Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria en Ingenierías y Tecnologías Avanzadas

Unidad de Aprendizaje: Estructura de Datos // Grupo: 1TV5

Miembros del equipo:
Cabrera Vázquez Itzel Berenice
Martín Moreno Cesar Sadrak
Rodríguez Márquez José Antonio

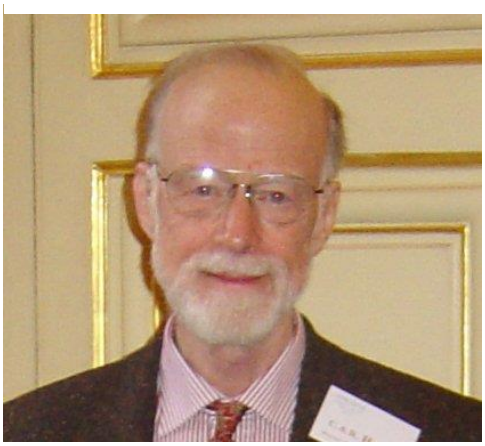
ORDENAMIENTO QUICKSORT

Profesor: *GALVAN RODRIGUEZ JULIO ESTEBAN*

Fecha de entrega: 29 de octubre de 2020

¿QUÉ ES?

Ilustración 1 Tony Hoare



El ordenamiento rápido o también conocido como *quicksort* fue creado por Tony Hoare en 1960. Este método de ordenación es de tipo indirecto (o avanzado) y es el algoritmo de ordenamiento más utilizado debido a que se le considera muy rápido y eficiente.

Se podría decir que el algoritmo aplica la técnica “divide y vencerás”, esto debido a que se basa en la división en sublistas de la lista inicial a ordenar. Este método tiene tres principales elementos: una división izquierda, un elemento llamado pivote y una división derecha.

Siendo el caso en que la lista se desee ordenar de forma ascendente, a continuación, se describen los pasos del algoritmo.

Este al ser uno de los métodos de ordenamiento son de los más eficaces pero que ocupan un poco más de espacio en la memoria.

