CS1102 – PROGRAMACIÓN ORIENTADA A OBJETOS 1
CICLO 2019-1

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CIENCIA DE LA COMPUTACIÓN

Unidad 2: Funciones y recursividad

http://bit.ly/2HRBWgq

Profesores:

Ernesto Cuadros- Vargas, PhD. *ecuadros@utec.edu.pe*

María Hilda Bermejo, M. Sc. *mbermejo@utec.edu.pe*

Telegram:

1. Configurar tu cuenta

2. http://bit.ly/2TJnwBq

Logro de la sesión:

Al finalizar la sesión, los alumnos desarrollan sus programas utilizando funciones propias y recursividad.

Demo.

Conceptos previos:

Ámbito o *Scope* de una variable.

Creando tipos de datos.

Scope o ámbito:

```
int main()
{
    int a, b, c;                // variables que tienen alcance de función main
    cin >> a;
    cin >> b;
    c = mayor(a,b);
    cout << "El mayor valor es: "  << c << endl;

    for (int i = 0; i < 5; i++) // la variable i solo existe en el bucle "for"
    {
        cout << "Valor de i : "<< i << endl;
    }
    // A partir de esta parte del código la variable i no existe.

    if (a < b)
    {   int temporal = a;       // el alcance de la variable temporal es el bloque if
        a = b;
        b = temporal;
    }
}
```

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " << linea;
 return 0;
}
```

linea* y *contador

Son variables globales.

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0

Lo que imprime:

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea

0

contador

0

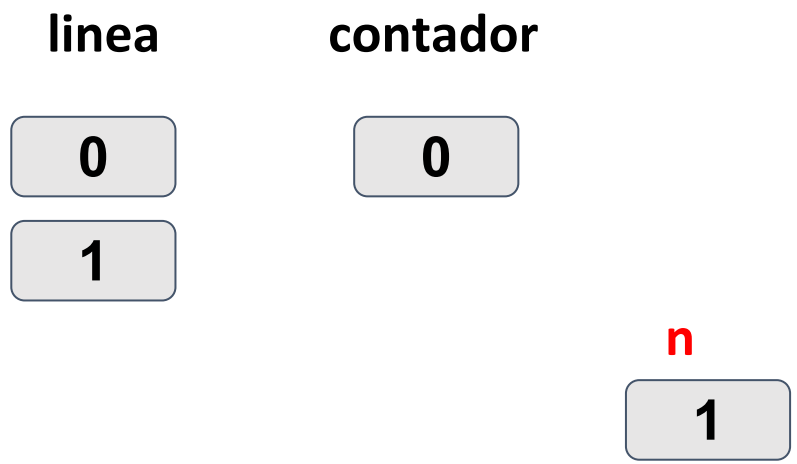
n

1

Lo que imprime:

Scope o ámbito - Variables Globales:

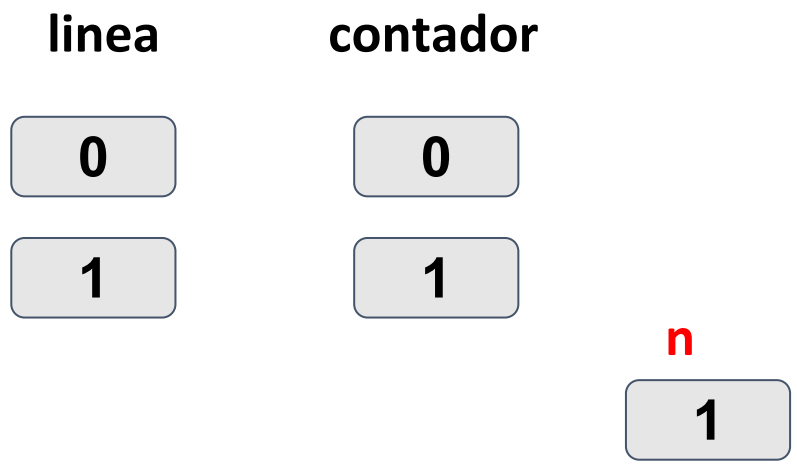
```
#include <iostream>
using namespace std;

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```



Lo que imprime:

Scope o ámbito - Variables Globales:

