Universidad de San Buenaventura

Jefferson Andrey Daza Caro y Alejandro Padilla Guevara

30000048740 - 30000045273

TALLER 4 – ANALISIS DE ALGORITMOS

Algoritmos recursivos

Un algoritmo recursivo es una llamada de una función desde la misma función. Son recursivos aquellos algoritmos que están dentro de ellos mismos y son llamados más de una vez.

Tipos de recursión

Recursión lineal

Si cada llamada recursiva genera como mucho otra llamada recursiva.

Final

Si la llamada recursiva es la última operación que se efectúa, devolviéndose como resultado lo que se haya obtenido de la llamada recursiva sin modificación alguna.

No final

El resultado obtenido de la llamada recursiva se combina para dar lugar al resultado de la función que realiza la llamada.

Recursión múltiple

Si alguna llamada puede generar más de una llamada adicional.

Como diseñar un algoritmo recursivo

- Determinamos parámetros del problema planteado.
- Resolvemos los casos triviales.
- Minimizamos el caso general en términos de un caso más simple.
- Se descompone el algoritmo en caso base y caso general. Caso base cuando se plantea desde el principio y se resuelve sin la recursividad. Caso general cuando el problema es muy complejo y se utiliza en forma recursiva como creación de subprogramas.

Ejemplo algoritmo recursivo

```
static int sumaRecursivaBuena(int valor){
    int resultado=0;
    if(valor==1){
        return 1;
    }else{
        resultado=valor+sumaRecursivaBuena(valor-1);
    }
    return resultado;
}
```

Algoritmos iterativos

Son algoritmos que se caracterizan por ejecutarse mediante bucles o ciclos. Estos algoritmos son muy útiles al momento de realizar tareas repetitivas (como recorrer un arreglo de datos). Los bucles o ciclos son un tipo de estructuras que permiten ejecutar varias veces un conjunto determinado de instrucciones, los cuales son WHILE, DO WHILE y FOR.

Tipos de bucles

While

Es una estructura iterativa que indica un conjunto de instrucciones que se deben repetir mientras que la respuesta a la condición colocada en el lugar del símbolo de decisión sea verdadera, por lo tanto, cuando la respuesta sea falas se termina.

Do-While

Es una estructura iterativa que indica un conjunto de instrucciones que se deben repetir mientras que la respuesta a la condición colocada en el lugar del símbolo de decisión sea falsa, por lo tanto, cuando la respuesta sea verdadera se termina.

For

Esta estructura iterativa indica un rango de valores exacto que una variable tendrá para repetir un conjunto de instrucciones. Las instrucciones por ejecutar que se encuentran dentro del ciclo se ejecutarán mientras la respuesta a la expresión colocada en el símbolo de decisión sea verdadera de lo contrario el ciclo se termina.