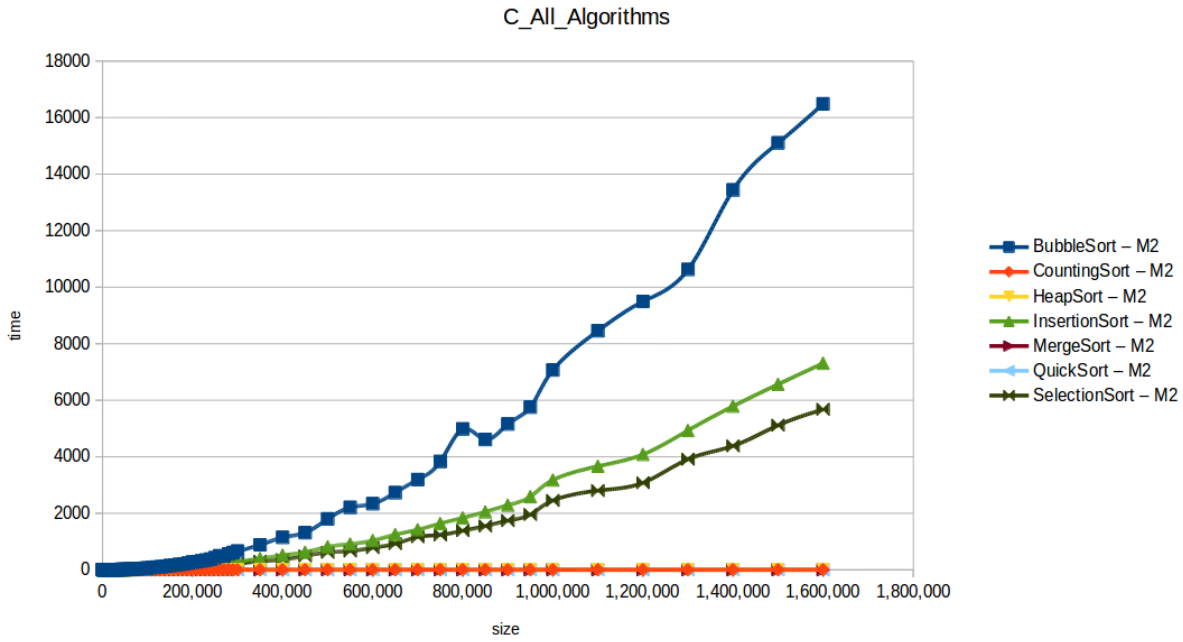


Ilustración 2 Comparación con otros métodos

PASOS DEL ALGORITMO QUICKSORT

I. Seleccionar el elemento pivote

II. Dividir los elementos restantes en divisiones izquierda y derecha. Asegurándose de que ningún elemento de la sublista izquierda tenga un valor mayor que el pivote y que ningún elemento a la derecha tenga un valor más pequeño que el del pivote.

Utilizando 2 índices, i -> contador por la izquierda y j -> contador por la derecha.

A. Recorrer la lista simultáneamente con i y j: se recorre por la izquierda desde el primer elemento con i y se recorre por la derecha desde el último elemento con j.
B. Cuando el valor del elemento con índice i sea mayor que el valor del pivote **y** el valor del elemento con índice j sea menor, se intercambian las posiciones de estos elementos.
C. Se repiten estos pasos hasta que se “crucen” los índices; la posición donde se cruzan es la posición en donde se realizará la posición de las dos sublistas

III. Ordenar la sublista izquierda y la derecha utilizando *Quicksort* recursivamente

IV. La solución final es la concatenación de la sublista izquierda, el pivote y la sublista derecha.

* El paso 3 se repite hasta que todas las sublistas contengan un solo elemento

APLICACIONES DEL MÉTODO SON:

I. Para ordenar de forma ascendente o descendente vectores grandes.
II. Los programadores muchas veces trabajan con registros de datos; suele ser más fácil manejar estos datos si están ordenados.
III. Para realizar búsquedas binarias, ya que para ello se requiere una lista (o vector) ordenado
IV. En la estadística, es útil para encontrar la mediana de un conjunto de datos que inicialmente estaban desordenados

