Unidad II: Recursividad

2.1 Definición

Recurrencia, recursión o recursividad es la forma en la cual se especifica un proceso basado en su propia definición. Siendo un poco más precisos, y para evitar el aparente círculo sin fin en esta definición:

Un problema que pueda ser definido en función de su tamaño, sea este N, pueda ser dividido en instancias más pequeñas (< N) del mismo problema y se conozca la solución explícita a las instancias más simples, lo que se conoce como casos base, se puede aplicar inducción sobre las llamadas más pequeñas y suponer que estas quedan resueltas.

Para que se entienda mejor a continuación se exponen algunos ejemplos:

- Factorial: Se desea calcular $n!$ (el factorial de n, que se define como el producto de todos los enteros positivos de 1 a n). Se puede definir el problema de forma recurrente como $n(n-1)!$; como $(n-1)!$ es menor que $n!$ podemos aplicar inducción por lo que disponemos del resultado. El caso base es $0!$ que es 1.
- Algoritmo de ordenación por fusión: Sea v un vector de *n* elementos, podemos separar el vector en dos mitades. Estas dos mitades tienen tamaño *n*/2 por lo que porinducción podemos aplicar la ordenación en estos dos subproblemas. Una vez tenemos ambas mitades ordenadas simplemente debemos fusionarlas. El caso base es ordenar un vector de cero o un elemento, que está trivialmente ordenado y no hay que hacer nada.

2.2 Procedimientos recursivos

a) Secuencia Fibonacci:

La secuencia de FIbonacci es aquella secuencia de enteros donde cada elemento es la suma de los dos anteriores.

0 1 2 3 4 5 6 7 8 ..... <--- fib

0,1 ,1 ,2 ,3 ,5 ,8 ,13 ,21 ,... ,

Definicion Recursiva

if n == 0 or n == 1 entonces

fib(n) = n

if n >= 2 entonces

fib(n) = fib(n-2) + fib(n-1) //Llamadas recursivas

Algoritmo de la Secuencia Fibonacci

Funcion FIB(n)

Inicio

Si n == 0 o n == 1 entonces

FIB<--n

Si No

FIB<--Fib(n-2)+FIB(n-1)

Fin_Si

Fin_Funcion

Ejemplo:

n=2

fib(2)=fib(2-2)+fib(2-1)

fib(2)=fib(0)+fib(1)

fib(2)=0+1

fib(2)=1

Procedimientos Recursivos

1) Funcion Factorial

2) Secuencia Fibonacci

3) Torres de Hanoi

4) Busqueda Binaria

b) Torres de Hanoi

Se tienen tres torres y un conjunto de n discos de diferentes tamanos. Inicialmente los discos estan ordenados de mayor a menor en la primer torre.El problema consiste en pasar los discos a la tercer torre utilizando la segunda como auxiliar.

Reglas:

1) En cada movimiento solo puede moverse un disco.

2) No puede quedar un disco mayor sobre uno menor.

Instrucciones (Fig.1)

Mover 1 disco de A a B.

Mover 1 disco de A a C.

Mover 1 disco de B a C.

Algoritmo de las Torres de Hanoi

```
Procedimiento Hanoi(N,ORIGEN,AUXILIAR,DESTINO)
Si N=1 entonces
Escribir "Mover un Disco de",ORIGEN,"a",DESTINO
Si No
regresar Hanoi(N-1,ORIGEN,DESTINO,AUXILIAR) //Llamada recursiva
Escribir "Mover Disco de"ORIGEN,"a",DESTINO
regresar Hanoi(N-1,AUXILIAR,ORIGEN,DESTINO) //Llamada recursiva
Fin_Si
Fin_Procedimiento
```

2.3 Ejemplos de casos recursivos

En el siguiente ejemplo se muetra un procedimiento recursivo.

```
using System;using System.Collections.Generic;using
System.ComponentModel;using System.Data;using System.Drawing;using
System.Text;using System.Windows.Forms;namespace
WindowsApplication1{public partial class Recursividad : Form{public
Recursividad(){InitializeComponent();}double r;int fin = 0;private void
button1_Click(object sender, EventArgs e){fin =
int.Parse(textBox4.Text.ToString());listBox1.Items.Clear();


listBox1.Items.Add("x\ty");evaluar();}//Procedimiento recusivopublic void
evaluar(){while (fin <= int.Parse(textBox5.Text.ToString())){r =
int.Parse(textBox1.Text.ToString()) * (fin * fin) +int.Parse(textBox2.Text.ToString())
* fin +int.Parse(textBox3.Text.ToString());listBox1.Items.Add(fin.ToString() + "\t" +
r.ToString());fin++;evaluar();}}}
}
```