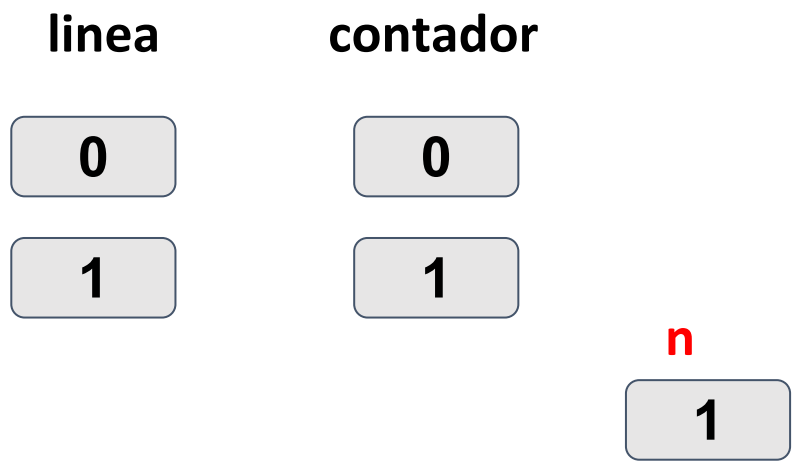
```
#include <iostream>
using namespace std;

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```



Lo que imprime:

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```



Lo que imprime:

Valor : 1

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1

Lo que imprime:

Valor : 1

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1

Lo que imprime:

Valor : 1

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1

n

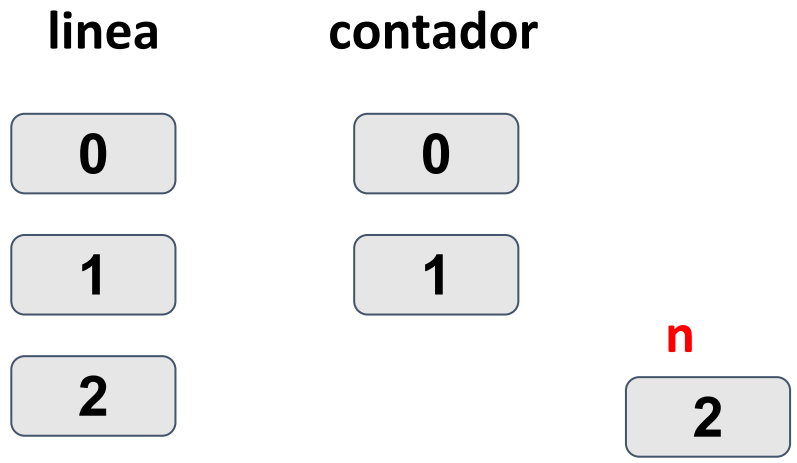
2

Lo que imprime:

Valor : 1

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

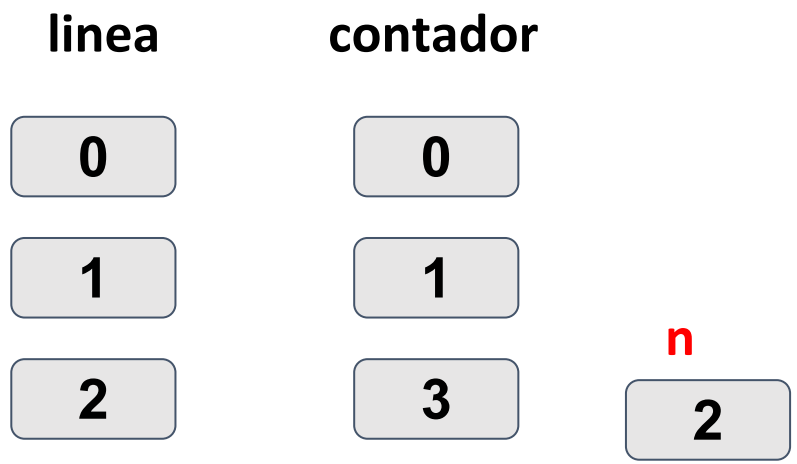


Lo que imprime:

Valor : 1

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

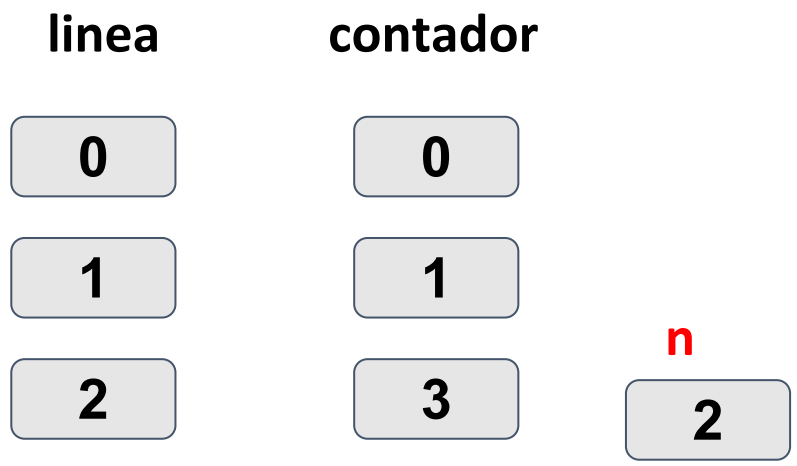


Lo que imprime:

Valor : 1

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```



Lo que imprime:

Valor : 1
Valor : 3

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3

Lo que imprime:

Valor : 1
Valor : 3

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

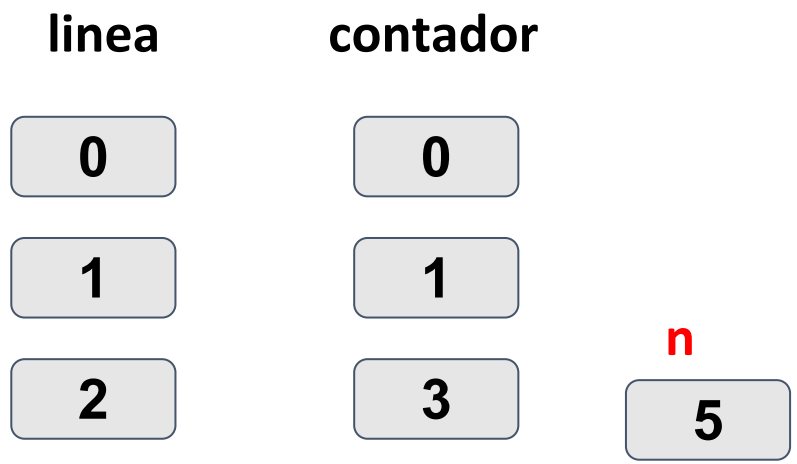
linea	contador
0	0
1	1
2	3

Lo que imprime:

Valor : 1
Valor : 3

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```



Lo que imprime:

Valor : 1
Valor : 3

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3
3	

n

5

Lo que imprime:

Valor : 1

Valor : 3

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3
3	8

n

5

Lo que imprime:

Valor : 1
Valor : 3

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3
3	8

n

5

Lo que imprime:

Valor : 1
Valor : 3
Valor : 8

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " << linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3
3	8

Lo que imprime:

Valor : 1
Valor : 3
Valor : 8

Scope o ámbito - Variables Globales:

CIENCIA DE LA COMPUTACIÓN

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " << linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3
3	8

Lo que imprime:

```
Valor : 1
Valor : 3
Valor : 8
Numero de Llamadas : 3
```

Scope o ámbito - Variables Globales:

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

int linea = 0;
int contador=0;

int imprimir(int n)
{
 linea++;
 contador = contador + n;
 cout << "Valor : " << contador << "\n";
}


int main()
{
 imprimir(1);
 imprimir(2);
 imprimir(5);
 cout << "Numero de llamadas : " <<
linea;
 return 0;
}
```

linea	contador
0	0
1	1
2	3
3	8

Lo que imprime:

Valor : 1
Valor : 3
Valor : 8

Type Aliases:

Algunas veces se necesita nuevos nombres para un tipo dato, porque facilita los cambios o mantenimiento en los programas:

Ej1:

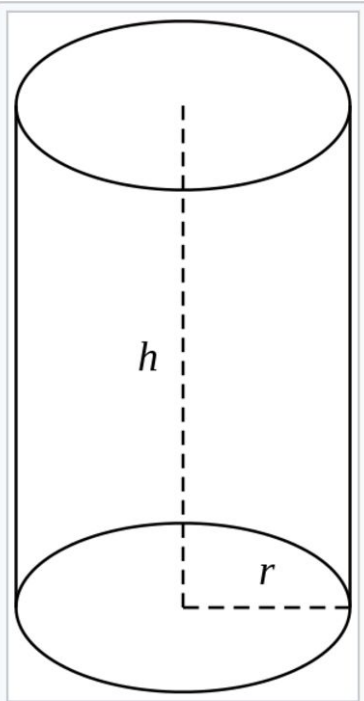
```
typedef  int int32_t;  //-- luego se puede declarar la variable como sigue:

 int32_t  numero=34;
```

Ej2:

```
typedef double tipoReal; //-- luego se puede declarar la variable como sigue:

tipoReal  dato=3.5643;
```

Ejemplo 1 :

Escribir un programa que permita hallar el área total y el volumen de un cilindro circular recto, si se conoce el valor del radio y la altura.



