**Algoritmos de ordenación Shellsort, Heapsort y MergeSort para C con memoria dinámica**

**Explicación de cada algoritmo:**

**1. Shellsort:**

* **Descripción:** Este algoritmo mejora el ordenamiento por inserción insertando elementos a mayor distancia entre sí, lo que reduce el número de comparaciones necesarias.
* **Funcionamiento:**
  1. Se define un intervalo inicial (distancia entre elementos a comparar).
  2. Se ordena el arreglo utilizando el ordenamiento por inserción con el intervalo definido.
  3. Se reduce el intervalo y se repiten los pasos 1 y 2 hasta que el intervalo sea 1.
* **Memoria dinámica:** Se utiliza memoria dinámica para almacenar el arreglo y el intervalo.

**Código:**

Texto

Descripción generada automáticamente

**2. Heapsort:**

* **Descripción:** Este algoritmo utiliza una estructura de datos llamada "heap" para ordenar el arreglo de manera eficiente.
* **Funcionamiento:**
  1. Se convierte el arreglo en un heap (una estructura de árbol con la propiedad de que el padre siempre es mayor o igual que sus hijos).
  2. Se intercambia el elemento raíz (el mayor del heap) con el último elemento del arreglo.
  3. Se reduce el tamaño del heap excluyendo el último elemento (ya ordenado).
  4. Se reordena el heap restante para mantener la propiedad de orden.
  5. Se repiten los pasos 2 a 4 hasta que el heap esté vacío (todo el arreglo ordenado).
* **Memoria dinámica:** Se utiliza memoria dinámica para almacenar el arreglo y el heap.

**Código:**

Texto

Descripción generada automáticamente

**3. MergeSort:**

* **Descripción:** Este algoritmo se basa en la técnica de "divide y vencerás". Divide el arreglo en subarreglos más pequeños, los ordena recursivamente y luego los fusiona en un solo arreglo ordenado.
* **Funcionamiento:**
  1. Si el arreglo tiene un solo elemento, ya está ordenado.
  2. Divide el arreglo en dos subarreglos de igual tamaño (o casi igual).
  3. Ordena recursivamente cada subarreglo utilizando MergeSort.
  4. Fusiona los dos subarreglos ordenados en un solo arreglo ordenado.
* **Memoria dinámica:** Se utiliza memoria dinámica para almacenar el arreglo y los subarreglos durante la recursión.

**Código:**

Texto

Descripción generada automáticamente

**Big O de cada algoritmo:**

* **Shellsort:** La complejidad temporal de Shellsort es **O(n log^2 n)** en el peor de los casos, donde n es el tamaño del arreglo. La notación Big O indica que el tiempo de ejecución del algoritmo crece logarítmicamente al cuadrado del tamaño del arreglo.
* **Heapsort:** La complejidad temporal de Heapsort es **O(n log n)** en el peor de los casos. La notación Big O indica que el tiempo de ejecución del algoritmo crece logarítmicamente al tamaño del arreglo.
* **MergeSort:** La complejidad temporal de MergeSort es **O(n log n)** en el peor de los casos. La notación Big O indica que el tiempo de ejecución del algoritmo crece logarítmicamente al tamaño del arreglo.

**Notación Big O:**

La notación Big O se utiliza para describir el comportamiento **asintótico** del algoritmo, es decir, cómo se comporta el algoritmo cuando el tamaño del problema (n) se vuelve muy grande. Esta notación ignora constantes y términos de bajo orden, enfocándose en la tendencia general del algoritmo.

**Cuadro de comparación:**

|  |  |  |  |
| --- | --- | --- | --- |
| Algoritmo | Big O | Memoria | Descripción |
| Shellsort | O(n log^2 n) | Dinámica | Mejora el ordenamiento por inserción |
| Heapsort | O(n log n) | Dinámica | Utiliza un heap para ordenar el arreglo |
| MergeSort | O(n log n) | Dinámica | Divide y vencerás, fusionando subarreglos ordenados |

**Referencias**

* **Wikipedia.** (2024, 21 de junio). *Ordenamiento Shell*. <https://es.wikipedia.org/wiki/Ordenamiento_Shell>.
* **Wikipedia.** (2024, 21 de junio). *Heapsort*. <https://en.wikipedia.org/wiki/Heapsort>.
* **Wikipedia.** (2024, 21 de junio). *Ordenamiento por mezcla*. <https://es.wikipedia.org/wiki/Ordenamiento_por_mezcla>.
* **GeeksforGeeks.** (s.f.). *Analysis of algorithms | Big O analysis*. <https://www.geeksforgeeks.org/analysis-algorithms-big-o-analysis/>.