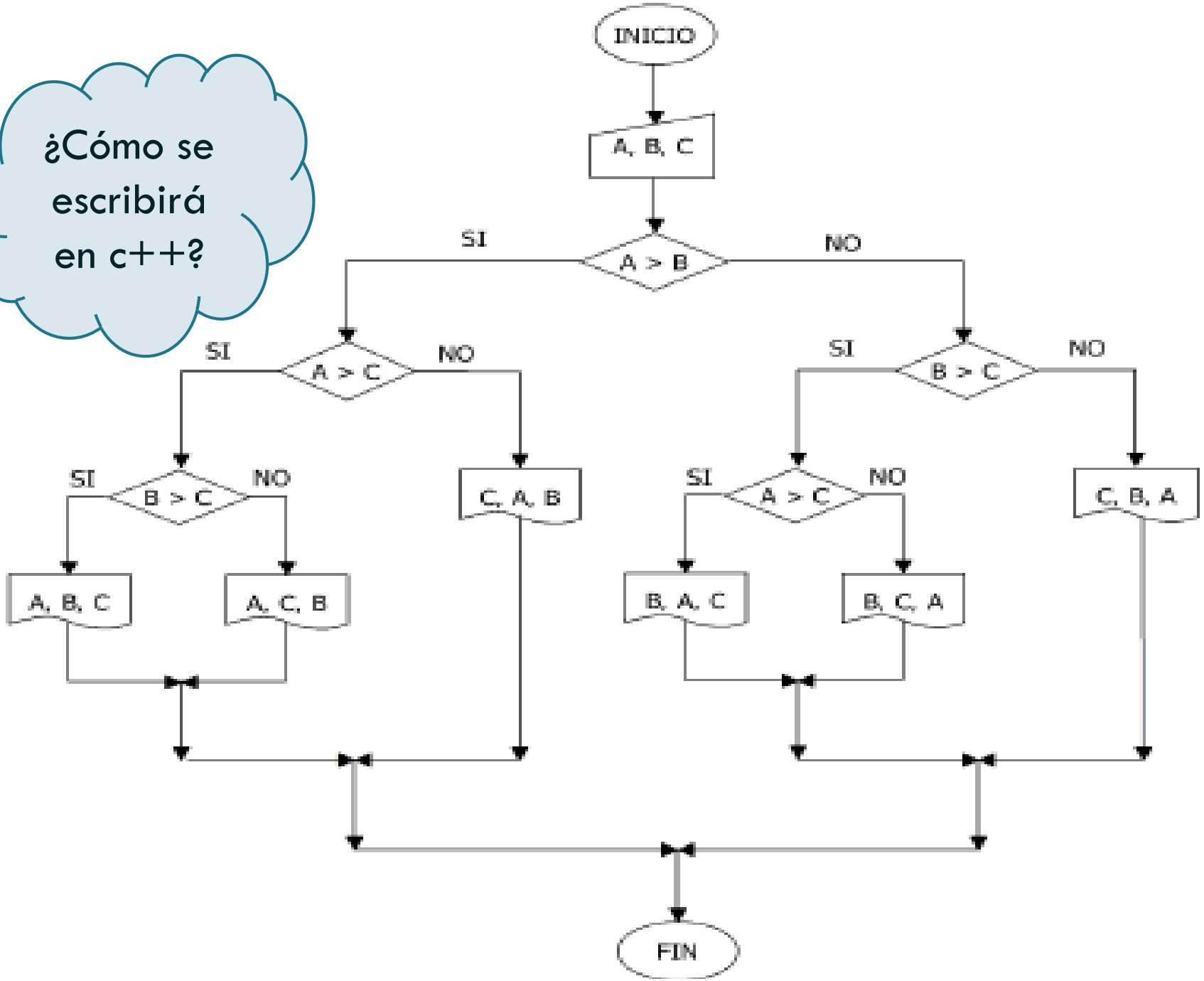

```
if(b!=0) if (b+3>a) sentencia0;
else if(a+b<c) sentencia1; else if (b<10) sentencia2;
else {sentencia3; if(a-b>x) sentencia4; } else sentencia5;
```

```
If (a>=0)
 if (a<=5)
 cout<<"a está entre 0 y 5";
 else cout<<"a debe ser menor que cero";
```

Si a toma un valor menor que 0 ¿qué pasará?

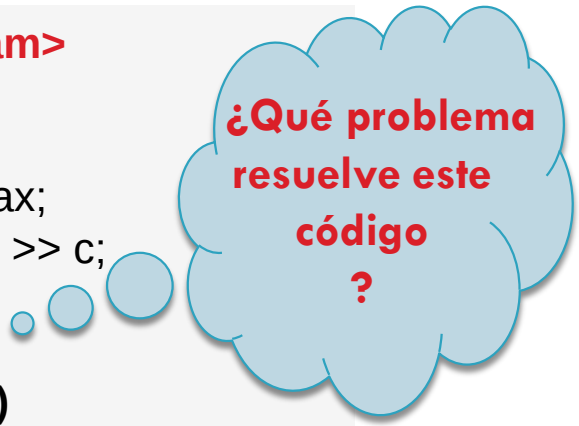
C++

¿Cómo se escribirá en c++?