Un cilindro circular recto.

$$\text{areaDeLaBase} = \pi\, r^2$$

$$\text{areaLateral} = 2\,\pi r h$$

$$\text{areaTotal} = 2\pi r h + 2\pi r^2$$

$$\text{volumen} = \pi r^2 \text{h}$$

Solución 1: Todo el código está en un solo archivo main.cpp

```
#include <iostream>
using namespace std;
const double PI=3.1415;

double LeeDato(string mensaje)
{//---------------------------------------
 double dato;
do
{ cout << mensaje;
  cin >> dato;
}while(dato<=0);
return dato;
}

double areaDeLaBase(double r)
{//-------------------------------------
 return(PI*r*r);
}

double areaTotal(double r, double h)
{//-----------------------------------------
 return(2*PI*r*h + 2*areaDeLaBase(r));
}

double volumen(double r, double h)
{//-----------------------------------------
 return(areaDeLaBase(r) * h);
}
```

```
int main()
{double radio, altura;

 radio=LeeDato("Radio: ");
 altura=LeeDato("Altura: ");
 cout << "Area total: " << areaTotal(radio, altura);
 cout << "\n";
 cout << "Volumen   : " << volumen(radio,altura);
 return 0;
}
```

Pantalla de salida :

```
Radio: 0
Radio: -1
Radio: 10
Altura: 3
Area total: 816.79
Volumen   : 942.45
```

Lo ideal es que el código, esté distribuido en varios archivos:

main.cpp

UFunciones.h (es el header)

UFunciones.cpp

main. cpp

```
#include <iostream>
#include "UFunciones.h"  //-- Header
using namespace std;

int main()
{double radio, altura;

 radio=LeeDato("Radio: ");
 altura=LeeDato("Altura: ");
 cout << "Area total: " << areaTotal(radio, altura);
 cout << "\n";
 cout << "Volumen   : " << volumen(radio,altura);
 return 0;
}
```

UFunciones.h

```
#ifndef CILINDRO1_UFUNCIONES_H
#define CILINDRO1_UFUNCIONES_H

#include <iostream>
using namespace std;

const double PI=3.1415;

double LeeDato(string mensaje);
double areaDeLaBase(double r);
double areaTotal(double r, double h);
double volumen(double r, double h);

#endif //CILINDRO1_UFUNCIONES_H
```

UFunciones. cpp

```
#include "UFunciones.h"

double LeeDato(string mensaje)
{//------------------------
        double dato;
        do
        {       cout << mensaje;
                cin >> dato;
        }while(dato<=0);
        return dato;
}

double areaDeLaBase(double r)
{//----------------------
        return(PI*r*r);
}

double areaTotal(double r, double h)
{//-----------------------------
        return(2*PI*r*h + 2*areaDeLaBase(r));
}

double volumen(double r, double h)
{//---------------------------
        return(areaDeLaBase(r) * h);
}
```

CIENCIA DE LA COMPUTACIÓN

Ahora utilizando tipos genéricos:

Esta es una buena práctica que facilita el mantenimiento del código

main. cpp

```
#include <iostream>
#include "UFunciones.h"
using namespace std;

int main()
{tipo_Real radio, altura;

 radio=LeeDato("Radio: ");
 altura=LeeDato("Altura: ");
 cout << "Area total: " << areaTotal(radio, altura);
 cout << "\n";
 cout << "Volumen   : " << volumen(radio,altura);
 return 0;
}
```

UFunciones.h

```
#ifndef CILINDRO1_UFUNCIONES_H
#define CILINDRO1_UFUNCIONES_H

#include <iostream>
using namespace std;

typedef double tipo_Real;

const tipo_Real PI=3.1415;

tipo_Real LeeDato(string mensaje);
tipo_Real areaDeLaBase(tipo_Real r);
tipo_Real areaTotal(tipo_Real r, tipo_Real h);
tipo_Real volumen(tipo_Real r, tipo_Real h);

#endif //CILINDRO1_UFUNCIONES_H
```

UFunciones.h

```
#ifndef CILINDRO1_UFUNCIONES_H
#define CILINDRO1_UFUNCIONES_H

#include <iostream>
using namespace std;

typedef double tipo_Real;

const tipo_Real PI=3.1415;

tipo_Real LeeDato(string mensaje);
tipo_Real areaDeLaBase(tipo_Real r);
tipo_Real areaTotal(tipo_Real r, tipo_Real h);
tipo_Real volumen(tipo_Real r, tipo_Real h);

#endif //CILINDRO1_UFUNCIONES_H
```

UFunciones. cpp

```
#include "UFunciones.h"

tipo_Real LeeDato(string mensaje)
{//------------------------
        tipo_Real dato;
        do
        {       cout << mensaje;
                cin >> dato;
        }while(dato<=0);
        return dato;
}

tipo_Real areaDeLaBase(tipo_Real r)
{//------------------------
        return(PI*r*r);
}

tipo_Real areaTotal(tipo_Real r, tipo_Real h)
{//-----------------------------
        return(2*PI*r*h + 2*areaDeLaBase(r));
}

tipo_Real volumen(tipo_Real r, tipo_Real h)
{//---------------------------
        return(areaDeLaBase(r) * h);
}
```

Ejemplo 2:

CIENCIA DE LA COMPUTACIÓN

Escribir un programa, que permita leer como dato un número entero de al menos 7 dígitos y el programa indique si el número es capicúa.

Un número es capicúa si se lee el mismo número, cuando se lee de izquierda a derecha o se lee de derecha a izquierda

Ejemplo 1:

Ingrese número: 1235321
Es número es capicúa

Ejemplo 2:

Ingrese número: 8765
El número no es capicúa.

main. cpp

```
#include <iostream>
#include "UMisFunciones.h"

using namespace std;

int main()
{tipo_Entero numero;

 numero=LeeNumero();
 if(numero == NumeroInvertido(numero))
   cout<<"Es capicua!";
 else
   cout<<"No es capicua!";
 return 0;
}
```

UMisFunciones. h

```
#ifndef CAPICUA_UMISFUNCIONES_H
#define CAPICUA_UMISFUNCIONES_H

#include <iostream>
using namespace std;

typedef long int tipo_Entero;

tipo_Entero LeeNumero();
tipo_Entero NumeroInvertido(tipo_Entero num);

#endif //CAPICUA_UMISFUNCIONES_H
```

UMisFunciones. cpp

```
#include "UMisFunciones.h"

tipo_Entero LeeNumero()
{//------------------------------
 tipo_Entero n;

        do{
                cout <<"Numero de por lo menos 7 digitos: ";
                cin>>n;
        }while(n<1000000);
        return n;
}

tipo_Entero NumeroInvertido(tipo_Entero num)
{//------------------------------------------------------------
        tipo_Entero numeroAlReves, digito;

        numeroAlReves=0;
        while(num>0)
        {
                digito = num%10;
                numeroAlReves = numeroAlReves*10 + digito;
                num/=10;
        }
        return numeroAlReves;
}
```

Algoritmo: por Ej si el número fuese: 1234567
numeroAlReves, se va formado asi:

0*10 + 7 = 7
7*10 + 6 = 76
76*10+5 = 765
765*10+4 = 7654
7654*10+3=76543
76543*10+2=765432
765432*10+1 = 7654321

