```
src > J App.java > ...
       public class App {
            private static void merge(int[] arr, int left, int mid, int right){
                   tempLeft[i] = arr[left + i];
               for(int j = 0; j < sizeright; j++){</pre>
                   tempright[j] = arr[mid + 1 + j];
               int i = 0;
               int j = 0;
               int k = left; //indice inicial para el arreglo fucionado
               while (i < sizeleft && j < sizeright) {</pre>
                   if (tempLeft[i] <= tempright[j]) {</pre>
                       arr[k] = tempLeft[i];
                   } else {
                        arr[k] = tempright[j];
                        j++;
            SALIDA CONSOLA DE DEPURACIÓN TERMINAL
27
38
43
82
                j\OneDrive\Escritorio\MergeSortJuli\MergeSortjuli>
PS C:\Users\
```

```
}
Run | Debug
public static void main(String[] args) throws Exception {
   int[] arr = {38, 27, 43, 3, 9, 82, 10};
   int n = arr.length;

   mergesort(arr, left, n - 1); //Llamada al metodo de ordenamiento MergeSort

   System.out.println(x:"Arreglo ordenado:");
   for (int num: arr){
       System.out.println(num + " ");
   }
}
```

El algoritmo de ordenamiento mergesort es altamente eficiente y se basa en el principio de "dividir y conquistar". Funciona dividiendo la lista en mitades, ordenando cada mitad de forma recursiva y luego combinando las mitades ordenadas para obtener la lista final ordenada. Este enfoque garantiza que la lista se ordene de manera eficiente, incluso en listas de gran tamaño.

Mergesort es un algoritmo estable, lo que significa que no cambia el orden relativo de elementos con claves iguales. Además, tiene un tiempo de ejecución garantizado de O(n log n), lo que lo hace muy eficiente para grandes conjuntos de datos. Esto lo convierte en una opción popular para aplicaciones que requieren ordenar grandes cantidades de datos de manera rápida y eficiente.

En resumen, mergesort es altamente eficiente y confiable, lo que lo convierte en una opción ideal para ordenar grandes conjuntos de datos en aplicaciones donde se requiere un alto rendimiento.

Abel Melquisedec Bermúdez Díaz

Número de carnet:2024-1669U