Проблем 1. Низ

Дат је низ a целих бројева дужине n. Под трансформацијом низа подразумевамо повећавање или смањивање једног елеменат низа за 1. Одредити минимални број трансформација низа потребних да се он преведе у строго растући низ b. За сваки елемент низа приказати разлику: $b_i - a_i$.

Улаз. (Улазни подаци се налазе у датотеци `niz.in`) У првом реду се налази природни број n ($1 \leq n \leq 5.000$) који означава број елемената низа. У наредном реду се налази n целих бројева који представљају елементе низа. Апсолутна вредност елемената није већа од 1.000.000.000.

Излаз. (Излазне податке уписати у датотеку `niz.out`) У првом реду треба испистати минимални број трансформација потребних за добијање строго растућег низа. У наредном реду штампати n бројева одвојених једним размаком који представљају промене бројева почетног низа. Уколико промене нису јединствене штампати било коју.

Пример 1.

niz.in	niz.out
4	4
1 1 1 1	-1 0 1 2

Пример 2.

niz.in	niz.out
5	5
1 1 1 6 4	-1 0 1 0 3

Објашњење. У примеру 1 описаним трансформацијама би добили растући низ $b = 0, 1, 2, 3$. У оба примера решење није јединствено, нпр. за други промене су могле бити и: $-1, 0, 1, -1, 2$.

Напомена. У 40% тест примера апсолутна вредност елемената низа неће бити већа од 2.000.

Решење. Проблем решавамо динамичким програмирањем. Иако се његова потреба релативно лако уочава, ефикасан опис стања није баш једноставан.

На почетку уколико уведемо смену $a_i = a_i - i$ проблем смо свели на минимални број трансформација потребних да се новодобијени низ a пребаци у неопадајући. Овим смо услов строге неједнакости заобишли на веома елегантан начин.

Како је сада низ b неопадајући можемо га представити на следећи начин:

$$b_{x_1} \equiv b_1 = b_2 = \ldots = b_{x_2-1} < b_{x_2} = b_{x_2+1} = \ldots = b_{x_3-1} < \ldots < b_{x_k} = b_{x_k+1} = \ldots = b_n$$

Дакле, треба извршити партицију низа a на поднизове узастопних елемената, а потом све елементе из истог подниза свести на једну вредност. Главна чињеница коју треба приметити је да ће вредности b_{x_i} представљати медијану елемената $a_{x_i}, a_{x_i+1}, \ldots, a_{x_{i+1}-1}$, тј. поднизове сводимо на њихове медијане.

Знамо да је сума

$$\sum_{k=1}^{n} |a_i - m| \tag{1}$$

минимална када је m медијана низа $(a_i)_1^n$. Покажимо сада да ће низ b имати горњи облик. Доказ изводимо индукцијом по k. За случај $k = 1$ из (1) директно добијамо да се низ своди на медијану. Покажимо сада да услов важи за $k + 1$ под претпоставком да важи за

k. Претпоставимо да вредност b_{k+1} није једнака медијани подниза $b_{x_{k+1}}, b_{x_{k+1}+1}, \ldots, b_n$. Означимо медијану са m_{k+1}. Уколико је $b_{x_k} < m_k$, тада смо елементе из последњег подниза могли да сведемо на вредност m_k која би довела до мањег броја трансформација, а одржали би монотоност низа. Претпоставимо сада да је $b_{x_k} > m_k$. Слично горњем резоновању, мањи број трансформација би добили уколико би последњи подниз свели на вреднот b_{x_k}. На овај начин смо добили неопадајући подниз са мањим бројем потребних трансформација при чему је број поднизова k. Дакле, мора бити да је $b_{x_k} = m_k$. Сличним разматрањем добијамо да све вредности представљају медијане.