Transferencia de parámetros por referencia

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a	b
10	33

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33

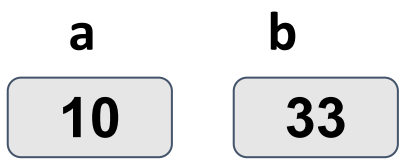
a	b
10	33

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

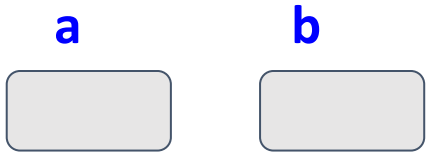
Lo que se imprime Programa

a=10 b=33



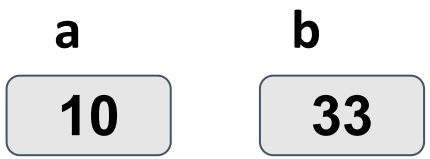
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



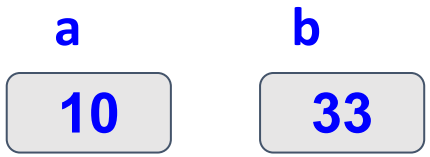
Lo que se imprime Programa

a=10 b=33



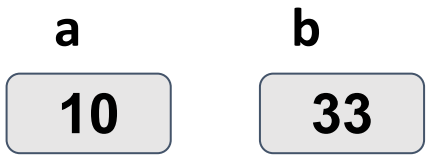
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



Lo que se imprime Programa

a=10 b=33

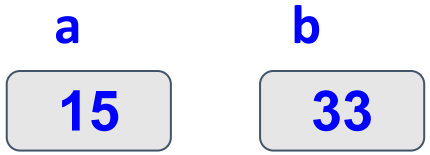


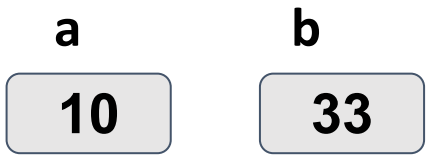
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33



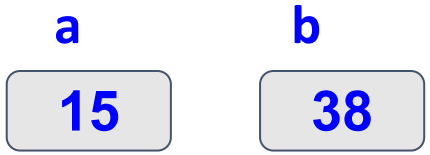


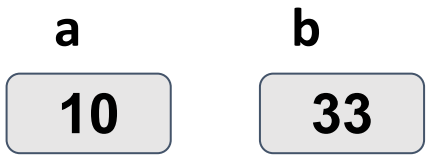
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33



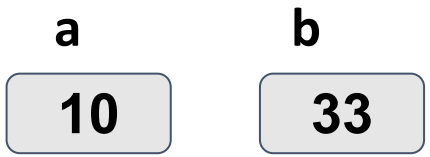


Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33

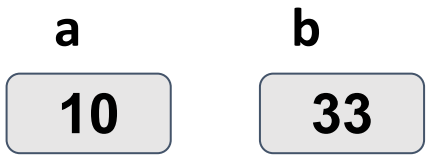


Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

```
a=10   b=33
a=10   b=33
```

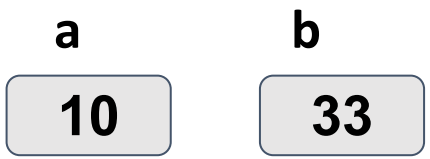


Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

```
a=10   b=33
a=10   b=33
```



Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

```
a=10   b=33
a=10   b=33
```

a	b
30	88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio

a	b
30	88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio

a	b
30	88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

```
a=10   b=33
a=10   b=33

Valores antes del intercambio
a=30  b=88
```

a	b
30	88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

```
a=10   b=33
a=10   b=33

Valores antes del intercambio
a=30   b=88
```

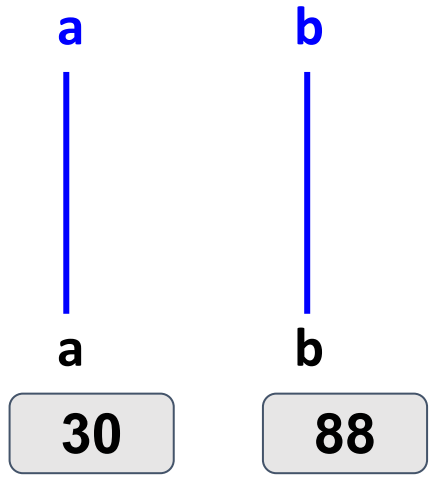
a b

a b

30 88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



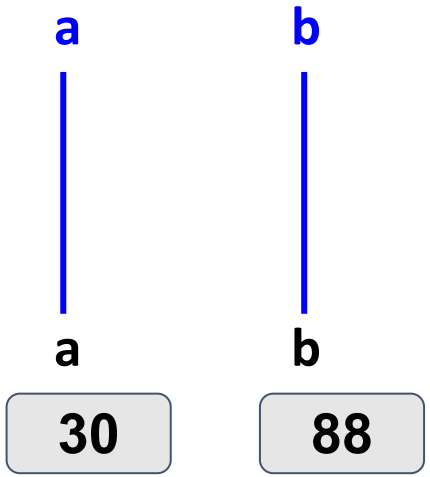
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



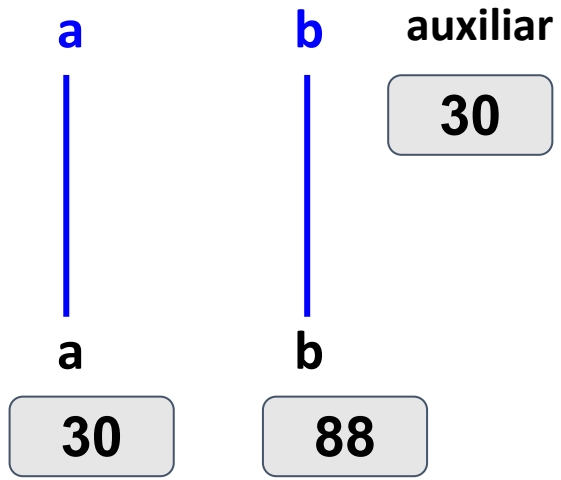
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



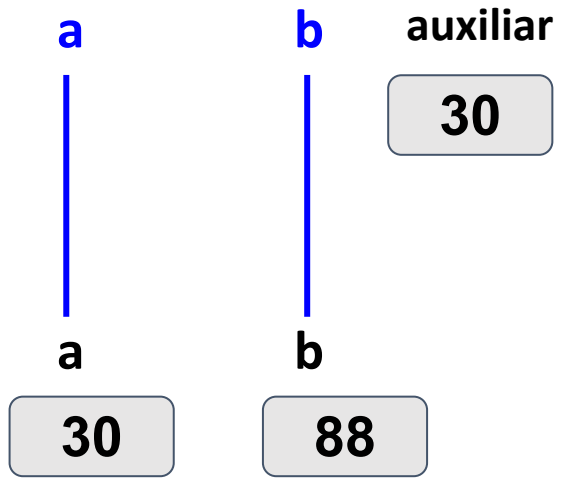
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



