



Universidad
Continental

Modularización de programas: Recursividad

FUNDAMENTOS DE PROGRAMACIÓN



Propósito

Teoría

Reconoce la sintaxis de la recursividad con tipos de módulos (función y procedimiento), en el lenguaje de programación

RECUERDA: Recursividad

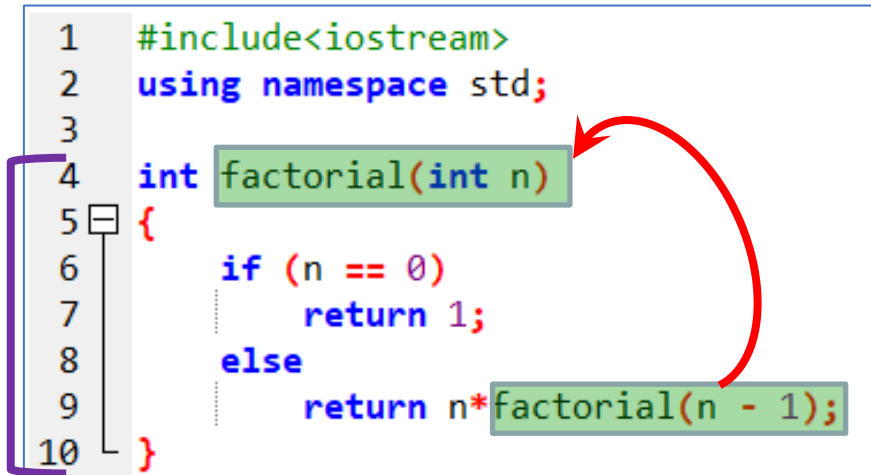
Módulos de programa que **se retroalimentan con valores generados por sí mismas** (se invocan a sí mismas).

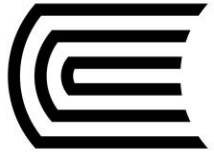
- Toda función recursiva **debe tener un punto de finalización**.
- La **función debe conducir a ese punto de finalización**.

Ejemplo:

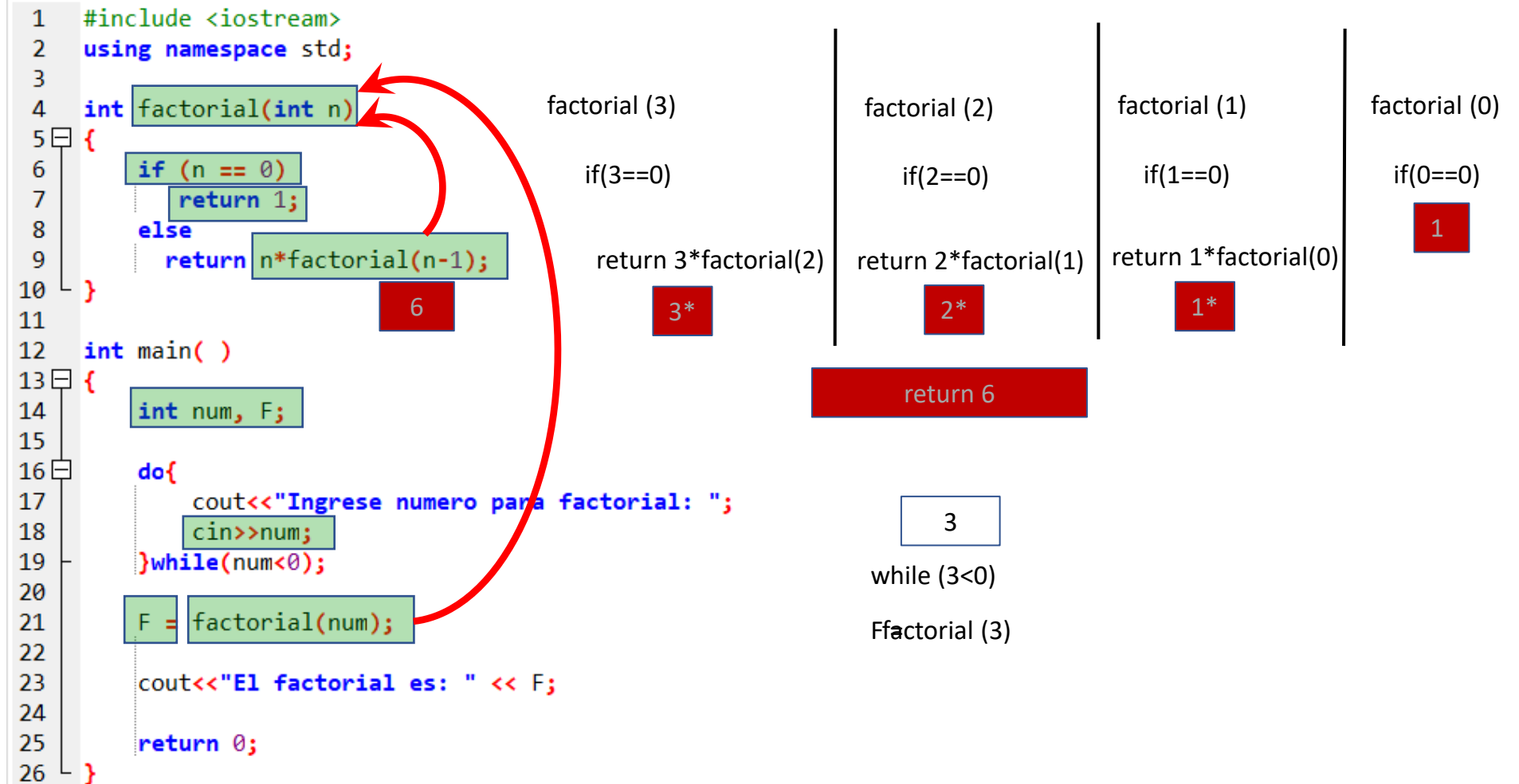
- Función Recursiva de Factorial.
- Función Recursiva de Multiplicacion.
- Función Recursiva de Potencia.
- Función Recursiva de Fibonacci
- Función Recursiva de Máximo Común Divisor
- Función Recursiva de Torres de Hanoi

```
1  #include<iostream>
2  using namespace std;
3
4  int factorial(int n)
5  {
6      if (n == 0)
7          return 1;
8      else
9          return n*factorial(n - 1);
10 }
```