Sentencias Condicionales


```
#include <iostream>
int main()
{
        int a, b, c, max;
        cin >> a >> b >> c;

        if (a > b)
                if (a > c)
                        cout << a;
                else
                        cout << c;
        else
                if (b > c)
                    cout << b;
                else
                    cout << c;

        return 0;
}
```




```
if (Nota == 10)
cout << "MH";
else
if (Nota >= 9)
cout << "Sobresaliente";
else
if (Nota >= 7)
cout << "Notable";
else
if (Nota >= 5)
cout << "Aprobado";
else
cout << "Suspenso";
```

C++

Ver si un número es primo sin ciclos


```
#include<iostream>
using namespace std;
int main()
{
    int n;
    cout << "Introduce un numero: "; cin >> n;
    if(n!=2 & n%2==0)
        cout << "No es primo";
    else
    {
        if(n!=3 & n%3==0)
            cout << "No es primo";
        else
        {
            if(n!=5 & n%5==0)
                cout << "No es primo";
            else
            {
if(n!=7 & n%7==0)
                cout << "No es primo";
  else
            {
                if(n!=11 & n%11==0)
                    cout << "No es primo";
                else
                {
                    cout << "Es primo";
                }
            }
        }
    }
  }
  system("pause");
  return 0;
}
```

es infalible hasta el 168

C++

Problemas clase

- Dado el sueldo de un trabajador, aplicar un aumento del 20 % si el sueldo es inferior a 600 euros.
- Determina si un entero es par y en caso de no serlo, determinar si es divisible por 3
- Leer tres números enteros de un supuesto triángulo, determinar si realmente forman un triángulo, y mostrar el tipo de triángulo que es (si es un triángulo).
  - ***triángulo*:** La suma de dos cualesquiera de los lados debe ser mayor que el otro.
  - ***equilátero***: todos los lados son iguales.
  - ***isósceles***: solo dos lados son iguales.
  - ***escaleno*:** no tiene dos lados iguales.
- Dados los goles del equipo local (golLocal) y los goles del equipo visitante (golvisita), devuelva una cadena de caracteres indicando qué equipo ha ganado (local,visitante, empate).
- Dado un número de mes escribir el nombre

C++

Calcular el área de diferentes polígonos

¿Qué área quieres calcular?

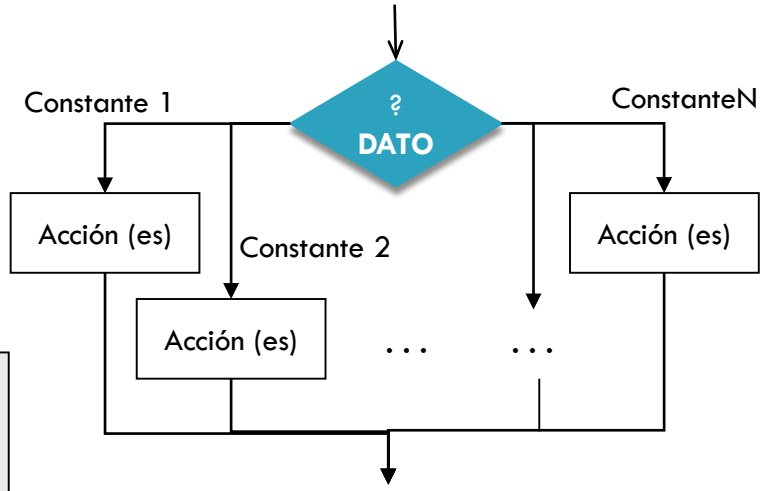
1. cuadrado
2. rectángulo
3. Triángulo

4. Fin

C++

Sentencia condicional múltiple : SWITCH

Estructura algorítmica



sentencia witch

```
switch  (selector)
   {
      case constante1:
         Sentencia1 ó bloque de Sentencias
         break;
      case constante2:
         Sentencia2 ó bloque de Sentencias
         break;
     defeault:
         Sentencia2 ó bloque de Sentencias
   }
```

Permiten comparar una 'variable' con distintos valores posibles, ejecutando para cada caso una serie de Sentencias específicas.

C++

Sentencia condicional múltiple : SWITCH

El valor de ***selector*** debe ser un **número entero**. Puede ser una variable, una expresión ó una llamada a una función.

sentencia switch

```
switch  (selector)
   {
      case constante1:
            Sentencia1 ó bloque de Sentencias
            break;
      case constante2:
            Sentencia2 ó bloque de Sentencias
            break;
     defeault:
            Sentencia3 ó bloque de Sentencias
   }
```

Cada caso comienza con un **case** y termina con un **break**

¿Qué ocurre si se me olvida algún break ?

C++

Sentencia condicional múltiple : SWITCH

Ejemplo

```
#include <iostream>
int main ()
{
  cin>> num;
  switch  (num)
     {
       case 1:
              cout << "Ha introducido el nº 1\n";
       case 2:
              cout << "Ha introducido el nº 2\n";
              break;
      defeault:
               cout << "Ha introducido otro nº";
     }
  return 0;
}
```

Si al ejecutar el programa introducimos un 2, obtenemos el mensaje:

'Ha introducido el nº 2'

Si al ejecutar el programa introducimos un 1, obtenemos el mensaje:

'Ha introducido el nº 1'

'Ha introducido el nº 2'

