ASIGNACIÓN
CIENCIAS DE LA COMPUTACIÓN II

METODOS DE ORDENAMIENTO C++

-GRUPO NRO 1-

INTEGRANTES

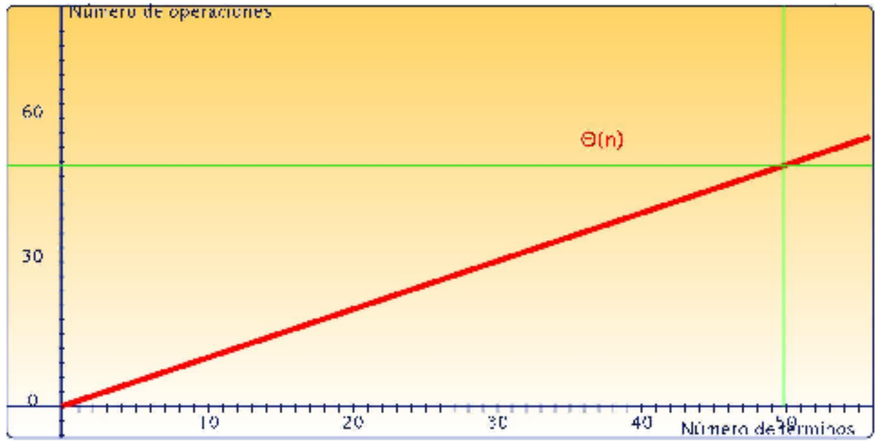
- MIGUEL ANGEL DEZA CUELA
- RICARDO ALEXANDER RODRIGUEZ PUMACAYO
- YANIRA ANGIE SUNI QUISPE
- KATHERINE NIKOLE BÉJAR ROMÁN
- JOSUE GABRIEL SUMARE USCCA

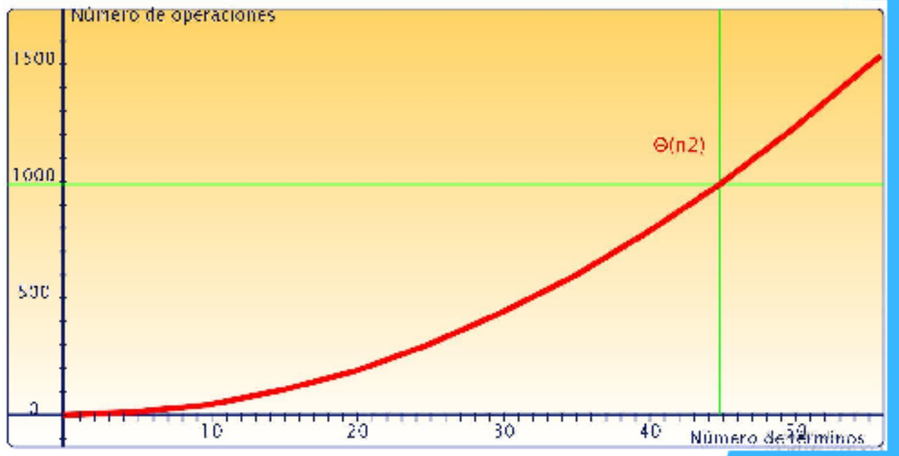
“INSERTION SORT”

Es una manera muy natural para el ser humano ,de ordenar y es muy util o facilmente identificable haciendo una analogia de la manera en que se ordena un mazo de cartas .

6 5 3 1 8 7 2 4

Requiere de O(n^2) operaciones para ordenar una lista de n elementos, en el caso medio o un caso desfavorable, y en el caso óptimo O(n).