Ejemplo: Recursividad del Factorial





Recomendación crear librería FRecurativas:

▼  Fundamentos de Programación

 Librerías

 Programas

Ahora, elaboremos los programas ejemplos..!

ucontinental.edu.pe



FRecurativas.h

```
1  #include<iostream>
2  using namespace std;
3
4  int factorial(int n)
5  {
6      if(n==0)
7          return 1;
8      else
9          return n*factorial(n - 1);
10 }
11
12 int potencia(int b, int e)
13 {
14     if(e==0)
15         return 1;
16     else
17         return b* potencia(b, e - 1);
18 }
19
20
21 int multiplica(int a, int b)
22 {
23     if(a == 0 || b == 0)
24         return 0;
25     else
26         if(b == 1)
27             return a;
28         else
29             return a + multiplica(a, b - 1);
30 }
31
32
33 int fibonaci(int n)
34 {
35     if(n==1 || n==2)
36     {
37         return 1;
38     }
39     else
40     {
41         return fibonaci(n-1)+fibonaci(n-2);
42     }
43 }
44
45 int division(int a, int b)
46 {
47     if(b>a)
48         return 0;
49
50     else
51         return division(a-b,b)+1;
52 }
```



Ejemplos de Funciones Recursivas

Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto1_FactorialRecursion

Nombre Programa: Programa1_FactorialRecursion

```
1 #include<iostream>
2 #include "FRecursivas.h"
3 #include "LecturaDatos.h"
4 using namespace std;
5
6 int main ()
7 {
8     int n, fact;
9
10    do{
11        cout<<"Ingrese numero: ";
12        n = leedatoe();
13        if(n<0)
14            cout<<"ERROR. Vuelva a ingresar >=0.";
15    }while(n<0);
16
17    fact = factorial(n);
18
19    cout<<"El valor de la potencia es: " << fact <<endl;
20
21 }
```

Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto2_PotenciaRecursion

Nombre Programa: Programa2_PotenciaRecursion

```
1 #include<iostream>
2 #include "FRecursivas.h"
3 #include "LecturaDatos.h"
4 using namespace std;
5
6 int main ()
7 {
8     int base, exponente, p;
9
10    do{
11        cout<<"Ingrese valor de base: ";
12        base = leedatoe();
13        if(base<0)
14            cout<<"ERROR. Vuelva a ingresar >=0.";
15    }while(base<0);
16
17    do{
18        cout<<"Ingrese valor de exponente: ";
19        exponente = leedatoe();
20        if(exponente<0)
21            cout<<"ERROR. Vuelva a ingresar >=0.";
22    }while(exponente<0);
23
24    p = potencia(base, exponente);
25
26    cout<<"El valor de la potencia es: " << p <<endl;
27
28 }
```

Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto3_MultiplicacionRecursion

Nombre Programa: Programa3_MultiplicacionRecursion

```
1 #include<iostream>
2 #include "FRecursivas.h"
3 #include "LecturaDatos.h"
4 using namespace std;
5
6 int main ()
7 {
8     int num1, num2, m;
9
10    do{
11        cout<<"Ingrese valor de numero1: ";
12        num1 = leedatoe();
13        if(num1<0)
14            cout<<"ERROR. Vuelva a ingresar >=0.";
15    }while(num1<0);
16
17    do{
18        cout<<"Ingrese valor de numero2: ";
19        num2 = leedatoe();
20        if(num2<0)
21            cout<<"ERROR. Vuelva a ingresar >=0.";
22    }while(num2<0);
23
24    m = multiplica(num1, num2);
25
26    cout<<"El valor de la potencia es: " << m <<endl;
27
28 }
```

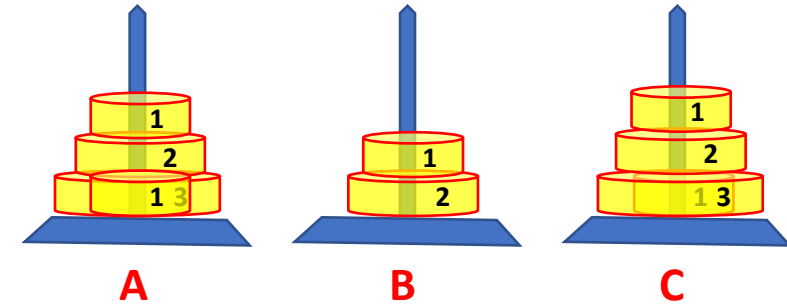


Ejemplo de Procedimiento Recursivo

Ejemplo:

Tres postes: A=Origen,B=Auxiliar,C=Destino

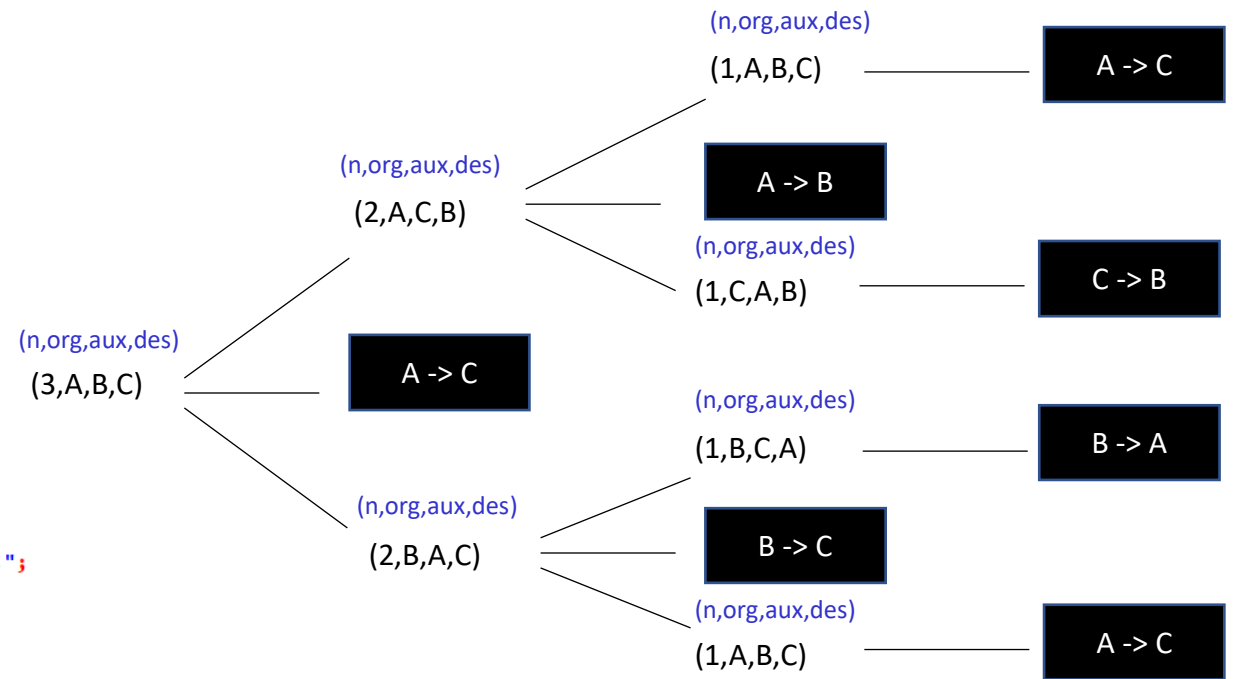
Cantidad de Discos: n=3

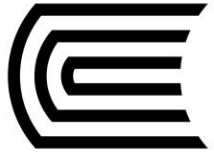


Carpeta: Programas Semana 14
Nombre Proyecto: Proyecto4_Hanoi
Nombre Programa: Programa4_Hanoi

```
void hanoi(int n, char org, char aux, char des)
{
    if(n == 1)
        cout<< org << " -> " << des;
    else
    {
        hanoi(n-1, org, des, aux);
        cout<< org << " -> " << des;
        hanoi(n-1, aux, org, des);
    }
}

int main ()
{
    int n;
    do{
        cout<<"Ingrese cantidad de discos: ";
        cin>>n;
        if(n<=0)
            cout<<"ERROR. Vuelva a ingresar >=0.";
    }while(n<=0);
    hanoi(n, 'A', 'B', 'C');
}
```





Conclusiones

Modularización
de programas:
Recursividad

Un módulo
genera una
invocación a sí
mismo.

Tiene un punto
de finalización
de la
recursividad.

El módulo de
recursión puede
ser función o
procedimiento.

Programa

```
1  #include <iostream>
2  using namespace std;
3
4  int factorial(int n)
5  {
6      if (n == 0)
7          return 1;
8      else
9          return n*factorial(n-1);
10 }
11
12 int main( )
13 {
14     int num, F;
15
16     do{
17         cout<<"Ingrese numero para factorial: ";
18         cin>>num;
19     }while(num<0);
20
21     F = factorial(num);
22
23     cout<<"El factorial es: " << F;
24
25     return 0;
26 }
```





Propósito

Práctica

Crea programas con recursividad, y con tipos de módulos (función y procedimiento), en el lenguaje de programación C/ C++.



Desarrollamos la Guía Práctica 14

Guía práctica N°14

Fundamentos de Programación

Cuarta Unidad: Módulos para la programación:

funciones propias del lenguaje de programación – recursividad

Sección :	Apellidos :
Docente :	Nombres :
	Fecha : / / Duración: 100 min
	Tipo de práctica: Individual <input checked="" type="checkbox"/> Grupal ()
Instrucciones: Lea detenidamente cada enunciado y desarrolle lo solicitado. Utilizar el Dev C++ para el desarrollo de los siguientes programas	

1. Elaborar un programa, que invoque al módulo recursivo de la suma (de la librería FRecursivas.h) y permita calcular su valor en un programa fuente (.cpp).
2. Elaborar un programa, que invoque al módulo recursivo de la resta (de la librería FRecursivas.h) y permita calcular su valor en un programa fuente (.cpp).
3. Elaborar un programa, que invoque al módulo recursivo del máximo común divisor (de la librería FRecursivas.h) y permita calcular su valor en un programa fuente (.cpp).
4. Elaborar un programa para calcular la siguiente ecuación, considerando las restricciones necesarias para su cálculo (ejemplo división entre cero).

$$z = \frac{n! - (a * b)}{n}$$

Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto5_SumaRecursion

Nombre Programa: Programa5_SumaRecursion

Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto6_RestaRecursion

Nombre Programa: Programa6_RestaRecursion

Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto7_MCDRecursion

Nombre Programa: Programa7_MCDRecursion

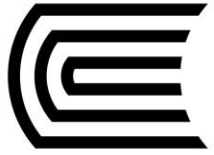
Carpeta: Programas Semana 14

Nombre Proyecto: Proyecto8_EcuacionRecursion

Nombre Programa: Programa8_EcuacionRecursion

Referencias bibliográficas consultadas y/o enlaces recomendados

- JOYANES, L. (2008). Fundamentos de Programación. Algoritmos, estructura de datos y objetos, 4ª Edición. Madrid: McGraw-Hill.
- Beekman, G., Pacheco, R. y Tábora, A. (2008). Introducción a la computación. México: Pearson Educación.



Conclusiones

Modularización
de programas:
Recursividad

Un módulo
genera una
invocación a sí
mismo.

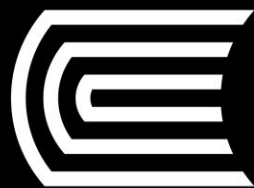
Tiene un punto
de finalización
de la
recursividad.

El módulo de
recursión puede
ser función o
procedimiento.

Programa

```
1  #include <iostream>
2  using namespace std;
3
4  int factorial(int n)
5  {
6      if (n == 0)
7          return 1;
8      else
9          return n*factorial(n-1);
10 }
11
12 int main( )
13 {
14     int num, F;
15
16     do{
17         cout<<"Ingrese numero para factorial: ";
18         cin>>num;
19     }while(num<0);
20
21     F = factorial(num);
22
23     cout<<"El factorial es: " << F;
24
25     return 0;
26 }
```





ucontinental.edu.pe