C++

Dada una nota escribir la calificación


```
#include<iostream>
using namespace std;
int main() {
	int nota;
	cout<<" Inserte una nota: ";
	cin>>nota;
	switch(nota) {
		case 0:	cout<<"\nSuspenso"; break;
		case 1:	cout<<"\nSuspenso"; break;
		case 2:	cout<<"\nSuspenso"; break;
		case 3:	cout<<"\nSuspenso"; break;
		case 4:	cout<<"\nSuspenso"; break;
		case 5:	cout<<"\nAprobado"; break;
		case 6:	cout<<"\nBien"; break;
		case 7:	cout<<"\nNotable"; break;
		case 8:	cout<<"\nNotable"; break;
		case 9:	cout<<"\nSobresaliente"; break;
		case 10:	cout<<"\nSobresaliente"; break;
		default: cout<< "esa nota es incorrecta";
	}
	return 0;
}
```

C++

Pasar de euros a pts o de Pts a euros


```
const  float EURO= 166.386;
float n,x; int opcion;
        cout<<"la cantidad: ";
        cin>>n;
        cout<<"1-Ptas a Euros 2-Euros a ptas";
        cin>>opcion;
        switch(opcion) {
              case 1: {
                      x=n/EURO;
                      cout<<n<<" Pesetas son "<< x<<" Euros"; break;
                      }
              case 2: {
                      x=n*EURO;
                      cout<<n<<" Euros son "<<x<<" Pesetas"; break;
                     }
              default:  cout<<"incorrecta";
        }
```

C++

Errores difíciles de encontrar

- En la condición de un if poner un = en vez de ==, no da error y considera la expresión siempre cierta, salvo que sea = 0 que la expresión la considerará falsa
  - If x<y<z **NO**
  - If ((x<y) &&(z<y)) **SI**

C++

Ejercicios

Programa que calcule el índice de masa corporal de una persona (IMC = peso [kg] / altura² [m]) e indique el estado en el que se encuentra esa persona en función del valor de IMC:

Valor de IMC	Diagnóstico
< 16	Criterio de ingreso en hospital
de 16 a 17	infrapeso
de 17 a 18	bajo peso
de 18 a 25	peso normal (saludable)
de 25 a 30	sobrepeso (obesidad de grado I)
de 30 a 35	sobrepeso crónico (obesidad de grado II)
de 35 a 40	obesidad premórbida (obesidad de grado III)
>40	obesidad mórbida (obesidad de grado IV)

C++

Ejercicios

En el Supermercado CompraMix se ofrece la promoción de llevarse tres productos por el precio de los dos más baratas. Escribir un programa que, dados los tres precios de los productos, determine la cantidad a pagar.

C++

Ejercicios

Programa que simule el funcionamiento de una calculadora que puede realizar las cuatro operaciones aritméticas básicas (suma, resta, producto y división) con valores numéricos enteros. El usuario debe especificar la operación con el primer carácter del primer parámetro de la línea de comandos: S o s para la suma, R o r para la resta, P, p, M o m para el producto y D o d para la división. Solicitar al usuario los dos enteros y la operación a realizar.

C++

Sentencias o sentencias repetitivas o iterativas

WHILE, DO-WHILE, FOR

Sentencias Iterativas o repetitivas

- Controlan la repetición de un conjunto de Sentencias denominado bloque o *cuerpo del bucle*, mediante la evaluación de una condición o mediante un contador.

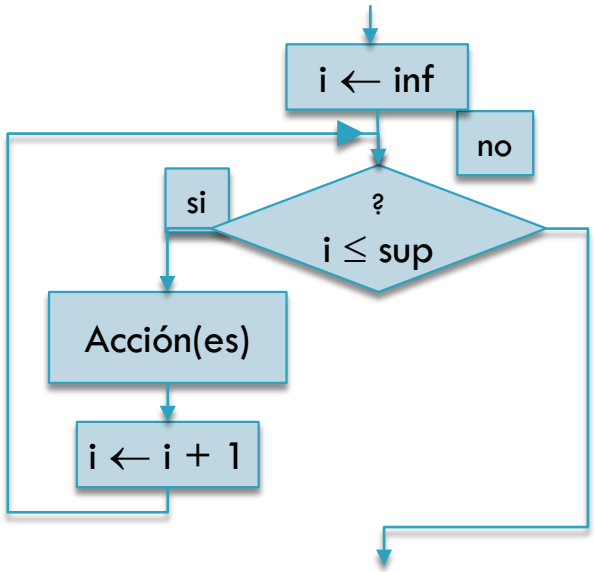
Sentencias
WHILE, DO-WHILE y FOR

C++

Sentencia de control repetitiva : FOR

Se utiliza para ejecutar un bloque de **Sentencias un número fijo de veces** que se conoce de antemano.

estructura algorítmica *hacer_para:*



sentencia for

```
for  ( inicialización ; condición ; actualización )
 {
      Sentencia 1;
            …
      Sentencia n;
 }
```

C++

Sentencia de control repetitiva : FOR

La Sentencia **for** se vale de una **variable de control del ciclo**. El ciclo se repite desde el límite inferior, hasta que la variable de control llegue la límite superior.

```
for  ( inicialización; condición; actualización )
 {
     Sentencia 1;
          …
     Sentencia n;
 }
```

Cabecera del **for**

C++

Sentencia de control repetitiva : FOR

En la cabecera del **for** nos podemos encontrar con 3 partes o secciones

for (inicialización; condición; actualización)