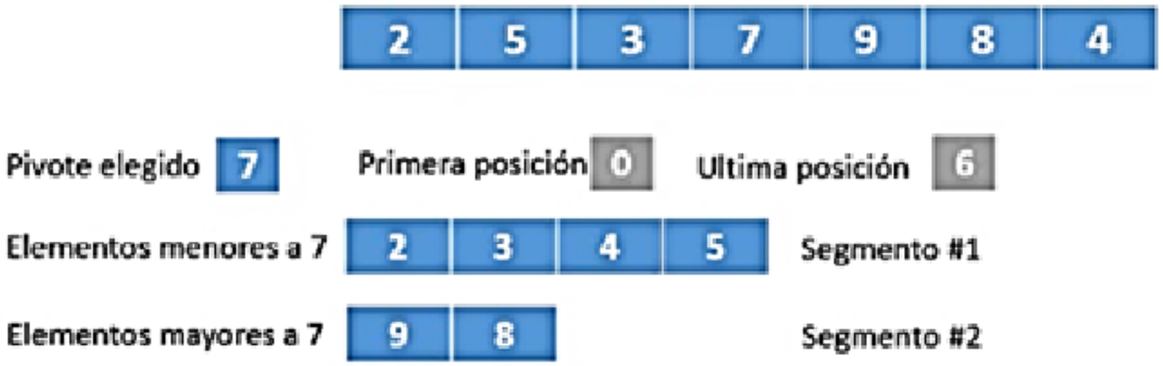


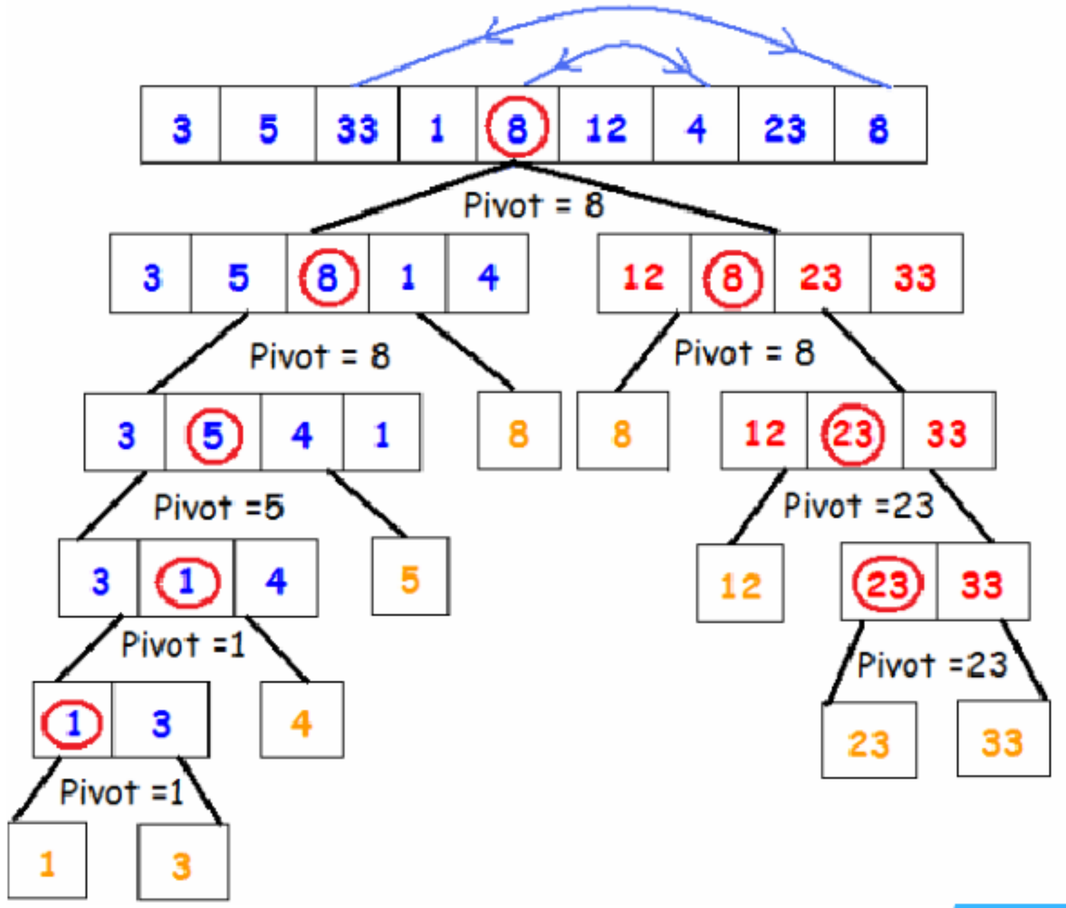
"QUICK SORT"

Quicksort es un algoritmo de ordenación considerado entre los mas rápidos y eficientes, este es un algoritmo que consiste en dividir.

Se selecciona un elemento como pivote y alrededor de esta se agrupan los elementos mayores y menores al pivote, hay muchas versiones diferentes de quicksort que selecciona el pivote de diferente manera.

