**1.Define el problema**

Necesitamos ordenar una secuencia de números en orden creciente.

Ejemplo:

2 6 7 8 4 3 1 5

1 2 3 4 5 6 7 8

ENTRADAS:

Una secuencia de n números

Ejemplo: 1, 10, 3, 65, 5, 7, 90

SALIDA:

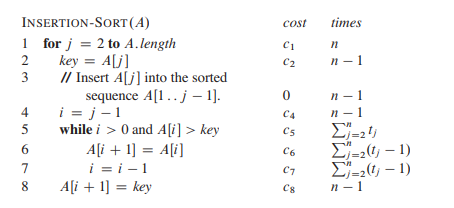
Una permutación (reordenamiento de los números) de tal forma que

Ejemplo: 1, 3, 5, 7, 10, 65, 90

**2.Presenta una propuesta de solución**

El tipo de inserción funciona de la misma manera que mucha gente clasifica una mano de cartas. Comenzamos con una mano izquierda vacía y las cartas boca abajo en la mesa. Luego retiramos una carta a la vez de la mesa y la insertamos en la posición correcta en la mano izquierda. Para encontrar la posición correcta de una carta, la comparamos con cada una de las cartas que ya están en la mano.

La ejecución de los procesos depende del procesador, recursos de la memoria, el número de operaciones que se realizaran y en este caso del tamaño del arreglo puesto que si es menor el arreglo será menos tiempo, menos operaciones y por el contrario entre más grande sea el arreglo más operaciones y más tiempo tardara en ejecutar y realizar el ordenamiento de los números, cada instrucción y cada operación es un tiempo de ejecución y siempre dependerá de la entrada de los datos el tamaño y número de operaciones a ejecutar.



1.- Indica la longitud para j que es de 2 a n veces, indica el número de veces que se ejecutara

2.- Indica el tamaño del arreglo -1 puesto que va tomando valor de 1 en 1 para el ordenamiento

3.-Comentario con costo de 0 puesto que no se ejecutan, pero como forma parte del programa se le resta el valor de 1 a n

4.- Se le resta 1 al valor de j para obtener el valor de i.

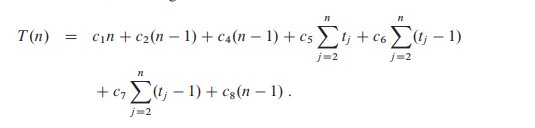
5.- Es la sumatoria del tamaño del arreglo que va desde 2 a n veces (j es el tamaño del arreglo). Es la suma de los tiempos de ejecución para cada declaración indica el total de veces que se ejecuta.

6.-El costo es la sumatoria del arreglo que es desde 2 a n veces -1 para ir quitando cada valor que se va ordenando y va tomando el lugar de i

7.- Es la sumatoria del arreglo y n veces y ese valor es restado por 1 para ir ocupando su lugar de orden y queda guardado en i

8.- es el valor de n veces -1 que por ultimo quedara el arreglo en orden secuencial.

Para calcular el tiempo de ejecución de ordenamiento de n entradas se suma el costo de los productos y el tiempo de las columnas obteniendo:



El tiempo en que corre dependerá de cuál es el tamaño de la entrada.

En el mejor de los casos ocurre cuando el arreglo esta ordenado y el cálculo para saber el tiempo de ejecución es el siguiente:



Lo anterior da una función lineal

Como conclusión tenemos que cada elemento del arreglo se va comparando y se va ordenando en el mismo. En una inserción de ordenamiento el tiempo de ejecución de un algoritmo será fijado por la entrada.

**3.Diseña un algoritmo**



**4.Convierte el algoritmo en un programa**

**package** com.tcca;

**public** **class** Insercion {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**int** cartas[] = { 1, 10, 3, 65, 5, 7, 90}; // el arreglo se llama cartas

**int** pos; // pos =posicion

**int** val; // val= valor

// mostrar cartas desordenadas

System.***out***.print("El desorden de las cartas es: ");

**for** (**int** i=0; i<cartas.length; i++)

{

System.***out***.println(cartas[i]);

}

//ordenamiento de las cartas

**for** (**int** i=0; i<cartas.length; i++) {

pos=i; val= cartas[i];

**while**( (pos>0) && (cartas [pos-1] > val)) {

cartas[pos] = cartas [pos-1];

pos --;

cartas[pos] = val;

//imprime los valores ordenados

System.***out***.print("El orden de las cartas es: ");

**for** (**int** i1=0; i1<cartas.length; i1++){

System.***out***.println(cartas[i1]);

}

}

}

**5.Verifica la solución**