Алгоритам ћемо објаснити у две фазе. Дефинишимо прво матрицу d на следећи начин:

$d[i][j]$ = минимални број трансформација за пребацивање низа $a_1, a_2, \ldots, a_i$ у неопадајући низ $b_1, b_2, \ldots, b_i$ при чему је $b_i \leq j$

Размотримо сада како можемо рачунати елемент $d[i][j]$ матрице d. За њега постоје два случаја: елемент b_i је једнак j или је строго мањи од j. Уколико је мањи тада имамо да је то управо вредност $d[i][j-1]$, пошто су онда и сви остали елементи мање или једнаки $j-1$. Иначе, имамо да је потребан број трансформација да $a[i]$ сведемо на j управо $|a[i]-j|$. Остали елементи имају вредности мање или једнаке од j, а за то знамо да је најоптималније решење $d[i-1][j]$. Сада можемо дефинисати рекурентну везу између елемената матрице као:

$$d[i][j] = \min\{d[i][j-1], d[i-1][j] + |a[i]-j|\}$$

Проблем код овог приступа је тај што су ограничења за вредности елемената велика, тако да сложеност горњег алгоритма, $O(n \cdot maxValue)$, не задовољава временска ограничења.

Међутим, уколико се вратимо на разматрање о облику решења, видимо да су на крају вредности елемената низа b управо и вредности елемената низа a (као медијане). Дакле, уколико елементе низа a сортирамо у низ $value$ при чему избацимо дупликате, матрицу d можемо дефинисати као

$d[i][j]$ = минимални број трансформација за пребацивање низа $a_1, a_2, \ldots, a_i$ у неопадајући низ $b_1, b_2, \ldots, b_i$ при чему је је $b_i \leq value[j]$

Рекурента веза за ову матрицу је потпуно иста као и претходна, стим што је сада њена димензија не већа од $n \times n$. Базу можемо дефинисати као

$$d[1][j] = \min_{j \in [1,j]} \{|a[1] - value[j]|\}$$

$$d[i][1] = \sum_{k=1}^{k=i} |a[k] - value[1]| = d[i-1] + |a[i] - value[1]|$$

Минимални број трансформација сада представља: $\min_{j=1}^{length(value)}\{d[n][j]\}$. Помераје добијамо једноставним памћењем претходника преко којих смо добијали вредности за елементе матрице d, тачније дефинишемо матрицу p као

$p[i][j]$ = $index$, уколико се минимални број трансформација за пребацивање низа $a_1, a_2, \ldots, a_i$ у неопадајући низ $b_1, b_2, \ldots, b_i$ добија при $b_i = value[index]$

На крају треба обратити пажњу на могућу величину решења. Због великих ограничења вредности елемената низа, за елементе матрице d треба се користити $longlong$ или $int64$.

Елементе матрице рачунамо у константном времену, чиме добијамо да је укупна сложеност алгоритма $O(n^2)$.

Рб	Опис алгоритма	Сложеност алгоритма	Број поена
01	Динамички по свим могућим вредностима	$O(n \cdot maxValue)$	40 – 50
02	Динамички по медијанама поднизова	$O(n^2 \log n^2)$	80 - 90
03	Динамички по вредностима елемената	$O(n^2)$	100

Tabela 1: Очекивани број поена у зависности од алгоритма

Рб	n	Granice za vrednosti niza
01	6	3
02	20	100
03	50	200
04	100	200
05	200	500
06	500	1000
07	500	1000
08	800	2000
09	1000	10.000
10	1000	10.000
11	2000	100.000
12	2000	100.000
13	2500	500.000
14	2500	1.000.000
15	3000	10.000.000
16	4000	10.000.000
17	4000	100.000.000
18	4500	1.000.000.000
19	5000	1.000.000.000
20	5000	1.000.000.000

Tabela 2: Вредности улазних параметара тест примера

Тестирање. Тестирање решења се вршило над корпусом од 20 тест примера. Вредност сваког примера је 5 поена. Већина података је генерисано случајно или патерном, који су потом модификовани. Међу њима било је и специјалних случајева који су обухватали

- растуће (опадајуће) поднизове
- тестерасте поднизове
- 'пирамиде' низове

Аутор:
Андреја Илић,
Природно Математички Факултет у Нишу