En este caso el algoritmo de quicksort selecciona al elemento del centro como pivote para dividir en dos segmentos

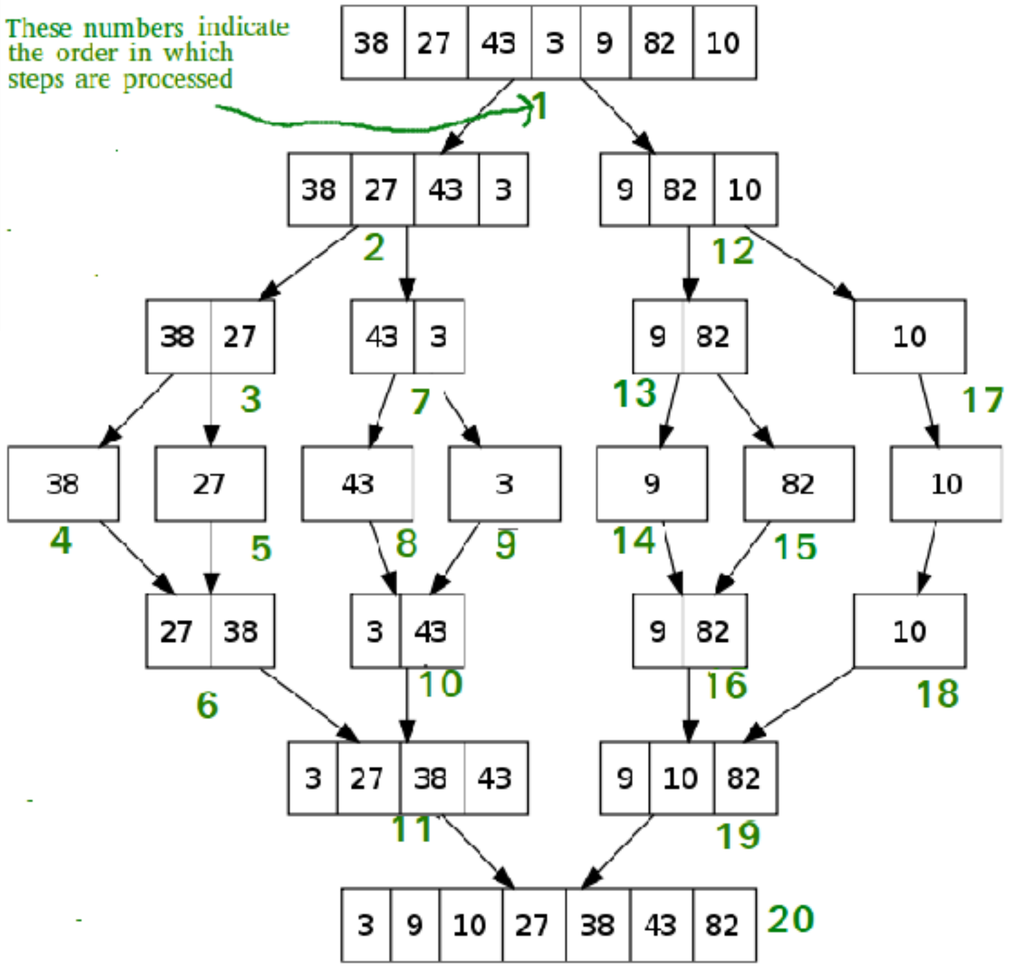




"MERGE SORT"

```
MergeSort (arr [], l, r)
Si r> l
     1. Encuentre el punto medio para dividir la matriz en dos mitades:
            medio m = l + (rl) / 2
     2. Llame a mergeSort para la primera mitad:
            Llamar mergeSort (arr, l, m)
     3. Llame a mergeSort para la segunda mitad:
            Llamar mergeSort (arr, m + 1, r)
     4. Combine las dos mitades ordenadas en los pasos 2 y 3:
            Combinación de llamadas (arr, l, m, r)
```

Merge Sort es un algoritmo de dividir y conquistar . Divide la matriz de entrada en dos mitades, se llama a sí mismo para las dos mitades y luego fusiona las dos mitades ordenadas. La función merge () se usa para fusionar dos mitades, De ahi el nombre.



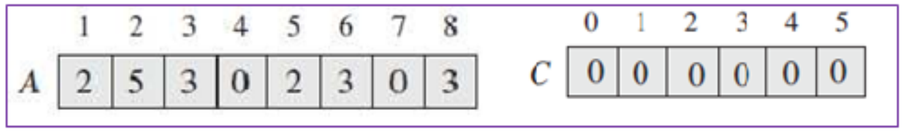
"COUNTING SORT"

Es un algoritmo que no se basa en comparaciones y lo que hace es contar el número de elementos de cada clase en un rango de 0 - k para después ordenarlos determinando para cada elemento de entrada el número de elementos menores a él. Por lo tanto, la lista o arreglo a ordenar solo pueden utilizar elementos que sean contables (enteros).