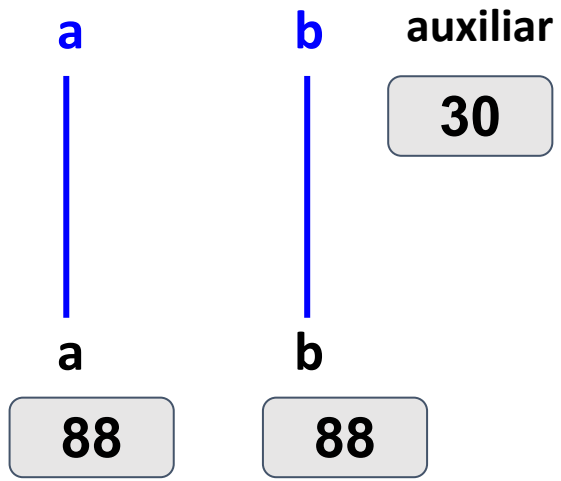
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



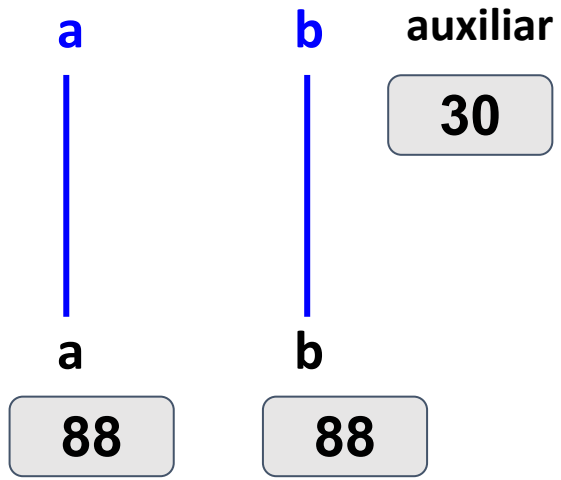
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



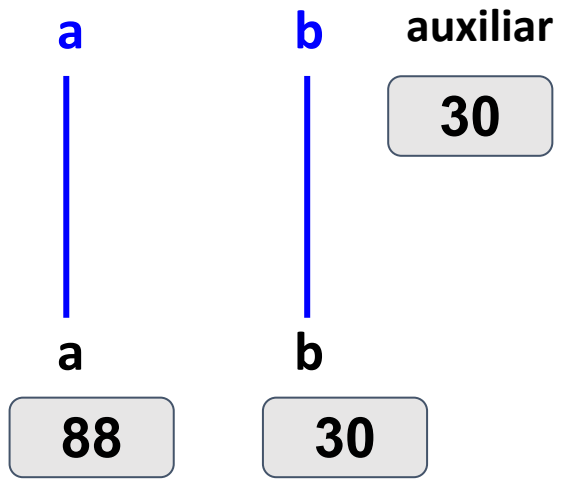
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



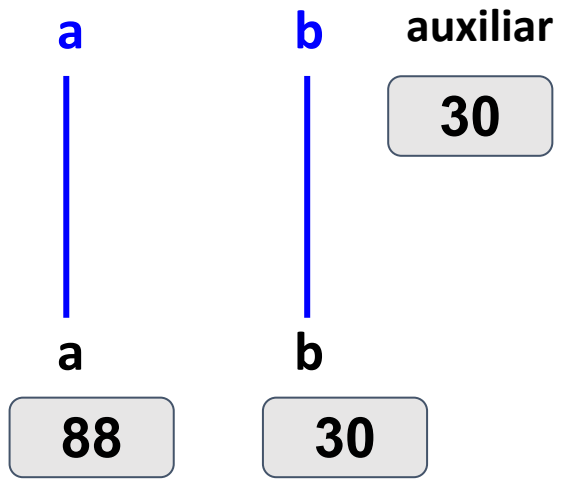
Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```



Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88

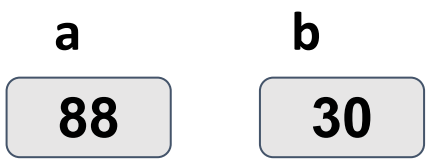
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88



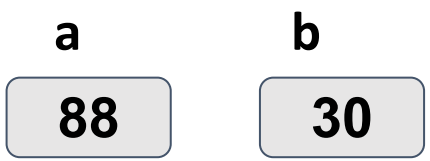
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88
Valores después del intercambio



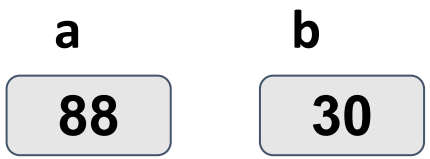
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88
Valores despues de intercambio
a=88 b=30



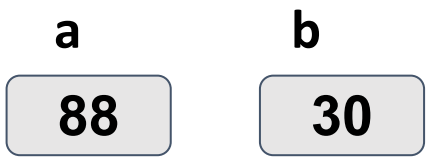
Transferencia por valor y Transferencia por Referencia

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

void AumentaenCinco(int a,int b)
//---------------------------------------
{ a = a + 5;
 b = b + 5;
}

void IntercambiarValores(int &a, int &b)
//-----------------------------------------------
{  auto auxiliar = a;
  a= b;
  b=auxiliar;
}

int main()
{  int a=10, b=33;

  cout << "a = " << a << " " << "b = " << b << "\n";
  AumentaenCinco(a,b);
  cout << "a = " << a << " " << "b = " << b << "\n";
  cout << "\n";
  a=30; b=88;
  //--- ahora usamos transferencia por referencia
  cout << "Valores antes del intercambio\n";
  cout << "a = " << a << " " << "b = " << b << "\n";
  IntercambiarValores(a,b);
 cout << "Valores despues del intercambio\n";
 cout << "a = " << a << " " << "b = " << b << "\n";
 return 0;
}
```

Lo que se imprime Programa

a=10 b=33
a=10 b=33

Valores antes del intercambio
a=30 b=88
Valores después del intercambio
a=88 b=30



Ejemplo 3:

Desarrolla un programa que permite leer como dato una cantidad de segundos (número mayor a 1) y el programa realice la conversión a través del uso de una función a horas, minutos y segundos.

main. cpp

```cpp
#include <iostream>
#include "UMisFunciones.h"
using namespace std;

int main()
{tipo_Entero segundos;
 tipo_Entero Horas, Min,Seg;

 segundos=LeeSegundos();
 ConvertirAHorasMinSeg(segundos,Horas, Min,Seg);
 cout<<"Equivale a ";
 cout<< Horas<<" horas " << Min <<" minutos y " << Seg << "
segundos";
 return 0;
}
```

UMisFunciones. h

```cpp
#ifndef SEGUNTOSAHORASMINSEG_UMISFUNCIONES_H
#define SEGUNTOSAHORASMINSEG_UMISFUNCIONES_H

#include <iostream>
using namespace std;

typedef long int tipo_Entero;

tipo_Entero LeeSegundos();
void ConvertirAHorasMinSeg(tipo_Entero Segundos,
                           tipo_Entero &horas,
                           tipo_Entero &min,
                           tipo_Entero &seg);

#endif //SEGUNTOSAHORASMINSEG_UMISFUNCIONES_H
```

UMisFunciones. cpp

CIENCIA DE LA COMPUTACIÓN

