Extraordinaria 22-23.pdf

user_er

Fundamentos de Algoritmia

2º Grado en Ingeniería Informática

Facultad de Informática
Universidad Complutense de Madrid

Fundamentos de Algoritmia

Grados en Ingeniería Informática

Convocatoria extraordinaria, 21 de junio de 2023. Grupos B, D y G

1. (3.5 puntos) Decimos que una secuencia no vacía de números enteros es *minimal* cuando su elemento mínimo aparece una única vez. Decimos que una secuencia no vacía de números enteros es *minimalista* cuando todas las subsecuencias de elementos consecutivos que contiene que comienzan en el primer elemento son minimales.

Por ejemplo, la secuencia `3 2 3 1` es minimalista porque la subsecuencias `3`, `3 2`, `3 2 3` y `3 2 3 1` son todas minimales. Sin embargo, la secuencia `2 3 2 1` no es minimalista porque la subsecuencia `2 3 2` no es minimal.

Desarrolla un algoritmo **iterativo eficiente** que, dada una secuencia no vacía de números enteros, determine si es o no minimalista. Debes, asimismo, justificar la corrección (precondición, postcondición, invariante, cota) y el orden de complejidad del algoritmo.

Tu algoritmo se probará mediante casos de prueba que constarán de dos líneas:

- En la primera línea aparecerá el número n de elementos de la secuencia ($0 < n \leq 10^6$).
- En la segunda línea aparecerán, en orden, los elementos de la secuencia.

La lista de casos de prueba finalizará con una línea con `-1` que no se debe procesar. Para cada caso de prueba, el programa imprimirá `SI` si la secuencia es minimalista y `NO` en caso contrario.

A continuación se muestra un ejemplo de entrada/salida:

Entrada	Salida
4 3 2 3 1 4 2 3 2 1 1 7 5 1 3 4 3 3 5 4 1 4 2 1 -1	SI NO SI SI NO

2. (3.5 puntos) Una secuencia no vacía de números enteros es *hiperminimalista* cuando el mínimo de sus elementos aparece una única vez en la secuencia y, además, las dos subsecuencias antes y después del elemento central son también *hiperminimalistas*. Recuerda que el índice del elemento central es la suma de los índices de los extremos entre dos (división entera).

Diseña e implementa un algoritmo **recursivo eficiente** que dado un vector devuelva si es *hiperminimalista*. Debes, así mismo, determinar justificadamente el coste de este algoritmo, planteando y resolviendo las recurrencias apropiadas.

Tu algoritmo se probará mediante casos de prueba que constarán de dos líneas:

- En la primera línea aparecerá el número n de elementos del vector ($0 < n \leq 10^6$).
- En la segunda línea aparecerán los elementos del vector.

La lista de casos de prueba finalizará con una línea con `-1`. Para cada caso de prueba, el programa imprimirá `SI` si el vector es hiperminimalista y `NO` en caso contrario.

A continuación se muestra un ejemplo de entrada/salida:

Entrada	Salida
7 4 3 2 1 8 6 8 5 4 3 7 6 5 7 4 1 2 3 8 7 1 -1	SI SI NO

3. (3 puntos) Desarrolla un algoritmo **vuelta atrás** que, dado un conjunto no vacío de enteros positivos C y un umbral $U \geq 0$, determine el número de secuencias no vacías *minimalistas* (véase el enunciado del ejercicio 1) que pueden formarse considerando únicamente valores en C, y para las cuáles la suma de todos sus valores no exceda a U.

Por ejemplo, dado el conjunto $C = \{1, 2\}$, y el umbral $U = 5$, hay exactamente 6 secuencias no vacías *minimalistas* formadas por elementos de C, para las cuáles la suma de todos sus valores no excede a U: `1`, `1 2`, `1 2 2`, `2`, `2 1` y `2 1 2`. Por tanto, el resultado del algoritmo para este conjunto y este umbral debe ser 6.

Tu algoritmo se probará mediante casos de prueba que constarán de tres líneas:

- En la primera línea aparecerá el número n de elementos del conjunto C ($0 < n \leq 100$).
- En la segunda línea aparecerán los elementos del conjunto C (sin duplicados).
- En la tercera línea aparecerá el umbral U ($U \geq 0$)

La entrada termina con un -1 que no se debe procesar.

Para cada caso de prueba se escribirá el número de secuencias no vacías *minimalistas* computado por el algoritmo.

A continuación se muestra un ejemplo de entrada/salida:

Entrada	**Salida**
2 1 2 5 3 4 5 6 12 3 8 11 15 7 5 50 90 180 500 200 1000 -1	6 9 0 4425