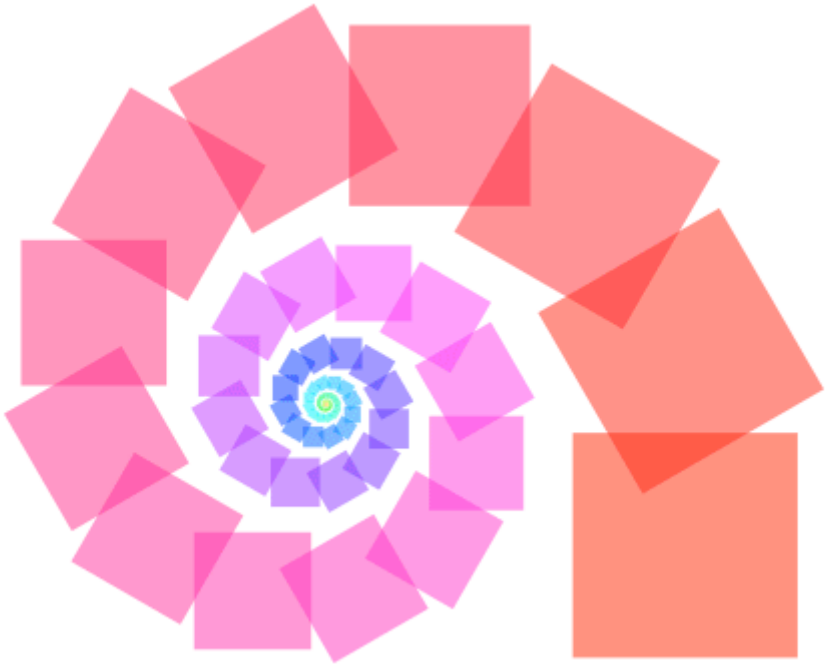
UNIDAD CURRICULAR: ALGORITMICA Y PROGRAMACIÓN

UNIDAD XIII. RECURSIVIDAD

CONTENIDO:



UNIDAD XIII

RECURSIVIDAD

FUNDAMENTOS TEÓRICOS

Una función recursiva es una función que se invoca a si misma directa o indirectamente. Un proceso recursivo debe tener una condición de terminación, ya que no se puede ejecutar indefinidamente.

La recursividad es una herramienta muy útil en aplicaciones de cálculo y en problemas complejos de naturaleza recursiva. Puede ser utilizada como una alternativa a la estructura repetitiva.

VENTAJAS Y DESVENTAJAS DE LA RECURSIVIDAD

Ventajas: existen numerosos problemas complejos que poseen naturaleza recursiva y, en consecuencia, son más fáciles de comprender, depurar e implementar con algoritmos recursivos .

Desventajas: Resulta costoso en tiempo de procesador y espacio de memoria ya que la recursión invoca repetidamente al mecanismo de recursividad y por cada llamada recursiva se produce una copia de las variables de dicha función.

DISEÑO Y ESCRITURA DE PROGRAMAS RECURSIVOS

Cuando se diseña una función recursiva es preciso considerar una condición de terminación, porque de lo contrario la función continuaría indefinidamente invocándose a sí misma hasta que se agote la memoria.

Un ejemplo típico de recursividad es la función que determina el factorial de un número.

Analicemos el cálculo del factorial:

La función factorial se define como:

n! = 1 si n=0

n! = n x (n-1) x (n-2) x (n-3) x …..x 3 x2 x 1 si n>0

Si calculamos 5! daría como resultado: 5 x 4 x 3 x 2 x 1=120

La función recursiva en C++ sería:

```
double factorial(int numero)
{ if (numero>1)
    return numero * factorial(numero - 1);
```

```
    return 1;
  }
```

EJERCICIOS RESUELTOS

a) Realice un programa que utilice una función recursiva para resolver la suma de los primeros n números.

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

double funcionsuma(int numero)
  { if (numero>1)
     return numero+funcionsuma(numero - 1);
     return 1;
    }

int main()
{  int n;
   double resultado;
   cout<<"Suma de números de 1 hasta n"<<endl;
   cout<<"Ingrese el valor de n: ";
   cin>>n;
   resultado=funcionsuma(n);
   cout <<"La suma de los números de 1 hasta "<<n<<" es: "<<resultado;
return 0;}
```

b)Realice un programa que resuelva la siguiente función matemática, a través de una función recursiva.

Q(n)= 1*1+2*2+3*3+4*4+...n*n, para cualquier valor de n .

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

double funcionQ(int numero)
{ if (numero>1)
     return numero*numero + funcionQ(numero - 1);
     return 1;
  }

int main()
{  int n;
   double resultado;
   cout<<"Función Q(n)= 1*1+2*2+3*3+4*4+...n*n"<<endl;
   cout<<"Ingrese el valor de n: ";
   cin>>n;
```

```
   resultado=funcionQ(n);
   cout <<"el valor de la función Q(“<<n<<”)  es"<<resultado;
return 0;}
```

REFERENCIAS BIBLIOGRÁFICAS

Bassard, G y Bratley, P. (2010). Fundamentos de algoritmia. Prentice-Hall.

Joyanes, L. (2008). Fundamentos de programación. Algoritmos , Estructuras de datos y objetos. Mc Graw Hill. Tercera edición.

Joyanes, L. y Zahonero, I. (2005). Programación en C. Metodología, algoritmos y Estructura de datos. Mc Graw Hill. Segunda Edición

Martí, N. y Ortega, Y. (2004). Estructuras de datos y Métodos Algorítmicos. Ejercicios Resueltos. Pearson Education.