Proyecto Final

Sebastian García 201630047, Nicolás Sotelo 201623026

Diseño y Análisis de algoritmos

Ingeniería de Sistemas y Computación

*Universidad de Los Andes*

{ js.garcial1, n.sotelo }@uniandes.edu.co

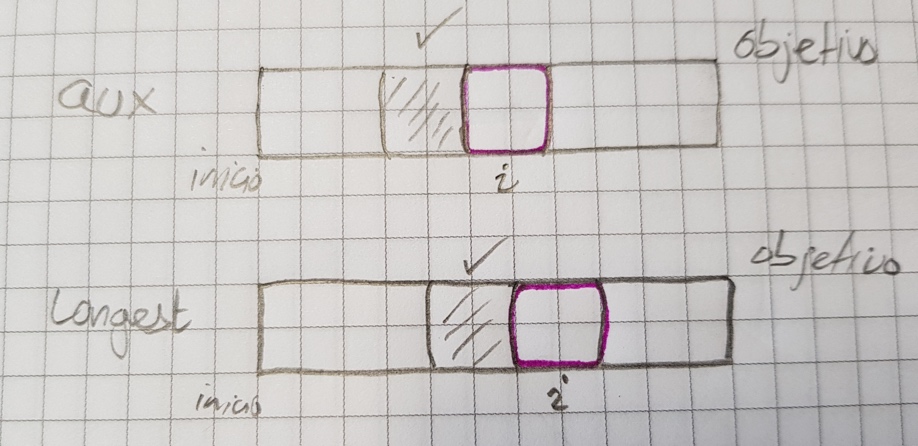
Diciembre 11 de 2018

**Problema A**

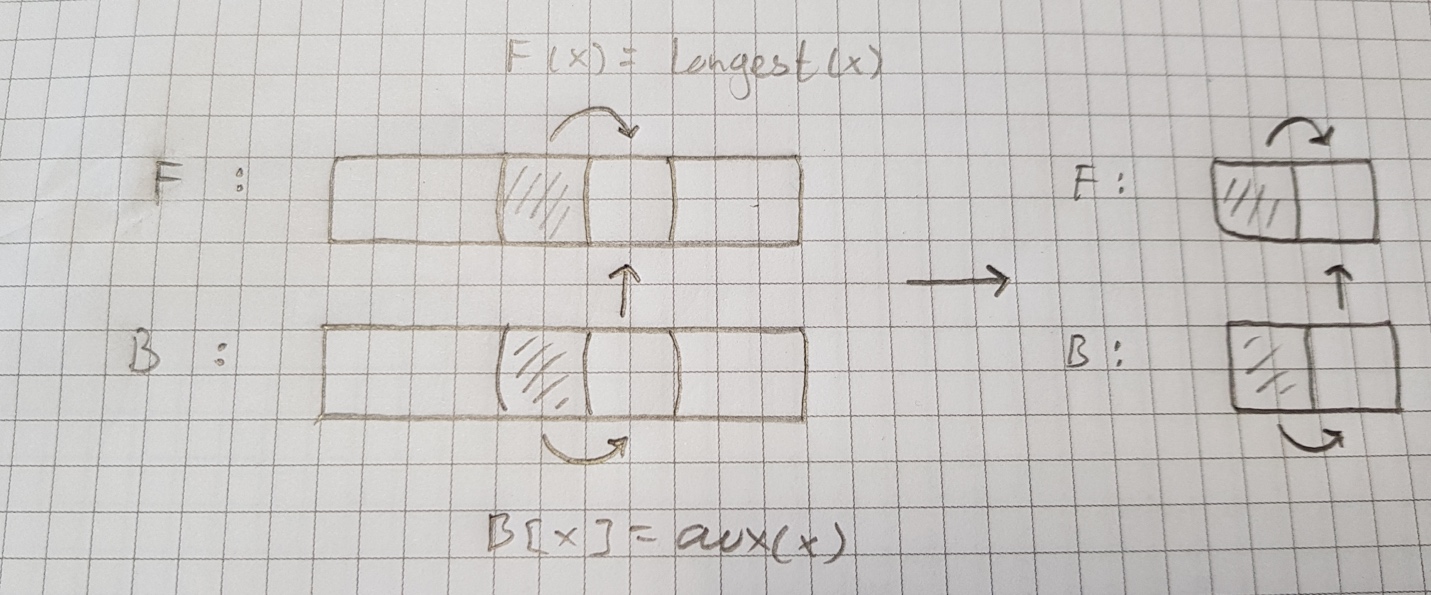
* Algoritmo de solución

**Programación dinámica.**

1. Tamaño del problema: **n**
2. Lenguaje
   1. dominio f: [0, n) contenido en nat 🡪 nat
   2. lenguaje f(k) = “tamaño del arreglo más largo de elementos ordenados casi ascendentemente. Donde ”.
   3. F(n) = ¿?
3. Recurrencia
   1. aux(i, error) =
   2. longest(i) =
4. Diagrama de necesidad



1. Estructura de Datos e Invariante



* Análisis de complejidades espacial y temporal

A partir del diagrama de necesidades se obtiene la complejidad temporal **T** con un orden de complejidad (N).

Similarmente, basándose en la estructura de datos utilizada, se obtiene la complejidad espacial **S** con un orden de complejidad (1).

* Comentarios

1. Documentación

/\*\*

\* Metodo auxiliar que calcula la longitud del arreglo casi ascendente mas largo.

\*

\* **@pre** el arreglo no esta vacio y solo contiene enteros. El error empieza con valor de 1.

\* **@post** Se hace una recursion sumando en una unidad si el el elemento del arreglo actual

\* le permite al arreglo cumplir con la condicion de orden casi ascendente. Si se encuentra

\* un unico valor que hace que no se cumpla la anterior condicion, se tendra en cuenta.

\* **@param** i posicion en el arreglo

\* **@param** array arreglo sobre el cual se va a hacer la operacion

\* **@param** error como es un orden casi ascendente, a lo sumo hay un elemento que no es ascendente

\* **@return** la longitud del arreglo casi ascendente mas largo

\*/

**public** **static** **int** aux(**int** i, **int**[] array, **int** error) {...}

/\*\*

\* Metodo principal que retorna la longitud del subarreglo casi

\* ascendente mas lago.

\*

\* **@pre** el arreglo no esta vacio y solo contiene enteros.

\* **@post** se retorna un entero, sin modificar el arreglo original.

\* **@param** i tamano del arreglo a calcular.

\* **@param** array arreglo sobre el cual se van a calcular los sibarreglos.

\* **@return** cantidad de elementos pertenecientes al subarreglo casi ascendente mas largo

\*/

**public** **static** **int** longest(**int** i, **int**[] array) {...}

1. Observaciones