

# COMPARACIÓN **ALGORITMOS DE** ORDENAMIENT

# Cuál sería el más indicado para implementar

CONSIDERACIÓN INICIAL: USUALMENTE, AL MOMENTO DE COMPARAR <u>ALGORITMOS. SE CAE EN LA ERRÓNEA CREENCIA DE OUE LA EFICIENCIA SE</u> MIDE **EXCLUSIVAMENTE** EN VELOCIDAD. **PARA EVITAR** ESTO, CONTINUACIÓN SE DAN VENTAJAS Y DESVENTAJAS PARA LOS ALGORITMOS ANALIZADOS, CON EL FIN DE OFRECER UN MEJOR PANORAMA QUE TRASPASE LOS LIMITES DE LA VELOCIDAD COMO ÚNICA VARIABLE **DECISORIA.** 

# **QUICKSORT**



- El algoritmo más rápido posible en su mejor caso
- Su tiempo de ejecución arrasa el de los algoritmos iterativos (arreglos grandes).



- Implementación sencilla.
- Amplio código fuente funcional gratuito en la web.



La recursividad hace mucho uso de memoria por el apilamiento de procesos en stack



- Su eficiencia promedio (Big O notation) es logarítimca
- Ideal para grandes volúmenes de datos
- La probabilidad de que su eficiencia no sea la esperada disminuye conforme aumenta el tamaño del arreglo.

#### SELECTION SORT



Si el tamaño del arreglo es muy grande, el tiempo de ejecución se hace abismal



- Implementación sencilla.
- Amplio código fuente funcional gratuito en la web.



No necesita memoria temporal adicional



- Eficiencia promedio de orden cuadrático (de las peores)
- Cantidad inoficiosa de operaciones para arreglos ordenados a priori o en alguna iteración posterior
- Cantidad de reposiciones menor respecto al bubbesort, pues necesariamente no deben ser adyacentes

### **BUBBLE SORT**



Si el tamaño del arreglo es muy grande, el tiempo de ejecución se hace abismal



- Implementación sencilla.
- Amplio código fuente funcional gratuito en la web.



No necesita memoria temporal adicional



- Eficiencia promedio de orden cuadrático (de las peores) Cantidad inoficiosa de operaciones para arreglos ordenados a priori o en alguna iteración posterior
- Cantidad de reposiciones abismal, puesto que solo pueden ser adyacentes

#### **BUBBLE SORT FIXED**



- Si el tamaño del arreglo es muy grande, el tiempo de ejecución se hace abismal El tiempo de ejecución disminuye con respecto al bubblesort clásico para casos de ordenamiento
- alcanzado en etapa posterior (si aplica).



- Implementación sencilla.
- Amplio código fuente funcional gratuito en la web.



No necesita memoria temporal adicional

100

QuickSort

SelectionSort

Eficiencia promedio de orden cuadrático (de las peores)



10

- Disminuye la cantidad de operaciones al verificar ordenamiento en arreglo
- Cantidad de reposiciones menor respecto al bubbesort, pues necesariamente no deben ser adyacentes

## Comparación cantidad de operaciones N° Operaciones por algoritmo por prueba

1000

■ BubbleSort Fixed

10000

+1000000