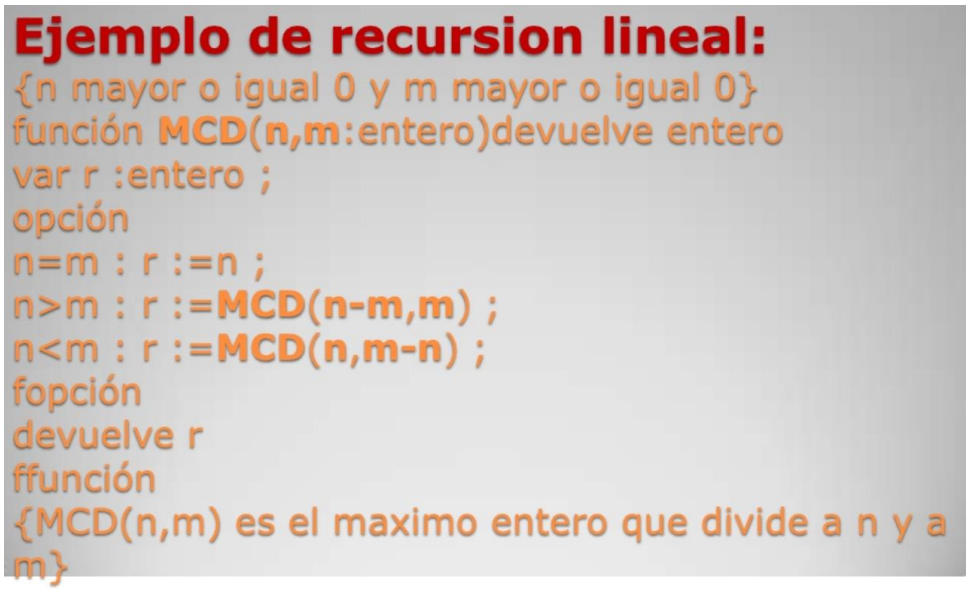
Fases de un diseño iterativo

- Diseñar la variante, para indicar el resultado que se obtendrá a cada paso del bucle.
- Determinar las inicializaciones para que se cumpla la variante.
- Determinar la condición de continuación, que nos permite saber cuando abandonamos el bucle.
- Determinar el cuerpo del bucle, para comprobar que se sigue cumpliendo la variante cada vez.
- Verificar que el bucle termina.
- Razonar sobre el estado obtenido a la terminación y determinar las acciones finales, para que se cumpla la postcondición.

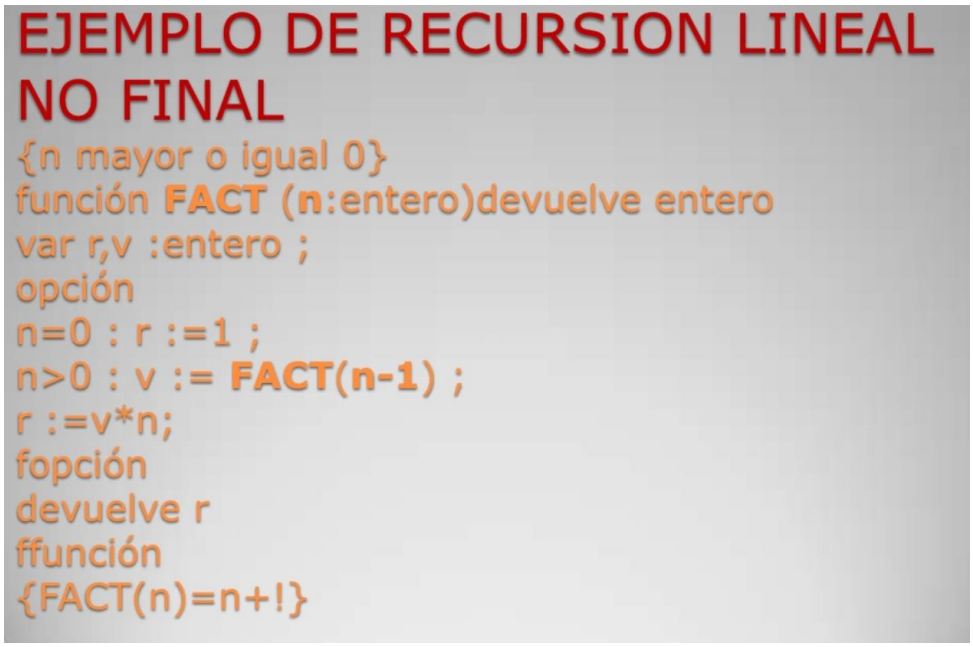
Ejemplo algoritmo iterativo

```
public class Recursividad1 {

    static int sumarIterativa(int a) {
        System.out.println(inventorsuma());
        return a + a;

    }
```

