Diversión

El objetivo de este problema es familiarizarse con la especificación formal de algoritmos, así como iniciarse a la implementación eficiente de dichas especificaciones y al análisis de la complejidad de los algoritmos resultantes.

El problema

Considera la siguiente especificación, en la que aún falta por decidir la precondición:

P ≡ { … }

fun divertida(**int** a[], **int** n) **return** (**bool** resul)

Q ≡ { resul = ($(\forall i:0\leq i<n:(\Sigma j:0\leq j\leq i:a[j])\geq 0) \wedge ((\Sigma i:0\leq i<n:a[i])=0))$ }

Debes:

(1) Determinar una precondición que garantice que el resultado del algoritmo siempre está definido.
(2) Realizar una implementación **eficiente** del algoritmo especificado.

Trabajo a realizar

Para realizar el control se proporciona un archivo `divesion.cpp` que contiene un programa que lee por la entrada estándar vectores, invoca a la función `divertida` sobre dichos vectores, imprimiendo **SI** en caso de que `divertida` devuelva `true` y **NO** en caso de que devuelva `false`.

A continuación, se muestra un ejemplo de entrada procesable por este programa, y de salida producida (suponiendo una implementación adecuada de `divertida`) (los vectores se introducen indicando, como primer valor, el número de elementos, y, a continuación, enumerando los distintos elementos; el final se indica indicando -1 como tamaño del vector):

Entrada	Salida
4 1 3 5 -9 **1** 1 **1** 0 **4** 1 -3 5 3 **-1**	**SI** **NO** **SI** **NO**

Tu trabajo consiste en:

- Determinar razonadamente la precondición para el algoritmo, rellenando el hueco correspondiente, entre comentarios, antes de `divertida`
- Implementar eficientemente este algoritmo.
- Determinar razonadamente la complejidad del algoritmo, rellenando el hueco habilitado para ello.
- Entregar `diversion.cpp` a través del juez en línea de la asignatura.