Problem Subarray Sort

C header	`subarray.h`
C++ header	`subarray.h`

După ce și-a petrecut vacanța de iarnă la bunicul lui, *Little Square* s-a întors acasă. Cât timp el a fost plecat, prietenul lui, *Little Triangle*, s-a jucat cu jucăriile lui, numerotate de la 1 la N. Pentru ca *Little Square* să nu se supere pe el, *Little Triangle* trebuie să pună jucăriile înapoi în ordine: $1, 2, \ldots, N$.

Inițial, tote jucăriile sunt aliniate pe raft într-o ordine oarecare.

Știind că *Little Triangle* poate sorta o secvență $[i, j]$ de jucării în $\lfloor\sqrt{j-i+1}\rfloor$ secunde, ajutați-l să găsească timpul minim în care ar putea ordona toate jucăriile.

Protocol de interacțiune

Concurentul trebuie să implementeze o funcție având semnătura:

```
int solve(int N, int P[]);
```

`solve` va fi apelat exact o dată.
Această funcție are ca parametri N, numărul jucăriilor , și P, un tabou (indexat de la 0) conținând ordinea inițială a jucăriilor. Trebuie să returneze un `int` ce reprezintă timpul minim necesar pentru sortarea de către *Little Triangle* a tuturor jucăriilor.

Grader-ul citește datele de intrare de la intrarea standard în următorul format:

- pe prima linie, numărul N
- pe a doua linie, permutarea P

Grader-ul tipărește rezultatul returnat de `solve` la ieșirea standard.

Atenție! Concurentul nu trebuie să implementeze funcția main.

Restricții

- $1 \leq N \leq 4 \cdot 10^6$
- $\lfloor x \rfloor$ reprezintă cel mai mare întreg $k \leq x$.
- Fiecare număr de la 1 la N va apărea exact o dată în P.
- Grader-ul oferit concurenților nu neapărat este asemănător cu grader-ul folosit pentru evaluare.

Subtask 1 (7 puncte)

- P este generat aleator.

Subtask 2 (8 puncte)

- $1 \leq N \leq 9$

Subtask 3 (35 puncte)

- $1 \leq N \leq 2000$

Subtask 4 (25 puncte)

- $1 \leq N \leq 100000$

Subtask 5 (25 puncte)

- Fără alte restricții.

Exemplu

input	output
5 3 1 4 2 5	2
3 1 2 3	0

Explicație

În primul exemplu, *Little Triangle* poate sorta intervalul $[0, 1]$ in $\lfloor\sqrt{1-0+1}\rfloor = \lfloor\sqrt{2}\rfloor = \lfloor 1.41421\ldots\rfloor = 1$ secunde. Permutarea devine 1 3 4 2 5. El poate acum sorta intervalul $[1, 3]$ in $\lfloor\sqrt{3-1+1}\rfloor = \lfloor\sqrt{3}\rfloor = \lfloor 1.73205\ldots\rfloor = 1$ secunde. Permutarea devine 1 2 3 4 5. În total *Little Triangle* poate sorta toate jucăriile în $1 + 1 = 2$ secunde, ceea ce este valoarea timpului minim.

În al doilea exemplu jucăriile sunt deja sortate.