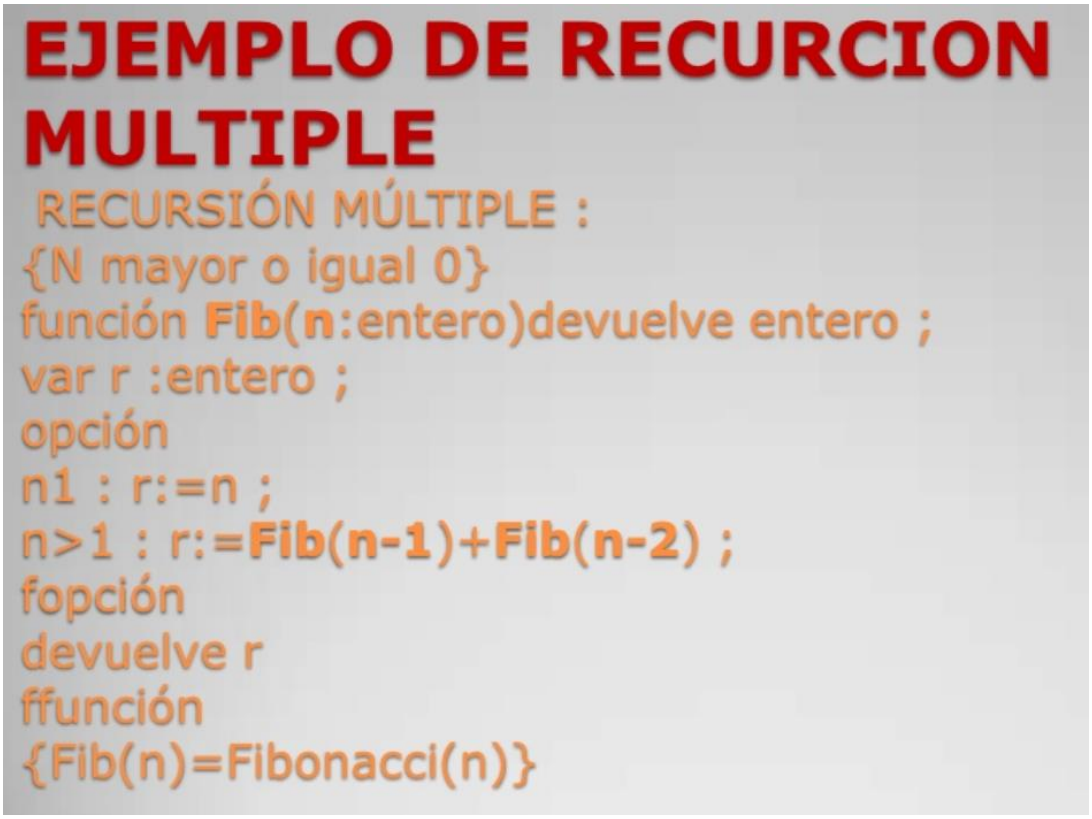
Ejemplo de recursión lineal



Ejemplo de recursión no final



Ejemplo de recursión múltiple



Ejemplo de recursión final

```
int suma(int a, int b);
int suma_im(int a, int b, int res);

int main(){

    int res_suma;

    res_suma = suma(8,25);

    printf("El resulta de la suma es: %i.\n",res_suma);

    system("pause");
    return 0;
}

int suma(int a, int b){
    return suma_im(a, b , 0);
}

int suma_im(int a, int b, int res){
    if(a == 0){
        return res + b; //res = 8 y en b = 10; 18
    }
    else{
        suma_im(a - 1, b, res + 1);
    }
}
```

Referencias

[1] Feytho, «Slideshare,» 8 Julio 2011. [En línea]. Available: https://es.slideshare.net/feytho/algoritmos-recursivos-8544616.

[2] M. G. Harbour, «unican,» 10 Noviembre 2009. [En línea]. Available: https://www.ctr.unican.es/asignaturas/programacion1/cap4-Bloque1-iteracion-recursion-2en1.pdf.

[3] Universidad Veracruzana, «UV,» 12 Febrero 2016. [En línea]. Available: https://www.uv.mx/personal/clgarcia/files/2012/10/IV-Algoritmos-iterativos.pdf.