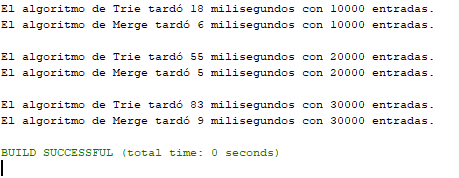
Natalia Hernández Cornejo 183420

Comparación MergeSort-TrieSort

A partir de la clase Trie y la clase Sort, comparé el desempeño de los dos diferentes tipos de sort para arreglos de palabras con diferente tamaño, desde 1,000 a 50,000 palabras. El método de ordenamiento lexicográfico en la clase Trie es un método recursivo que primero ingresa todas las palabras a un trie y luego va poniendo las palabras alfabéticamente en un ArrayList. En el caso del método mergeSort, este algoritmo separa el arreglo, para luego juntarlos de manera ordenada.

Para comparar ambos algoritmos, decidí usar el tiempo como indicador de desempeño; esto lo medí con System.currentTimeMillis. Más aún las palabras a ordenar están en un archivo de texto que se lee en el main y se va agregando a un arreglo de String.

El resultado obtenido fue que el algoritmo mergeSort es más rápido, y por lo tanto tiene mejor desempeño, que el algoritmo ordLex de la clase Trie. Llegué a esta conclusión a partir de varias pruebas con arreglos de diferente tamaño y comparando el tiempo de un algoritmo contra el tiempo de otro.



En clase de Estructuras de Datos Avanzadas, vimos muchos algoritmos de ordenamiento que podían ordenar todo tipo de objetos comparables; no obstante, el algoritmo de ordenamiento lexicográfico de tries solo se puede usar para ordenar cadenas de caracteres. Más aún, se debe tomar en cuenta que los caracteres aceptados en los tries se reducen a los caracteres que se encuentran en el arreglo de símbolos. Estas dos restricciones reducen los casos de implementación para este tipo de ordenamiento.