- ***Parte de inicialización:*** Inicializa la variable de control del bucle. Se pueden utilizar una o varias variables.
- ***Parte de condición:*** Expresión lógica. El cuerpo del bucle se repite mientras la expresión sea verdadera.
- ***Parte de actualización:*** Incrementa o decrementa el valor de la variable o variables de control.

```
int main()
{
        ....
        for (int i = 0; i<10; i++)
        {
            cout << "Número: " << i  ;
            cout << endl;
        }
         ....
        return 0;
}
```

C++

¿Cuál será la salida?


```
int main() {
 {
  for(int i=0; i<10; i++) {
     cout << "Dentro del bucle, i = " << i << endl;
  }
  cout << i << endl;
 }
```


```
int main() {
 {
  for(int i=0; i<10; i++) {
     cout << "Dentro del bucle, i = " << i << endl;
  }
  cout << “ADIÓS” << endl;
 }
```

C++

Sentencia de control repetitiva : FOR


```
int main()
{
      ….
   for (int n = 1; n<=10; n=n+2)
   {
      cout << “Número: ” << n <<“\t” << n*n  ;
      cout <<< endl;
    }
   ….
    return 0;
}
```

La variable de control es *n*

Es de tipo entero, su valor inicial es 1

Su valor final es 10

Se incrementa de 2 en 2.

Pasada	Valor de *n*
1	1
2	3
3	5
4	7
5	9

En la primera iteración ó pasada, el valor de n es 1, en la segunda pasada n vale 3. La última pasada se realiza cuando n vale 9.

C++

Sentencia de control repetitiva : FOR

Se puede inicializar más de una variable, se puede poner más de una condición ó más de un incremento.

```
int main()
{
      ….
      for ( int i = 0, j = 100;  i<j ;  i++, j--)
      {
            int k;
            k =  i+2*j ;
             cout << k << endl;
      }
       ….
      return 0;
}
```

Se declaran 2 variables de control: *i* y *j*

Se inicializan a 0 y 100 respectivamente

El cuerpo se ejecutará mientras *i* sea menor que *j*

Se incrementa la variable i y se decrementa la variable j

C++

Sentencia de control repetitiva : FOR


```
int main()
{
      ….
       for ( int i = 0, j = 5;  i+j <100 ;  i++, j =j+5 )
       {
              cout << i <<"\t" <<j << (i+j) ;
              cout << endl;
       }
        ….
       return 0;
}
```

Valor de las variables de control en cada pasada

valor de *i*	valor de *j*	*i+j*
0	5	5
1	10	11
2	15	17
3	20	23
4	25	29
5	30	35
6	35	41
7	40	47
8	45	53
9	50	59
10	55	65
11	60	71
12	65	77
13	70	83
14	75	89
15	80	95

Se realizan un total de 16 pasadas.

Se ejecuta el cuerpo de la Sentencia **for** mientras se cumpla la condición.

C++

Sentencia de control repetitiva : FOR

Se puede omitir cualquiera de las partes de la cabecera

```
int main()
{
      ....
       contador = 1;
       for ( ; contador <10  ; )
       {
              cout << contador ;
              contador++ ;
       }
        ....
       return 0;
}
```

```
int main()
{
      ....
      for ( ;  ; )
       {
              ...
       }
        ....
       return 0;
}
```

Bucle infinito

Bucle infinito que espera la pulsación de una tecla

-
```
for( ; ; ) {
   if(getch()!=0) break;
}
```

C++

Ejemplo


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

int main()
 {
    int numero, acumulador=0, contador=0;
    cin>>numero;
    for(int i=numero; i>0; i--)
    {
      acumulador+= numero;
      contador++;
      numero--;
     }
   cout<<acumulador;
    system("PAUSE");
    return 0;
 }
```

C++

¿Cuál será la salida?


```
for(int i=1;i<= impar ;i=i+2)
   {
    cout<<setw(MARGEN)<<" ";
    for(int j=1; j<=i; j++)
      cout<<"*";
    cout<<endl;

   }
   for(int i=impar-2;i>=1 ;i=i-2)
   {
    cout<<setw(MARGEN)<<" ";
    for(int j=1; j<=i; j++)
      cout<<"*";
    cout<<endl;
   }
```

C++

Ejercicios

Múltiplos de 3 menores o iguales a 20

Leer 50 caracteres del input y hacer un resumen de las letras minúsculas, mayúsculas, dígitos y otros caracteres

N primeros términos de la sucesión de Fibonacci

Potencia, x^n

Factorial de un número n!= n*(n-1)*(n-2)*(n-3)* ….. 3*2*1

Escribe un programa que calcule la suma de todos los números entre 1 y 100 que son múltiplos de 5 o de 7.

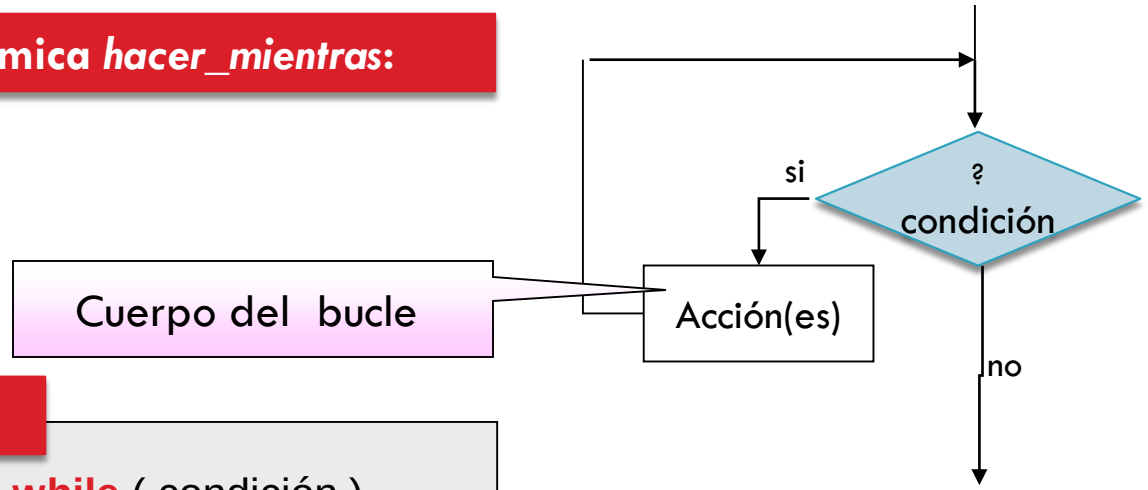
Escribir un programa que calcule el sumatorio $\sum_{i=0}^{10} i^3$

Tablas de multiplicar

C++

Sentencia de control repetitiva : WHILE

estructura algorítmica *hacer_mientras*:



sentencia while

