Problema A

## Identificación

Juan Pablo Campos 201630726

Santiago Beltrán 201632541

## Algoritmo de solución

Entradas

|  |  |  |
| --- | --- | --- |
| Nombre | Tipo | Descripción |
| a | Array of int | Arreglo de números |
| n | int | Número de elementos de a |

Salidas

|  |  |  |
| --- | --- | --- |
| Nombre | Tipo | Descripción |
| z | int | Tamaño del segmento del subarreglo de a casi ascendente más largo. |
| s | Array of int | Subarreglo de a casi ascendentemente más largo. |

Especificación del problema:

cumple(r)= (i| 1 ≤i<n: (r[i]> r[i+1])

Pre: {n >0}

Post:{ (max(s) a | cumple(s)) }

El algoritmo cuenta con el método cumple que especifica que no pueden existir 2 elementos en un arreglo r, tal que su predecesor sea menor que él.

Y su post condición afirma que s debe ser el máximo subarreglo de a tal que cumpla con la condición cumple.

## Análisis complejidad temporal/espacial

O(n) = n

S(n) = n

Se tiene que se recorre el arreglo a original solamente una vez y se pasa por todos sus elementos.

## Comentarios del desempeño

Tener un orden de complejidad espacial y temporal lineal resulta ser un buen desempeño para el programa realizado. En el caso del desempeño está en un buen margen ya que lee todas las entradas del problema solamente una vez.