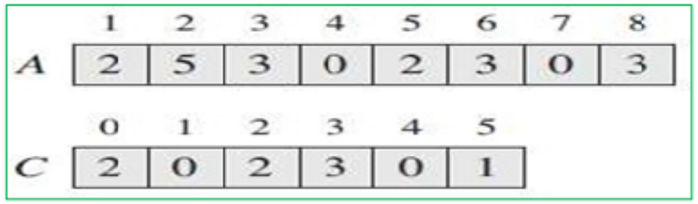
Para la descripción del algoritmo se asumen 3 arreglos lineales:

- El arreglo de entrada A a ordenar con n elementos.
- Un arreglo B de n elementos, para guardar la salida ya ordenada.
- Un arreglo C para almacenamiento temporal de k elementos.

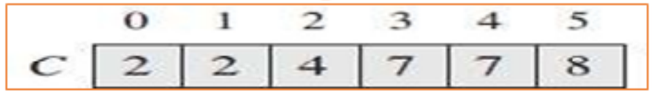
1. En el arreglo A de la figura se tiene que el rango de valores de los elementos es de 0 a 5, por lo tanto, el arreglo C tendrá 6 elementos con índices de 0 a 5 el cual se inicializa en 0.



2. Después se recorren todos los elementos del arreglo A a ordenar y se cuenta el número de apariciones de cada elemento para almacenarlo en C.

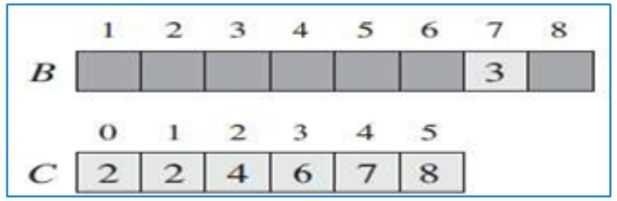


3. Posteriormente se determina para cada elemento C[i] (i=0,…,k) cuántos elementos son menores o iguales a él.

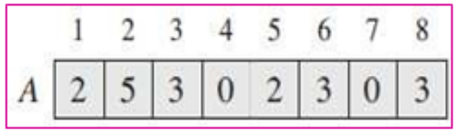


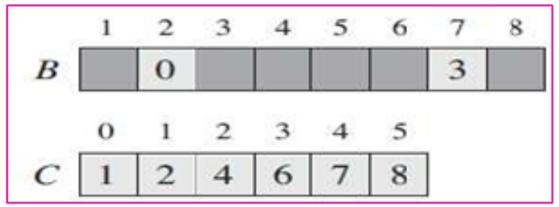
4. Para terminar, se coloca cada elemento del arreglo A, (A[j], j = n, … 1) en la posición correcta en el arreglo de salida B, de tal forma que cada elemento de entrada se coloca en la posición del número de elementos menores o iguales a él (B[C[A[j]]]). Cada vez que se coloca un elemento en B, se decrementa el valor de C[A[j]].

Así el elemento A[8] = 3 se coloca en B[C[A[8]] = B[C[3]] = B[7] .Figura 3.4.



5. El elemento A[7] se coloca en B[C[A[7]] = B[C[0]] = B[2] ..





6. Una vez que se revisan todos los elementos de A tenemos que el arreglo ordenado de B.

"RADIX SORT"

La idea en Radix Sort es ordenar digito por digito.

Este orden se hace desde el digito menos significativo al mas significativo.

Ademas se usa la ordenación por conteo como subrutina para ordenar.

Un numero es menos significativo mientras mas cerca este este del '0'.

Un problema de radix sort es su velocidad, ya que no es tan veloz como algunos algoritmos basados en comparacion como Quick sort.

Ejemplo:

170	45	75	90	802	24	2	66

Primero verificamos las unidades y ordenamos segun que tan significativos sean.

170	90	802	2	24	45	75	66

Observamos que "802" esta antes del "2". La razón es porque "802" sucedio antes que "2" en la lista original.
Pasa algo similar con "170" y "90" o con "45" y "75".

Segundo verificamos las decenas y ordenamos nuevamente.

802	2	24	45	66	170	75	90

Observamos que "802" esta antes que "02" ya que "802" sucedio antes que 2 en la anterior lista.

Por ultimo verificamos las centenas y ordenamos.

2	24	45	66	75	90	170	802

ORDENADO

“TAMPOCO ES INESCRUTABLE EL AZAR, TAMBIÉN ESTÁ REGIDO POR UN ORDEN.”

MUCHAS GRACIAS