**TRABAJO DE TERCER PREVIO**

**ANDRÉS BUITRAGO**

**1151257**

**ING. OSCAR GALLARDO**

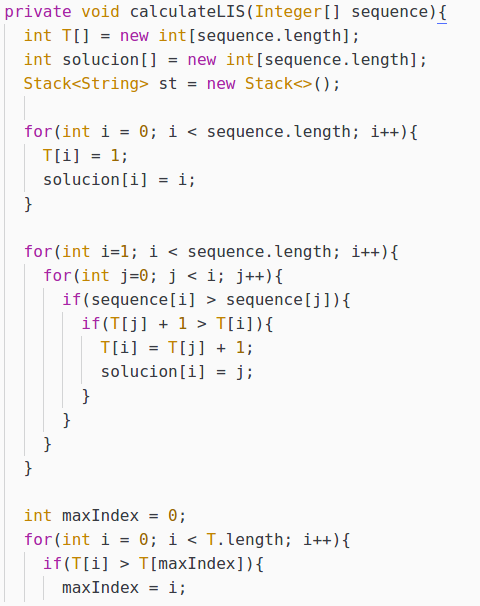
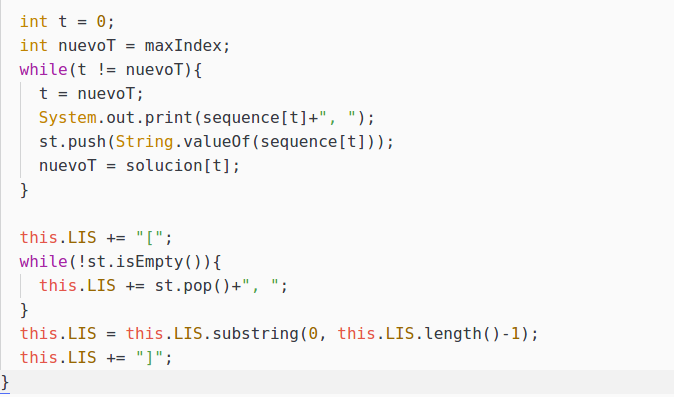
**ANÁLISIS DE ALGORITMOS**

**UNIVERSIDAD FRANCISCO DE PAULA SANTANDER**

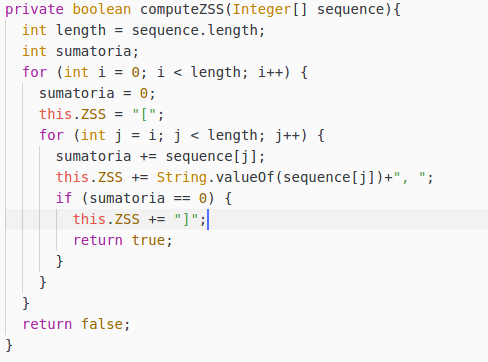
**INGENIERÍA DE SISTEMAS**

**2016**

LONGEST INCREASING SUBSEQUENCE (LIS)

* **Complejidad temporal:** Ο(n^2)
* **Algoritmo iterativo:**  
  
* **Algoritmo recursivo:**  
  *El código fuente implementado en Java está disponible en el repositorio: http://github.com/andbuitra/LIS*Por medio del **principio de invarianza** se puede concluir que ambas implementaciones tienen el mismo costo temporal.
* **Explicación:**El algoritmo toma un array con una secuencia dada y crea dos arrays temporales llamados “T” y “solucion”. La finalidad del array “T” es la de indicar la longitud del LIS dado que el máximo índice de este es la longitud del LIS mientras que la finalidad del array “solucion” es la de proveer la manera en la que debe ser recorrido el array de secuencia de fin a principio para obtener el LIS.  
    
  Por el concepto de subsecuencia se concluye que el LIS para cualquier posición del array va a ser por lo menos 1. Por ende se llena el array “T” con sólo “1” en todas sus posiciones y luego se recorre el array de secuencia poniendo un índice ***i*** en la segunda posición del array T y un índice ***j*** en la primera. Se mueve la j hasta que alcance la i y se mueve la i a lo largo de toda la secuencia. Todo esto con el fin de obtener la longitud del LIS. Luego de haber hecho todo el recorrido por el array se obtiene que el array “T” almacena la longitud del LIS (la cantidad de iteraciones que habrá que hacer sobre la secuencia para obtener el LIS) y en el arreglo “solucion” se obtiene el orden descendente en el cual este LIS debe ser recuperado.  
    
  Para finalizar el algoritmo, y devolver al usuario el resultado esperado, se almacena todo en una lista tipo LIFO (stack) y se hace pop() para obtener el orden deseado.

Zero-Sum Sub-sequence (ZSS)

* **Complejidad temporal:** Ο(n^2)
* **Algoritmo Iterativo:**  
  *El código fuente implementado en Java se encuentra disponible en el repositorio <https://github.com/andbuitra/zss-repo>*
* **Explicación**El algoritmo es muy básico y consiste en encontrar la primera subsecuencia (o para mayor facilidad de entendimiento: el primer subarray) que cumpla con la condición de que la suma de sus componentes sea igual a cero. De igual manera se cumpliría la condición si hubiese un cero en su contenido.