```
while ( condición )
  {
     Sentencia 1;
            …
     Sentencia n;
  }
```

```
while ( condición )
     Sentencia;
```

Cada repetición del cuerpo del bucle se denomina *iteración*

Las Sentencias se ejecutan mientras se cumpla la condición, que será evaluada siempre **antes** de cada repetición.

C++

Sentencia de control repetitiva : WHILE


```
#include <iostream>
using namespace std;
int main()
{
        …
   contador = 0;
   while (contador < 100)
    {
       cout << “Hola”;
       contador ++;
    }
    ...

   return 0;
}
```

cuerpo

La condición será evaluada siempre antes de cada iteración.

El cuerpo del bucle while se repite mientras la condición sea cierta.

El cuerpo de un bucle while se ejecutará **cero o más veces.**

C++

Sentencia de control repetitiva : WHILE

Inicialización
Se realiza antes de la Sentencia **while**

```
...
contador = 0;
while (contador < 100)
 {
    cout << "Hola";
    contador ++;
 }
...
```

cuerpo

Actualización
Se realiza dentro del cuerpo del bucle durante cada iteración

Comprobación
Se comprueba el valor de la variable antes de comenzar la repetición

La variable que representa la condición del bucle se denomina ***variable de control del bucle***.
Dicha variable debe ser:

- inicializada
- comprobada
- actualizada

C++

Escribir Hola cont-veces


```
#include <iostream>
using namespace std;
int main()
{
  int cont;
  cout<<"Numero de veces a repetir hola"<<endl;
  cin>>cont;
  while (cont >0)
   {
     cout<<"Hola ";
     cont=cont-1;
    }
   system("pause");
   return 0;
}
```

C++

Calcular la media de n-números


```
#include <iostream>
using namespace std;
int main()
{
  float num, media,suma=0;
  int n, cont=0;
  cout<<" Cantidad de números a sumar "<<endl;
  cin>>n;
  while (cont<n)
   {
     cout<<"Dime un numero "<<endl;
     cin>>num;
     suma=suma+num;
     cont=cont+1;
    }
   media=suma/n;
  cout<<"MEDIA= "<<media<<endl;
  system("pause");
  return 0;
}
```

C++

Raíz cuadrada de un número dado


```
#include <iostream>
#include <stdlib.h>
using namespace std;
int main()
{
  int raiz=0,num;
  cin>>num;
  while( (raiz+1)*(raiz+1) <= num) raiz ++;
  cout<<raiz;
  system("pause");
  return 0;
}
```

C++

Sentencia de control repetitiva : WHILE


```
int main()
{
        const bool continuar = true;
        bool a = true;
        int n;
        while (a ==continuar)
        {
                cin >> n;
                if (n<0)
                  {
                     cout << "No se admiten negativos";
                     a = false;
                  }
                else
                  cout << “Uno más: " ;
        }
        return 0;
}
```

Inicialización

Comprobación

Es importante comprobar que en algún momento, la condición del bucle se hace falsa, de forma que no obtengamos bucles infinitos.

Actualización

C++

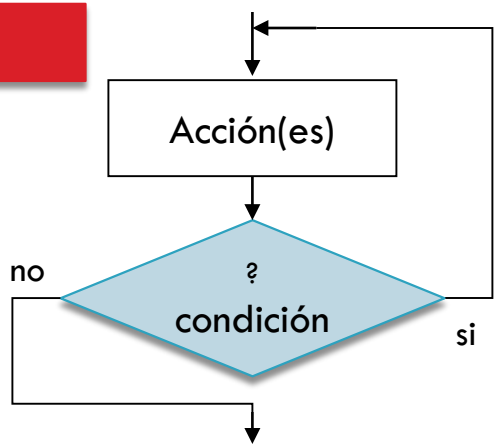
Euros en monedas de 10, 5, 2, 1, 0.50