```
#include "UMisFunciones.h"

tipo_Entero LeeSegundos()
{//--------------------------------
 tipo_Entero n;

      do{
            cout <<"Segundos <dato mayor a 1>: ";
            cin>>n;
      }while(n<=1);;
      return n;
}

void ConvertirAHorasMinSeg(tipo_Entero Segundos, tipo_Entero &horas, tipo_Entero &min, tipo_Entero &seg)
{//-------------------------------------------------------------------------------------------------
      horas = Segundos/3600;
      Segundos %= 3600;
      min = Segundos/60;
      seg = Segundos%60;
}
```

La función *ConvertiraHorasMinSeg*, en su identificador o nombre no retorna ningún valor, recibe 4 parámetros: el primero por valor y los 3 siguientes por referencia.

Recursividad

La recursividad ocurre cuando ….

Para hallar el resultado una función se invoca a si misma.

El algoritmo para hallar el factorial de un número es un ejemplo clásico del uso de recursividad.

fact(7) = 7*6*5*4*3*2*1 ó **fact(7) = 7 * fact(6)**

fact(9) = 9*8*7*6*5*4*3*2*1 ó **fact(8) = 8 * fact(7)**

fact(14) = 14 * fact(13)

Generalizando:

```
fact(n) = n * fact(n-1)
```

Veamos cómo funciona la recursividad, hallando el factorial de 5

```
fact(5) = 5 * fact(4)
```

Recursión:

```
fact(5) = 5 * fact(4)
          4 * fact(3)
```

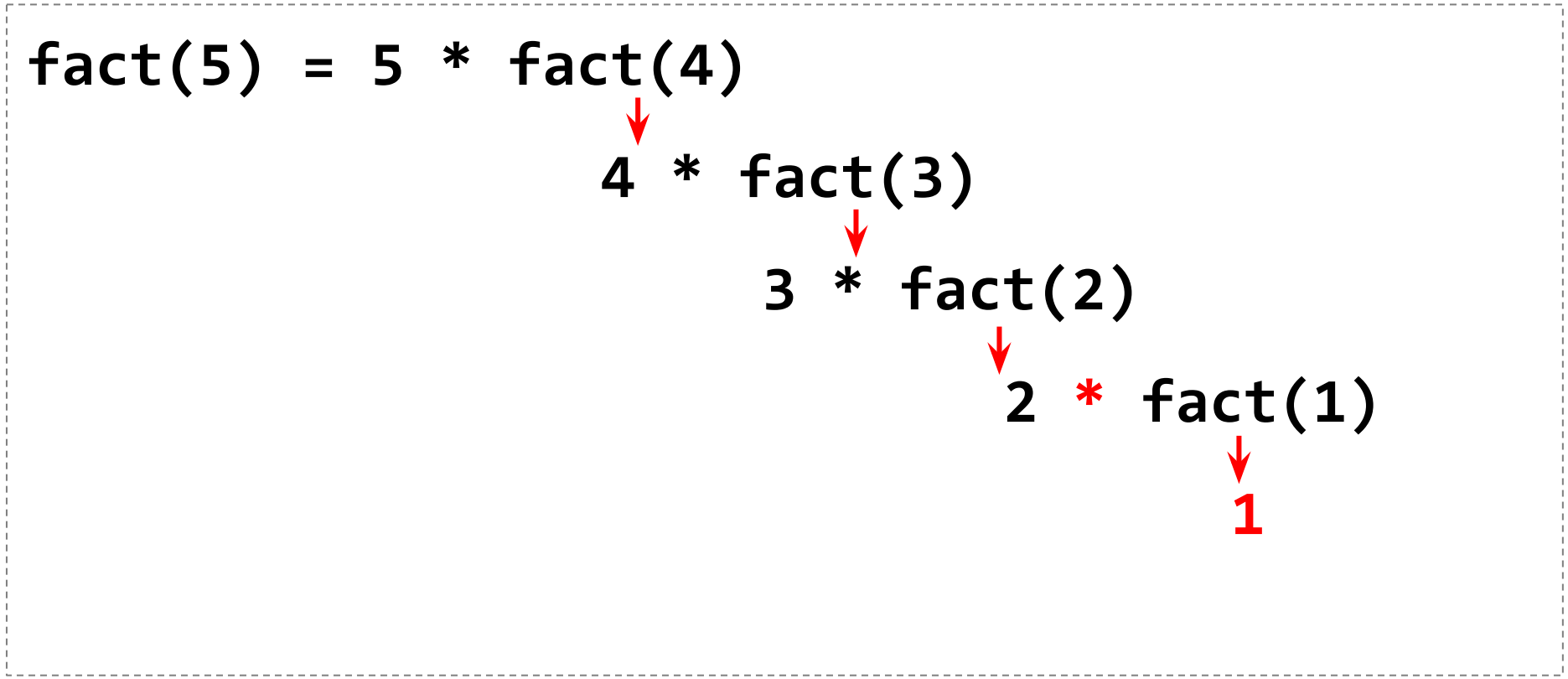
Recursión:

```
fact(5) = 5 * fact(4)
          4 * fact(3)
              3 * fact(2)
```

Recursión:

```
fact(5) = 5 * fact(4)
                  4 * fact(3)
                        3 * fact(2)
                              2 * fact(1)
```

Recursión:



Recursión:

```
fact(5) = 5 * fact(4)
                ↓
              4 * fact(3)
                    ↓
                  3 * fact(2)
                        ↓
                      2 * fact(1)
                            ↑
                            1
```

Recursión:

```
fact(5) = 5 * fact(4)
          4 * fact(3)
              3 * fact(2)
                  2 * 1
```

Recursión:

```
fact(5) = 5 * fact(4)
                ↓
          4 * fact(3)
                ↑
            3 * 2
```

Recursión:

```
fact(5) = 5 * fact(4)
```

```
          4 * 6
```

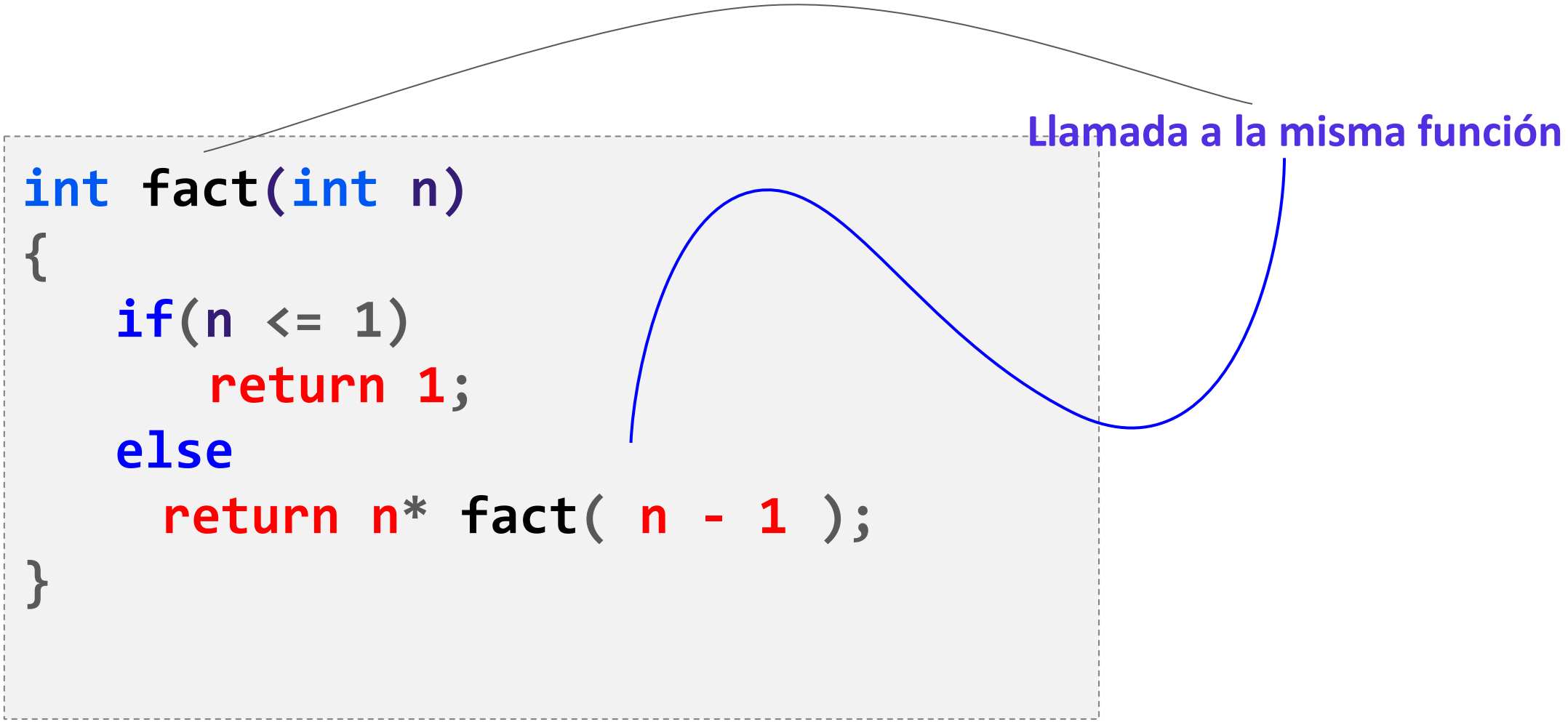
Recursión:

```
fact(5) = 5 * 24
```

Recursión:

```
fact(5) = 120
```

Recursión:


Llamada a la misma función
int fact(int n)
{
if(n <= 1)
return 1;
else
return n* fact(n - 1);
}

Recursión:

```
unsigned long fact(unsigned int n)
{
    if(n <= 1)
        return 1L;
    return n* fact( n - 1 );
}
```

Llamada a la misma función

Factorial - Iterativo

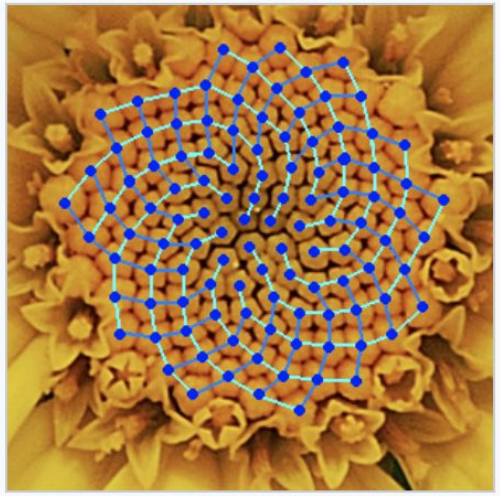
```
#include <iostream>
using namespace std;

unsigned  long factorial(unsigned int  n)
{unsigned long f;

 f=1;
 for(int i=2;i<=n; i++)
   f*=i;
 return f;
}

int main()
{
 unsigned int n;
 cout << "Numero: ";
 cin >> n;
 cout << "Factorial ("<< n <<") = ";
 cout << factorial(n);
 return 0;
}
```

Factorial - Recursivo

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

unsigned long factorial(unsigned int  n)
{
 if (n<=1)
   return 1L;
 return (n * factorial(n-1));
}

int main()
{
 unsigned int n;
 cout << "Numero: ";
 cin >> n;
 cout << "Factorial ("<< n <<") = ";
 cout << factorial(n);
 return 0;
}
```

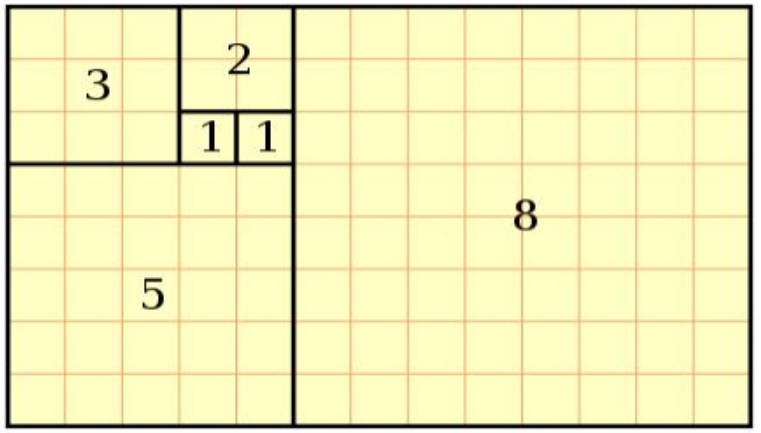
Serie de Fibonacci:

0 1 1 2 3 5 8 13 21 34 55 89 144 233 ...



Botón de Camomila amarilla mostrando la ordenación en espirales de módulos 21 (color azul) y 13 (color cian). Este tipo de arrollamientos utilizando números consecutivos de Fibonacci aparecen en una gran variedad de plantas.

Espiral de Fibonacci en la sección de la concha de un nautilus.



Al construir bloques cuya longitud de lado sean números de Fibonacci se obtiene un dibujo que se asemeja al rectángulo áureo (véase Número áureo).

