Algoritmos de ordenamiento

TD3

2º semestre de 2024

Repaso

Caracterizar los algoritmos vistos en la clase teórica:

- Selection Sort,
- Insertion Sort,
- Bubble Sort,
- Merge Sort,
- Quick Sort,
- Counting Sort.

1) Análisis de algoritmos

Decidir qué algoritmo de los vistos en la clase teórica es conveniente utilizar bajo las siguientes suposiciones. ¿Sería útil utilizar una versión modificada o solo una porción de dichos algoritmos?

- Ordenar una secuencia ya ordenada;
- insertar k elementos en su posición en una secuencia v ordenada, con k significativamente más chico que $|v|$;
- encontrar los k elementos más chicos de la secuencia v, con k significativamente más chico que $|v|$.

Más escenarios

- Dadas dos secuencias ordenadas, devolver una secuencia que contenga sus elementos ordenados;
- ordenar una secuencia que está ordenada de forma decreciente;
- encontrar los k elementos más grandes de una secuencia v, con k significativamente más chico que $|v|$;
- ordenar una secuencia v en el que sus elementos están desordenados en a lo sumo k posiciones, con k significativamente más chico que $|v|$.

2) Secuencia de cuentas

Definimos a una secuencia de enteros como "*de cuentas*" si sus elementos aparecen tantas veces en la secuencia como su valor lo indique.

Por ejemplo, {4, 1, 5, 5, 4, 4, 4, 5, 5, 5} es una secuencia "*de cuentas*".

Escribir un programa que determine si una secuencia de enteros cualquiera *v* es o no "*de cuentas*" con tiempo de ejecución de peor caso perteneciente a O(|*v*|). Justificar por qué este algoritmo cumple con dicha complejidad.

3) Más cercanos

Escribir un programa que, dado una secuencia de enteros v y dos enteros e y k (con $k < |v|$), devuelva los k números más cercanos a e. En caso de empates, considerar el de menor valor. Calcular el tiempo de ejecución de peor caso.