—Bloque 1—

¿Qué es Recursividad?

* Es la forma en la cual se especifica un proceso basado en su propia definición.

Es una función que se llama a sí misma, ya sea directa o indirectamente a través de otra función.

* Función recursiva.

Es el llamado de una función o programa a si mismo, para no llegar a la inestabilidad del sistema se debe de contar con un estado base que detiene el autollamado.

* La recursión.

Es cuando una función o subprograma se llama a si mismo.

* La recursión directa.

Es cuando una función llama a otra y esta a su vez la llama de nuevo.

* La recursión indirecta.

Es aquella cuyo orden de crecimiento es lineal. El número de llamadas recursivas dependía linealmente de el parámetro n, de la función.

* Recursión lineal.

Creará más de un nodo a cada paso y crecerá muy rápido. Esto hace que se puedan solucionar problemas de un tamaño muy pequeño, y para problemas más grandes, una implementación recursiva no nos sirva.

* Árbol recursivo.

Diseñar un procedimiento recursivo para calcular la suma de los n primeros elementos de la serie.

1 - 1/2 + 1/3 - 1/4 + 1/5 - ... - 1/(2\*n) + 1/(2\*n+1) -

1. Para utilizar este algoritmo, el array debe estar ordenado.

2. consiste en dividir el array por su elemento medio en dos sub-arrays más pequeños, y comparar el elemento con el del centro.

3. Si el elemento es menor, debe estar (si está) en el primer sub-array.

4. y si es mayor está en el segundo.

* Búsqueda binaria

**int** binaria (int\* vec, int valor, int inferior, int superior) {

**int** central;

**if** (superior <= inferior)

**return** -1;

central = inferior + (superior - inferior)/2;

**if** (vec[central] == valor)

**return** central;

**else if** (vec[central] < valor)

**return** binaria(vec, valor, central+1, superior);

**return** binaria(vec, valor, inferior, central-1);

}

Es un juego matemático que consiste en tres varillas verticales y un número indeterminado de discos que determinarán la complejidad de la solución.

* Torres de Hanoi

**void** hanoi(int n, int origen, int auxiliar,int destino) {

**if** (n==1) {

**printf**("Mover de %d a %d\n",origen, destino);

**return**;

} **else** {

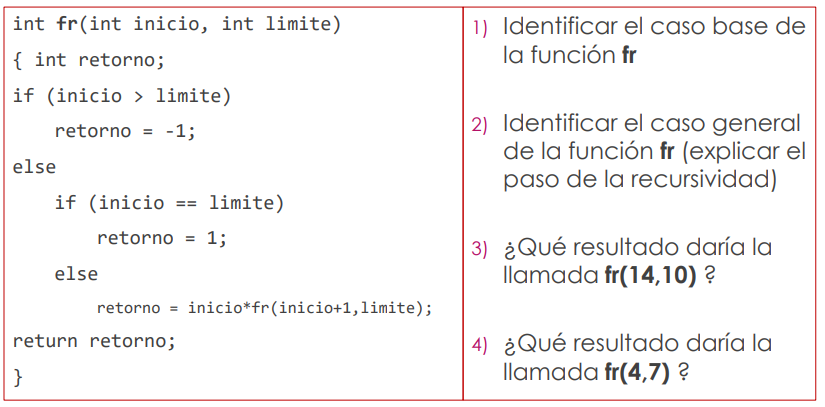
hanoi (n-1, origen, destino, auxiliar);

hanoi (1, origen, auxiliar, destino);

hanoi (n-1, auxiliar, origen, destino);

}

}

1)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Es la organización de la información en una forma que va mas allá que la definición misma de un tipo de dato.

* Estructuras de datos

Pueden contener diferentes tipos o ser del mismo tipo y en forma abstracta representar algo diferente.

* Las estructuras de datos

Cada bit que represente un dato puede contener solo dos estados de información 0 y 1, entonces un grupo de bits puede almacenar 2 a la n estados.

* Memoria

Determinan la razón de incremento de los punteros e índices según el tipo de dato con el que se asignaron las variables.

* los compiladores

Son grupos de datos del mismo tipo con el mismo nombre y distinguidos por un índice.

* Arreglos

//Me quedé em página 14 de “03 Memoria Arreglos Estructuras”