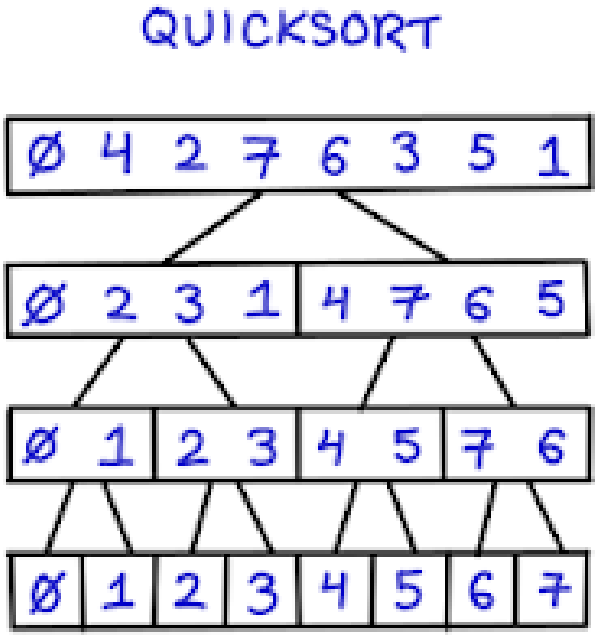
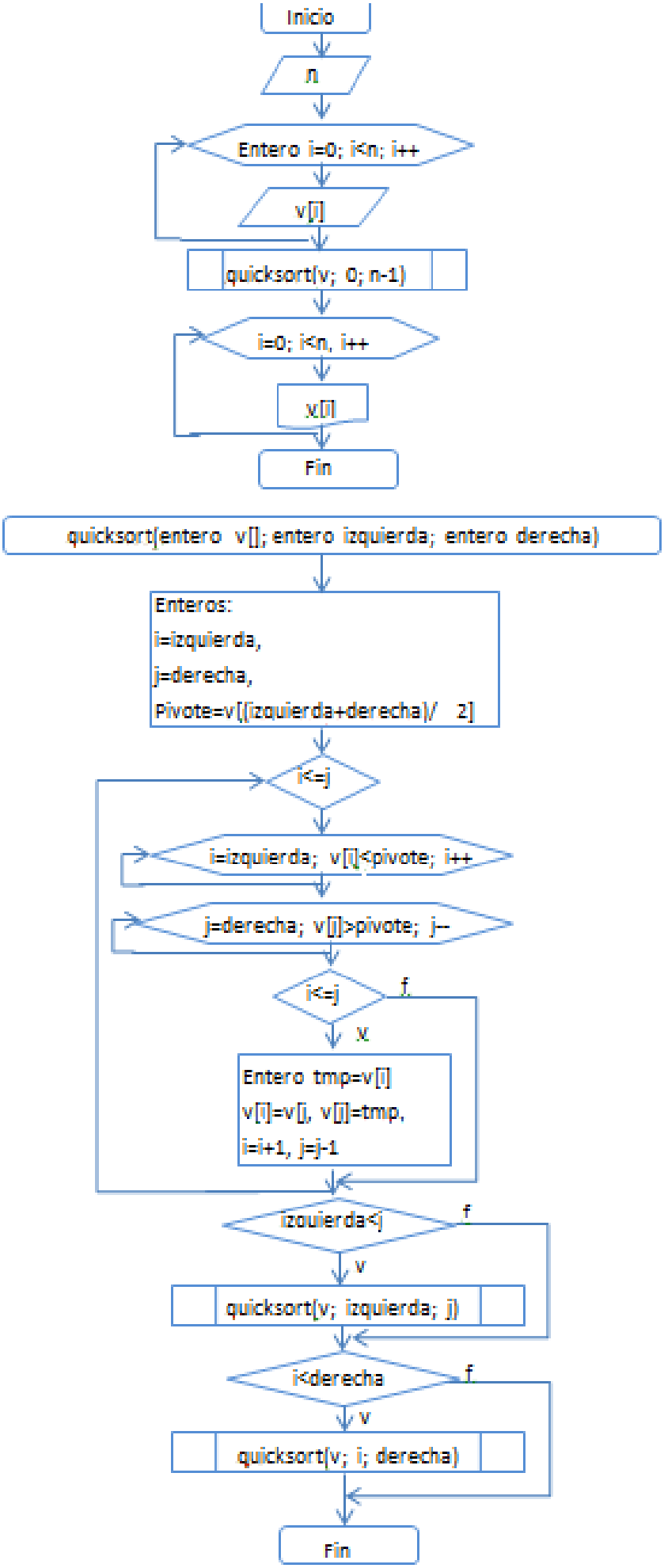
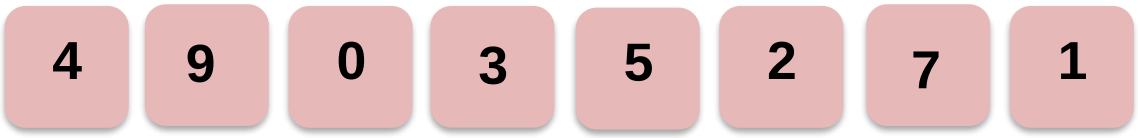


