**Căutare Bitonică**

O secvenţă de numere este bitonică dacă începe cu un şir ordonat strict crescător şi continuă cu un şir ordonat strict descrescător. De exemplu şirul 0 1 2 3 4 5 6 7 8 9 6 4 3 este un şir bitonic, dar şirul 0 1 2 3 4 5 6 7 4 3 5 6 nu este. Un şir care este ordonat strict crescător este şi el bitonic, ultima parte, şirul ordonat descrescător, se consideră vid. Analog pentru un şir ordonat strict descrescător.

**Problemă**

Fiind dat un șir bitonic se cere să se gasească toate pozițiile unde apare un număr x dat. Se garantează ca numărul x apare de cel mult două ori. Dacă numărul x apare de două ori el va apărea în partea cu șirul crescător și în partea cu șirul descrescător.

**Input**

Se va citi de la standard input

1. Pe prima linie: N -- numărul de elemente din șirul bitonic.
2. Pe a doua linie vor fi N numere reprezentând elementele din șir.
3. Pe a treia linie va fi scris numărul T care reprezinta numărul de intrebari de tipul: pe ce poziție/poziții se găsesc numărul x.
4. Pe fiecare din următoarele N lini se găseşte căte un număr x.

Șirul se considera indexat de la 0.

**Output**

Se va afișa pe standard output.

Se vor afișa T linii. Fiecare linie va conține toate pozițiile unde apare numărul x dat ca și input sau textul **Number Not Found** daca numărul nu exista. Dacă se afişează două numere, acestea vor fi separate de un singur spaţiu. Dupa ultimul număr de pe linie, se lasă de asemenea un spaţiu.

Ultima linie trebuie terminată cu caracterul \n.

**Exemple și Constrângeri:**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 13  0 1 2 3 4 5 6 7 8 9 6 4 3  2  0  3 | 0  3 12 |
| 13  0 1 2 3 4 5 6 7 8 9 6 4 3  1  11 | Number Not Found |

**3 <= N <= 500000, 1 <= T <= 10000**

**Timp maxim de execuție: 0.3 secunde**