```
#include<iostream>
using namespace std;
 int main()
{
   float cambio; int cambioint, m10=0, m5=0, m2=0,
     m1=0, m50c=0;
   do
   {
      cout << "Cambio?: "; cin >> cambio;
      cambioint = (int)cambio;
   }while((cambio - cambioint) != 0 && (cambio -
     cambioint) != 0.50);
   while(cambio != 0)
   {
      if(cambio>=10)
      {
         m10++;
         cambio-=10;
      }
      else if(cambio>=5)
      {
         m5++;
         cambio-=5;
      }
      else if(cambio>=2)
      {
         m2++;
         cambio-=2;
      }
      else if(cambio>=1)
      {
         m1++;
         cambio-=1;
      }
      else if(cambio>=0.5)
      {
         m50c++;
         cambio-=0.5;
      }
   }
   cout << m10 << ", " << m5 << ", " << m2 << ", " <<
     m1 << ", " << m50c;
   system("pause");
   return 0;
}
```

C++

Sentencia de control repetitiva : DO-WHILE

estructura algorítmica



sentencia do-while

```
do
{
      Sentencia 1;
            …
      Sentencia n;

 } while ( condición );
```

```
do
   Sentencia;
while ( condición );
```

- Ejecuta un bloque de Sentencias al menos una vez.
- El cuerpo del bucle **se repite mientras se verifica la condición**.
- La condición será evaluada **después** de cada repetición.

C++

Sentencia de control repetitiva : DO-WHILE


```
#include <iostream>
int main()
{
        char n;
        do
        {
            cout << "introducir número: ";
            cin >> n;
            if ((isdigit(n))== false)
                cout << ”Solo se admiten números";

        } while ((isdigit(n))  !=  false);

        return 0;
}
```

cuerpo

El cuerpo de un bucle **do-while** se ejecutará una o más veces.

El cuerpo del bucle **do-while** se repite mientras la condición sea cierta.

La condición será evaluada siempre después de cada iteración.

C++

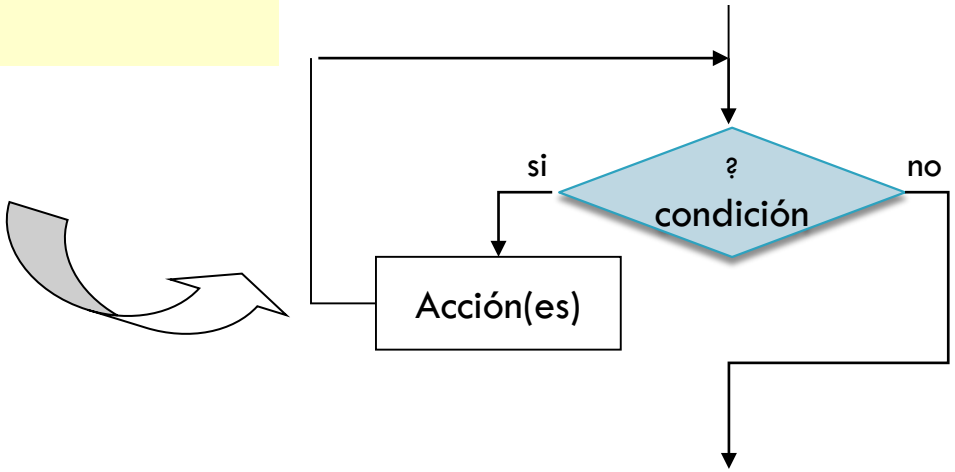
Sumar números hasta que introduzca un cero

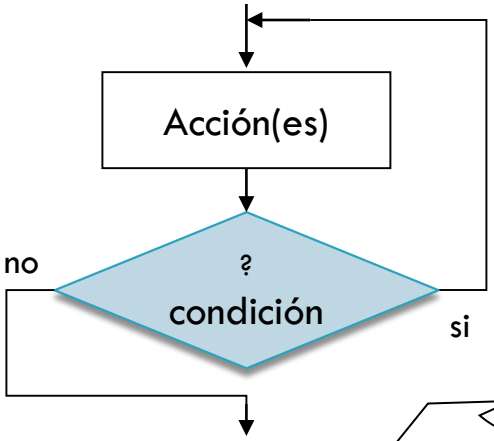

```
#include <iostream>
using namespace std;
int main() {
	int num=0,suma=0;
	do {
		suma=suma+num;
		cout<<"un número: ";
		cin>>num;
	} while(num>=0);
	cout<<"suma es: "<<suma;
	system("pause");
	return 0;
}
```

C++

Diferencias entre las Sentencias WHILE y DO-WHILE

En **while** primero se tiene que cumplir la condición, y luego se ejecuta el bloque de Sentencias.





En **do-while** primero se ejecuta el bloque de Sentencias y luego se comprueba la condición.

C++

Traducción de WHILE a DO-WHILE


```
while ( condición )
{
    Sentencia 1;
        …
    Sentencia n;
}
```

```
if (condición)
    do
    {
        Sentencia 1;
            …
        Sentencia n;
    } while ( condición );
```

C++

Errores comunes en los bucles

Cuerpo del bucle vacío

```
while (num !=0);
{ cout<<"hola"<<num;
   cin>>num;
}

for (i=0 ; i <=10; i++);
{  cubo = i * i * i;
    suma = suma + cubo;
}
```

Bucles que nunca acaban

```
for (i=0 ; i >=0; i++)
{   cubo = i * i * i;
   suma = suma + cubo;
}
```

Bucles donde nunca se ejecuta el cuerpo del bucle.