DIAGRAMA DE FLUJO DEL ORDENAMIENTO QUICKSORT

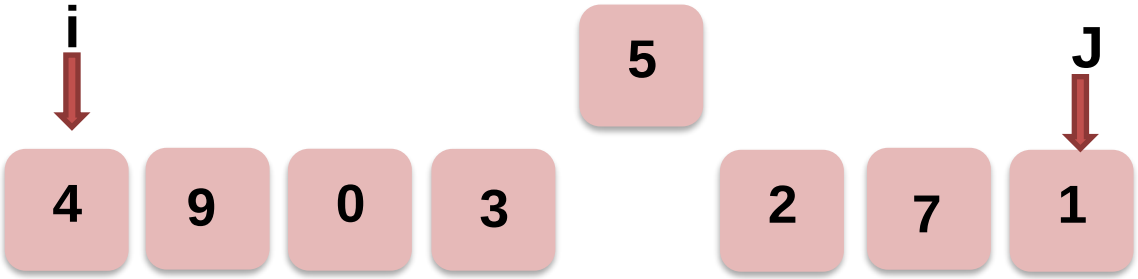


EJEMPLO DEL MÉTODO QUICKSORT

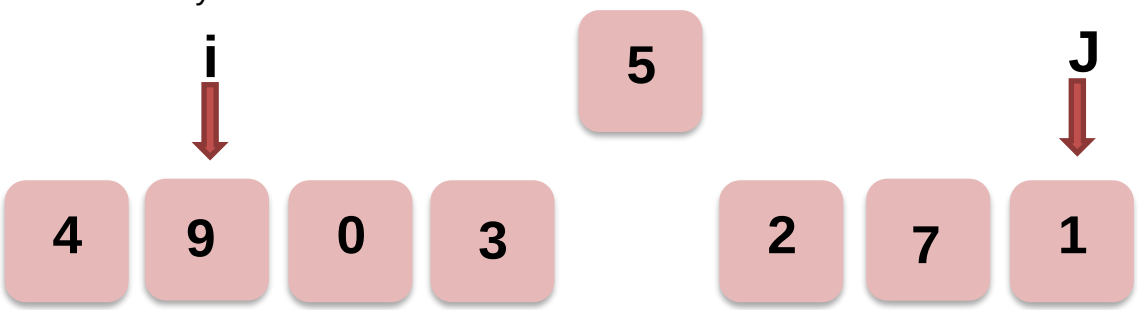
Se presentará un ejemplo del ordenamiento con una serie de 8 dígitos cualquiera sin ningún orden especifico o secuencia.



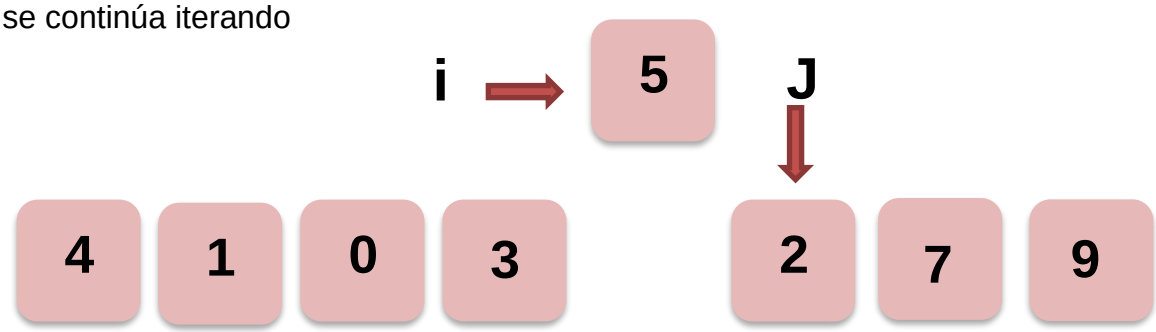
1.- Se selecciona un pivote al azar, en este caso se selecciona el numero 5 y nuestros iteradores empezaran en el 4 y 1



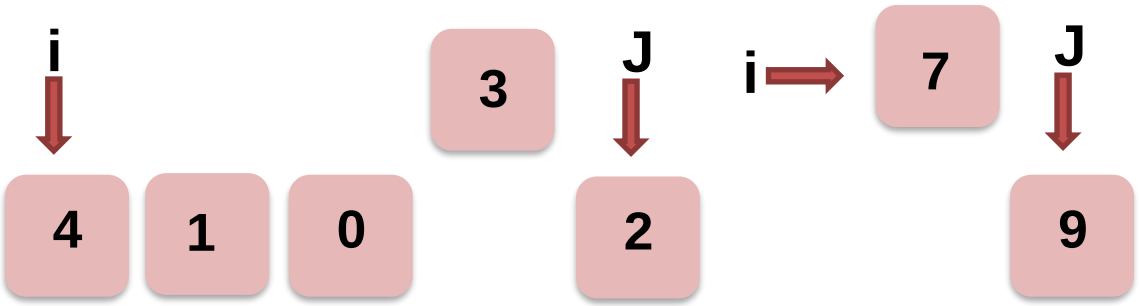
2.- Se hace la comparación y se recorre primero i hasta encontrar un numero mayor a 5



