Recursividad

La recursividad (recursión) es aquella propiedad que posee una función por la cual dicha función puede llamarse a sí misma. Se puede utilizar la recursividad como una alternativa a la iteración. Una solución recursiva es normalmente menos eficiente en términos de tiempo de computadora que una solución iterativa debido a las operaciones auxiliares que llevan consigo las llamadas suplementarias a las funciones; sin embargo, en muchas circunstancias el uso de la recursión permite a los programadores especificar soluciones naturales, sencillas, que serían, en caso contrario, difíciles de resolver. Por esta causa, la recursión es una herramienta poderosa e importante en la resolución de problemas y en la programación.

Un subprograma recursivo es un subprograma que se llama a si mismo ya sea directa o indirectamente. La recursividad es un tópico importante examinado frecuentemente en cursos de programación y de introducción a las ciencias de la computación.

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.

Procedimientos recursivos

- Función Factorial
- Secuencia Fibonacci
- Torres de Hanoi

- *Secuencia Fibonacci*:

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

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

Morgado Jacome Eduardo #17211545

Algoritmo de la Secuencia Fibonacci

```
Función FIB(n)
Inicio
Si n == 0 o n == 1 entonces
FIB<--n
Si No
FIB<--Fib(n-2)+FIB(n-1)
Fin_Si
Fin_Función
```

- *Torres de Hanoi*

Se tienen tres torres y un conjunto de n discos de diferentes tamaños Inicialmente los discos están ordenados de mayor a menor en la primer torre. El problema consiste en pasar los discos a la tercera 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.

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
```

Ejemplos de casos recursivos

- 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 por inducción 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.

El factorial de un número.

Ejemplo **Python**:

```python
def factorial(numero):
  if(numero == 0 or numero == 1):
        return 1
    else:
        return numero * factorial(numero-1)
```

Ejemplo **Java:**

```java
public static int factorial(int num){
        if(num == 0){
            return 1;
        }
        else
            return num * factorial(num-1);
```

Ejemplo **Javascript:**

```javascript
function factorial(num)
{
    if (num < 0) {
        return -1;
    }
    else if (num == 0) {
        return 1;
    }
    else {
        return (num * factorial(num - 1));
    }
}
```

Morgado Jacome Eduardo #17211545

Bibliografía

Luis Joyanes Aguilar. (2003). Fundamentos de programación – Algoritmos, Estructuras de datos y Objetos., de Mc Graw Hill

Sin especificar. (2004). Unidad II - Recursividad., del Instituto Tecnológico de Piedras Negras, Sitio web: http://itpn.mx/recursosisc/3semestre/estructuradedatos/Unidad%20I.pdf

Morgado Jacome Eduardo #17211545