Universidad de los Andes 28/05/2018

David Niño 201412734

María del Rosario León 201423755

Proyecto de Diseño y análisis de algoritmos problema B

* 1. Ecuación de recurrencia:
  2. Descripción de algoritmo: Al algoritmo le entra como parámetro un String que contiene el número, este es convertido a un arreglo de enteros por medio del algoritmo toIntArray(). A continuación, se le aplica el método sp() que iterativamente va construyendo desde el sp(N, 1) hasta el sp(N, k) utilizando la ecuación de recurrencia descrita anteriormente. Para este problema también se pensó que se podía aumentar N en 1 hasta encontrar el siguiente número que contuviera los mismos dígitos que N, sin embargo, nos pareció una forma muy ineficiente de realizar esta operación.
  3. Explicación y argumentación del algoritmo implementado
     1. Método sp():
        1. Contexto: n: arreglo de enteros que contiene los dígitos del número, k: entero que representa la cantidad de números mayores que se quiere buscar.
        2. Precondición:
        3. Invariante:
        4. Post-condición:
     2. Método toIntArray():
        1. Contexto: m: String que contiene el número que se busca, dado por parámetro.
        2. Precondición:
        3. Invariante:
        4. Post-condición:
     3. Método sortedIndex():
        1. Contexto: n: arreglo de enteros
        2. Precondición:
        3. Invariante:
        4. Post-condición:
  4. Complejidad temporal: O(k nlog n) pues minimizar la última parte del arreglo tiene un tiempo de nlog n, sin embargo, la función sp se debe repetir hasta que se pueda minimizar el arreglo restante.
  5. Complejidad espacial: S(n.length) pues por cada dígito del número se necesita un espacio en el array y el algoritmo no requiere de espacio extra para intercambiarlos.