Serie de Fibonacci:

Los números de Fibonacci quedan definidos por la ecuación:

$$f_n = f_{n-1} + f_{n-2}$$

partiendo de dos primeros valores predeterminados:

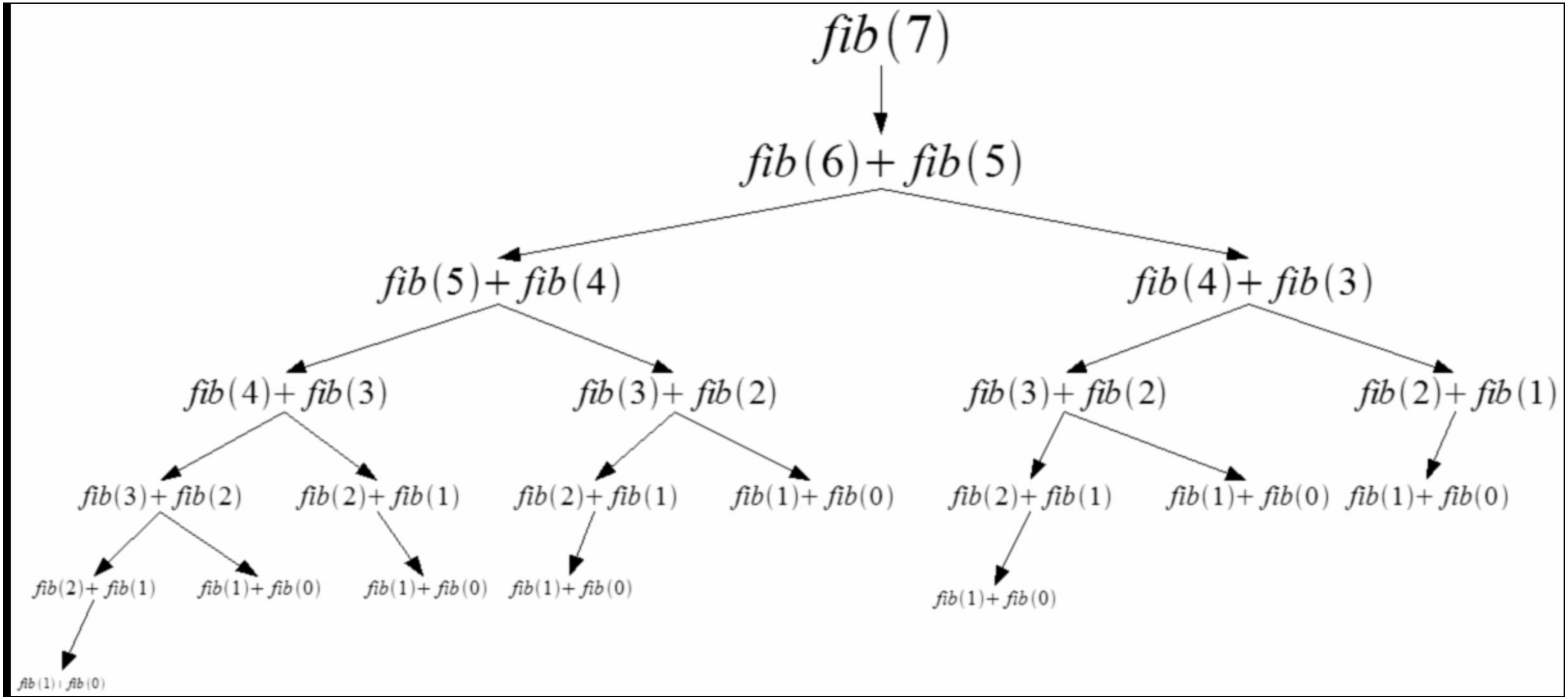
$$f_0 = 0$$
$$f_1 = 1$$

se obtienen los siguientes números:

- $f_2 = 1$
- $f_3 = 2$
- $f_4 = 3$
- $f_5 = 5$
- $f_6 = 8$
- $f_7 = 13$
- $f_8 = 21$

para $n = 2, 3, 4, 5, \ldots$

```
f(n) = f(n-1) + f(n-2)


Partiendo de:
f(0) = 0
f(1) = 1
```

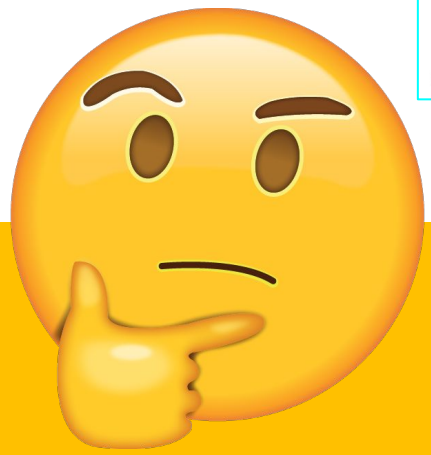

fib(7)
fib(6)+ fib(5)
fib(5)+ fib(4)
fib(4)+ fib(3)
fib(4)+ fib(3)
fib(3)+ fib(2)
fib(3)+ fib(2)
fib(2)+ fib(1)
fib(3)+ fib(2)
fib(2)+ fib(1)
fib(2)+ fib(1)
fib(1)+ fib(0)
fib(2)+ fib(1)
fib(1)+ fib(0)
fib(1)+ fib(0)
fib(2)+ fib(1)
fib(1)+ fib(0)
fib(1)+ fib(0)
fib(1)+ fib(0)
fib(1)+ fib(0)

Fibonacci - Iterativo

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

long int fibonacci (long int  n)
{
 if(n<2)
   return n;
 long int a=0,b=1, c;
 for(int i=2;i<=n;i++)
 {
   c = a+b;
   a = b;
   b = c;
 }
 return c;
}

int main()
{ long int n;
 cout <<"Termino : ";
 cin >> n;
 cout << "Fibonacci ("<< n <<") = ";
 cout << fibonacci(n);
 return 0;
}
```

Fibonacci - Recursivo

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

long int fibonacci (long int  n)
{
 if(n<2)
   return n;
 return( fibonacci(n-1) + fibonacci(n-2));
}

int main()
{ long int n;
 cout <<"Termino : ";
 cin >> n;
 cout << "Fibonacci ("<< n <<") =";
 cout << fibonacci(n);
 return 0;
}
```



1. **¿Qué es una función?**
2. **¿Cuándo es conveniente crear funciones?**
3. **¿Qué tipos de parámetros puede tener una función?**
4. **¿Cómo indicamos el valor que queremos que retorne la función?**
5. **¿Todas las funciones devuelven un valor?**
6. **¿Qué tipo de función es más simple de implementar: Recursivo o Iterativo?**
7. **¿Cuál es más óptimo en cuanto al consumo de los recursos (memoria, procesador, energía) del computador?**

Programación Competitiva

- **Nivel avanzado:**
  > Sábados 8-12 (am) (L301)
  http://bit.ly/icpc-avanzado

- **Nivel básico/intermedio:**
  > Lunes 7-9 (pm) (A803)
  > Viernes 6-8 (pm) (L604)

Taller de programación competitiva

Elige un horario (básico):

> Lunes, 22 de abril, 7-9pm, A803
> Viernes, 26 de abril, 6-8pm, L604

¿Ya tienes experiencia?

> Contest para horario avanzado:
  https://vjudge.net/contest/296812
> Forms de registro:
  http://bit.ly/icpc-avanzado

Recursos:

> #acm-announcements (Whatsapp):
  http://bit.ly/acm-announcements

> Gitlab/Materiales:
  https://gitlab.com/acmutec/icpc

> ACM Google Calendar:
  http://bit.ly/acmutec-calendar

CS1102 – PROGRAMACIÓN ORIENTADA A OBJETOS 1
CICLO 2019-1

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CIENCIA DE LA COMPUTACIÓN

Unidad 2: Funciones y recursividad

http://bit.ly/2HRBWgq

Profesores:

Ernesto Cuadros- Vargas, PhD. *ecuadros@utec.edu.pe*

María Hilda Bermejo, M. Sc. *mbermejo@utec.edu.pe*