UT9 – PD4

**1)**

**Secuencia de incrementos de Sedgewick**:

Sedgewick propuso una secuencia de incrementos que combina potencias de 2 y números de la forma 9⋅4^k−9⋅2^k+1. Esta secuencia tiende a ser bastante efectiva en la práctica y es ampliamente utilizada.

**Secuencia de incrementos de Ciura**:

Ciura propuso una secuencia de incrementos basada en observaciones empíricas de varios algoritmos de ordenación, incluyendo ShellSort. La secuencia de Ciura es 1,4,10,23,57,132,301,701. Esta secuencia ha demostrado ser muy eficiente para una amplia gama de tamaños de datos.

**Secuencia de incrementos de Hibbard**:

Hibbard propuso una secuencia de incrementos que son potencias de 2 menos 1 (2^k−1). Esta secuencia es sencilla y fácil de implementar, pero puede no ser tan eficiente como las anteriores en algunos casos.

Los incrementos recomendados para el algoritmo de ShellSort suelen derivarse de análisis matemáticos. Las secuencias de Sedgewick, Ciura y Hibbard son algunas de las que note son más usadas.

**2)**

**Formas de elegir pivote:**

**Pivote Aleatorio**:

* Seleccionar un pivote aleatorio puede servir para reducir la probabilidad de los peores casos. Esto es útil especialmente en entradas ya ordenadas o casi ordenadas.
* **Fuente**: Este método ha sido estudiado ampliamente y es reconocido por su capacidad para mejorar el rendimiento general del algoritmo​ (Amanote)​.

**Mediana de Tres:**

Se selecciona el pivote como la mediana del primer, último y elemento central del array. Esta técnica ayuda a evitar malos casos y proporciona un mejor balance en la partición.

* **Fuente**: Implementaciones modernas, como en las bibliotecas de Java y C++, utilizan esta técnica por su eficacia en diferentes escenarios de datos​ (SpringerLink)​.

**Mediana de Nueve**:

* Toma la mediana de tres grupos de tres elementos cada uno (inicio, medio y fin del array). Este método ofrece una mejora adicional al reducir aún más la probabilidad de particiones desbalanceadas.
* Fuente: Estudios probaron que la mediana de nueve puede ofrecer ventajas adicionales en términos de rendimiento, aunque es compleja de implementar​ (Amanote)​.

**Implementaciones en lenguajes modernos**:

1. **Java**:
   * Desde Java 7, se utiliza el Dual-Pivot QuickSort en la implementación de Arrays.sort(). Esta variante es más eficiente en la práctica, especialmente para datos aleatorios.
   * **Fuente**: La documentación y los estudios sobre la implementación de Java han respaldado el uso de esta técnica debido a su eficiencia mejorada​ ([SpringerLink](https://link.springer.com/chapter/10.1007/978-981-15-1480-7_33))​.
2. **Python**:
   * Python utiliza una combinación de TimSort, que es una mezcla de MergeSort y una variante de QuickSort. Para QuickSort específicamente, se suele usar la mediana de tres como técnica de pivote.
   * **Fuente**: La documentación de Python y estudios sobre su implementación destacan la robustez y eficiencia de esta combinación para manejar diferentes tipos de datos​ (Amanote)​.
3. **C++**:
   * La implementación estándar de std::sort() en C++ emplea QuickSort con mejoras como la mediana de tres y pivot aleatorio, asegurando un rendimiento robusto y eficiente en varios casos de uso.
   * **Fuente:** La documentación y pruebas empíricas en C++ STL confirman la efectividad de estas técnicas en su implementación​ ([SpringerLink](https://link.springer.com/chapter/10.1007/978-981-15-1480-7_33))​.

**3)**

**Texto

Descripción generada automáticamente**

Análisis del Tiempo de Ejecución:

Ordenar el conjunto A:

Utilizando QuickSort o MergeSort, la complejidad es 𝑂(𝑚log𝑚)

Buscar elementos de 𝐵 en 𝐴: utilizando búsqueda binaria:

La búsqueda binaria tiene una complejidad de 𝑂(log𝑚) por búsqueda.

Hacer esto para cada uno de los 𝑛elementos en 𝐵resulta en 𝑂(𝑛log𝑚.

Complejidad total del algoritmo:

O(m logm) + O(n logm) = O((m+n)logm)

Cuando 𝑚 es sustancialmente menor que 𝑛, la complejidad 𝑂((𝑚+𝑛)log𝑚) sigue siendo eficiente ya que log𝑚 es pequeño en comparación con 𝑚 y 𝑛. Este enfoque aprovecha el hecho de que ordenar un conjunto pequeño y realizar búsquedas binarias en él es más eficiente que otras posibles combinaciones de estrategias.