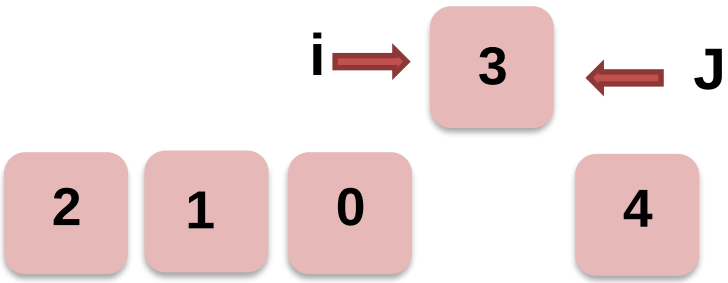
3.-Se llego al número nueve que es mayor, por lo cual ahora se pasa a iterar el índice J el cual es menor que el 5, por lo que pasamos a intercambiar los índices y se continúa iterando



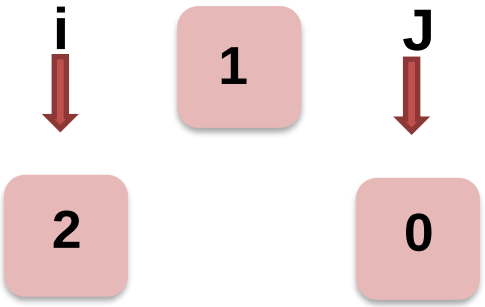
4.- Como se llego a nuestro pivote y este no es mayor a si mismo se coloca del lado derecho el 2, donde fue que se quedó nuestro iterador J
Y ahora se empieza a iterar cada parte por separado, escogiendo nuestro pivote y sus respectivos iteradores



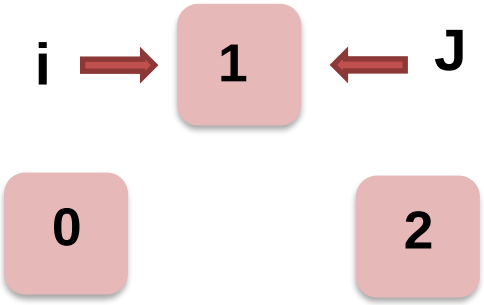
5.- Se itera la parte derecha hasta tener en su respectivo lado ordenado, como en la parte derecha solo había dos dígitos, y nuestro pivote es menor que el índice J, se deja tal cual esta y ese lado está listo



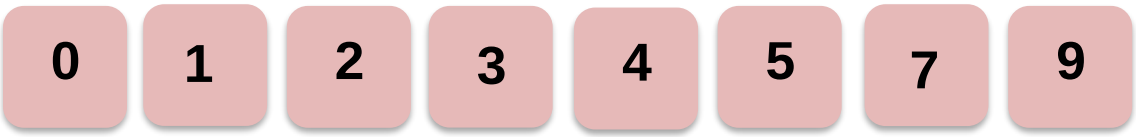
6.- Una vez que los iteradores se encuentran se pasa ahora a ordenar el lado izquierdo donde aún quedan dígitos, mientras tanto en lado derecha esta listo y se puede concatenar a nuestro pivote de valor 3



7.- Se iteran los dos dígitos y como en este caso i es mayor que el pivote y J es menor, estas se intercambian y llega al pivote por lo que aquí termina de ordenar y solo se concatenan los pivotes con sus respectivos valores mayores y menores



8.- Se fueron concatenando recursivamente los pivotes de la forma Menor + Pivote + Mayor, por lo cual al llegar al pivote inicial se obtiene la secuencia de números ordenada



BIBLIOGRAFÍA:

- **Joyanes Aguilar, L. and Zahonero Martínez, I., 2000. *Programación En C, C++, Java Y UML (2A. Ed.)*. 2nd ed. McGraw-Hill Interamericana, pp.261-271.**
- **Hidalgo, J., 2020. *Algoritmos De Ordenamiento. Capítulo Quicksort*. [online] C.conclase.net. Available at: <http://c.conclase.net/orden/?cap=quicksort> [Accessed 12 October 2020].**
- **Deitel, Harvey M., Paul J. Deitel, 2008, *Como programar en C++ Sexta Edición, Pearson Educación, pp 776 - 778***