```
for (i=0 ; i >= 10; i++)
{   cubo = i * i * i;
   suma = suma + cubo;
}
```

Una iteración de más o una iteración de menos

p.e. cálculo de la potencia

C++

Algoritmo ruso


```
#include <iostream>
//Algoritmo ruso de la multiplicación
using namespace std;

int main()
{
  int m,n;
  do {
      cout<<"Dame dos números"<<endl;
      cin>> n >>m;
      } while ((n<0)||(m<0));
  int x=0;
  while (m!=0){
        if (m % 2 !=0) x=x+n;
        n=n+n;
        m=m/2;
  }
  cout<<"el resultado es " << x <<endl;
  system("pause");
  return (0);
}
```

Se parte de dos enteros x e y. Sucesivamente uno se va multiplicando por 2 y el otro se va dividiendo por 2. Cuando la división (entera) de como resto 1, el entero que va a multiplicarse se añade a un acumulador. El proceso se repite hasta que el entero que se divide vale cero.

C++

Controlar el fin de línea

- c=**cin.peek**(); librería iostream
- Devuelve el carácter leído o EOF en caso de haber alcanzado el fin de archivo.
- **Nota:** No retira el carácter del flujo de entrada.

```
while((cin.peek())!= '\n')
```

También podríamos hacerlo con:

`while (cin>>num)` al final de la entrada de datos tenemos que dar CTRL+Z

C++

Dada una secuencia de enteros acabada en 0, diseña un programa que cuente cuantas veces aparece el primero


```
#include <iostream>
int main() {
    int num, primero, total;
    cout << "Introduzca una serie de numeros: ";
    cin >> num;
    primero = num;
    total = 1;

    while (num != 0)
    {
        cin >> num;
        if (num == primero)
        {
            total++;
        }
    }

    cout << "El numero de veces que aparece el numero " << primero << " es " << total << endl;
    return 0;
}
```

C++

programa que lea una secuencia de enteros acabada en 0 y escriba si la diferencia entre el mayor entero y el menor entero es superior a 100


```
#include <iostream>
int main() {
    int num, mayor, menor, diferencia;
    cout << "Introcuzca una serie de numeros: ";
    cin >> num;
    mayor = num;
    menor = num;

    while (num != 0)    {
         if (num > mayor)
                mayor = num;
         if (num < menor)
                menor = num;
        cin >> num;
     }
```

```
 diferencia = mayor - menor;
    cout << "El mayor numero introducido es: " << mayor
<< endl;
    cout << "El menor numero introducido es: " << menor
<< endl;
    cout << "La diferencia entre ambos es: " <<
diferencia << ", ";

    if (diferencia > 100)
          cout << "superior a 100" << endl;
     else  cout << "no superior a 100" << endl;
return 0;
}
```

C++

Ejercicio

Una empresa textil tiene una serie de datos que indican si el producto al cual hace referencia pasa los controles de calidad o no. Estos datos vienen dados por los caracteres 'A' o 'B' de manera que el indicador 'A' significa que el producto al cual hace referencia pasa los controles de calidad, pero el indicador 'B' significa que el producto no pasa los controles de calidad. Suponiendo que el último dato (el centinela) es el carácter 'C', diseña un programa que indique el porcentaje de productos que pasan los controles de calidad.

C++

```
#include <iostream>
int main(){
    char caracter;
    int totalA = 0, totalB = 0, proporcion=0;
    cout << "Introcuzca los controles de los productos: ";
    cin >> caracter;
    while ((caracter != 'A') && (caracter != 'B') && (caracter != 'C'))  {
       cin >> caracter;
    }
    while (caracter != 'C')   {
          if (caracter == 'A')
               ++totalA;
          else  ++totalB;
          cin >> caracter;
          while ((caracter != 'A') && (caracter != 'B') && (caracter != 'C'))
              {
                   cin >> caracter;
              }
    }

    proporcion = (totalA*100)/(totalA+totalB);
    cout << "El porcentaje de de productos que pasan los controes de calidad es " << proporcion <<
   "%." << endl;
       return 0;
   }
```

C++

Ejercicio

Un excursionista va anotando cada media hora la altura (valor entero) respecto al nivel del mar a la que se encuentra, y al final anota el valor -1. Dada una secuencia de estos valores enteros, diseña un programa que indique por cuantos picos (estrictos o no) con altura mayor de 1000 ha pasado el excursionista.

Diremos que una subsecuencica de enteros forma un pico (estricto o no) si el primer y el ultimo valor son menores que los términos comprendidos entre ellos dos, siendo estos términos iguales.

C++

```
#include <iostream>
int main(){
      int num, ant2, ant1, picos = 0, mayorMil = 0;
    cout << "Introduzca una serie de numeros: ";
    cin >> ant2 >> ant1 >> num;
        while (num != -1)     {
          if ((ant2 < ant1) && (ant1 > num))  {
             if (ant1 > 1000) {
                ++mayorMil;
             }
             ++picos;
          }
          if (ant1 != num)   {
             ant2 = ant1;
             ant1 = num;
          }
          cin >> num;
    }
    cout << "El excursionista ha pasado por un total de " << picos << " picos." << endl;
    cout << "El excursionista ha pasado por " << mayorMil << " picos mayores de 1000." << endl;
    return 0;
}
```

C++

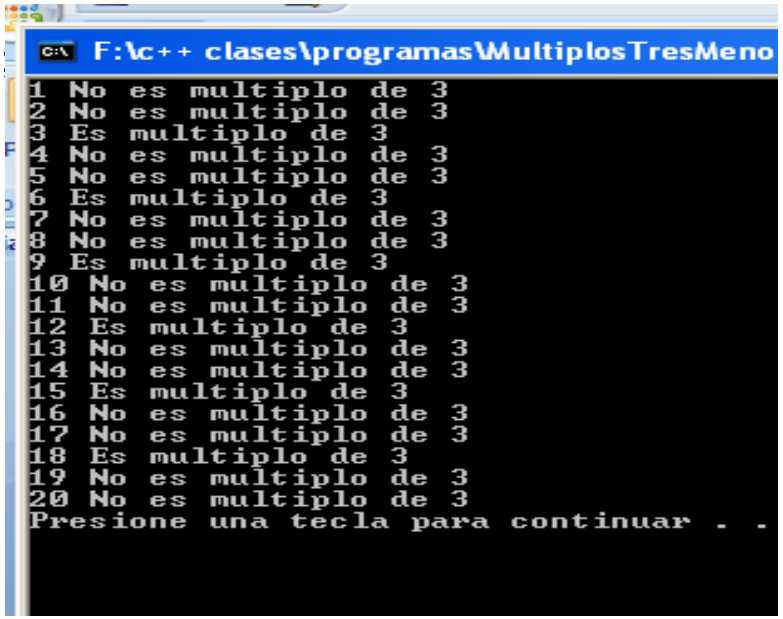
Sumatorio de la página 38


```
int main()
{
  int i, cubo, suma=0;
  for (i=0 ; i <=10; i++)
  {
     cubo = i * i * i;
     suma = suma + cubo;
  }
Cout<<“el sumatorio es”, suma);
}
```

C++

Multiplos de 3 menores o iguales a 20


```
#include <iostream>
using namespace std;
int main ()
{
   for (int i=1;i<=20;i++)
   {
    cout<<i;
    if (i%3==0) cout<<" Es multiplo de 3" <<endl;
    else cout<<" No es multiplo de 3"<<endl;
   }
   system("pause");
   return 0;
}
```



C++

Fibonacci 1 1 2 3 5 8

// Ejemplo de la Serie de Fibonacci simple

```
#include<iostream>
using namespace std;
int main()
{
  float y0=0,y1=1;
  int i;
  for(i=1;i<21;i++)
   { yn=y1+y0;    // yn acumula serie
     cout << yn << "\n";
     y1=yn;       // y1 toma valor del actual yn
     y0=y1-y0;    // y0 asume valor de y1 - y0 para continuar la fórmula
   }
   system ("pause");
   return 0;
}
```

$$F_1 = 1$$
$$F_2 = 1$$
$$F_{n+2} = F_n + F_{n+1}, \ \forall n \geq 1$$

C++

Al inventor del juego del ajedrez le dijeron que pidiera lo que quisiera, que estaba concedido y se le ocurrió pedir: 1 gramo de trigo por la primera casilla, 2 gramos por la segunda, 4 por la tercera, 8 por la cuarta, 16 por la quinta y así sucesivamente. Se pide un programa que escriba una matriz de ocho por ocho enteros, representando en cada posición la cantidad de granos correspondientes a la casilla respectiva:

1 2 4 8 16 32 64

128 .. ..

.. ..

¿Se puede hacer alguna observación sobre este problema?

C++

- Una **terna pitagórica** es un conjunto de tres números enteros positivos tales que . $x^2 + y^2 = z^2$

Encontrar ternas pitagóricas hasta el 100.

C++

Tabla de multiplicar


```
//Tablas de multiplicar
#include <iostream>
using namespace std;
int main()
{
  for(int i=1; i<=10; i++)
   {
     for(int j=1;j<=10;j++)
       cout<<i<<"x"<<j<<"="<<i*j<<endl;
     system("pause");
   }
  system("pause");
  return 0;
}
```

C++

Factorial


```
//Factorial de un numero
#include <iostream>
using namespace std;
int main()
{
  int num,factorial=1;
  cout<<"Dame un número"<<endl;
  cin>>num;
  for(int i=2; i<=num; i++)
    factorial=factorial*i;
  cout<< num<<"!= "<<factorial<<"\n";
  system("pause");
  return 0;
}
```

C++

Sentencias de control

Sentencias BREAK y CONTINUE
(no lo veo en clase)

Sentencias *break* y *continue*

La Sentencia **break**, además de terminar los case de una Sentencia **switch**, sirve para forzar la terminación inmediata de un bucle.

La Sentencia **continue** , interrumpe la ejecución del cuerpo del bucle y fuerza una nueva iteración.

C++

Sentencias *break* y *continue*


```
int main()
{
        ….
        for ( int t = 0; t <100 ;  t++ )
         {
                cout << t;
                if (t==10)
                     break;       // salir del for
         }
         ….
        return 0;
}
```

```
int main()
{
        ….
        for ( int t = 0; t <100 ;  t++ )
         {
                if (t%10==0)
                     continue;
                cout << t;
          }
          ….
        return 0;
}
```

C++

Sentencias *break* y *continue*


```
int main()
{
        ....
        do
         {
                cin >> x;
                if ( x < 0 )
                      continue;
                cout << x;

          }while (x !=100);
          ....
        return 0;
}
```

Ignora lo que sigue y vuelve a comprobar la condición.

C++