C. Pearls in a Row

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n pearls in a row. Let's enumerate them with integers from 1 to n from the left to the right. The pearl number i has the type a_i.

Let's call a sequence of consecutive pearls a `segment`. Let's call a segment `good` if it contains two pearls of the same type.

Split the row of the pearls to the maximal number of good segments. Note that each pearl should appear in exactly one segment of the partition.

As input/output can reach huge size it is recommended to use fast input/output methods: for example, prefer to use `scanf/printf` instead of `cin/cout` in C++, prefer to use `BufferedReader/PrintWriter` instead of `Scanner/System.out` in `Java`.

Input

The first line contains integer n ($1 \le n \le 3 \cdot 10^5$) — the number of pearls in a row.

The second line contains n integers a_i ($1 \le a_i \le 10^9$) – the type of the i-th pearl.

Output

On the first line print integer k — the maximal number of segments in a partition of the row.

Each of the next k lines should contain two integers l_j, r_j ($1 \le l_j \le r_j \le n$) — the number of the leftmost and the rightmost pearls in the j-th segment.

Note you should print the correct partition of the row of the pearls, so each pearl should be in exactly one segment and all segments should contain two pearls of the same type.

If there are several optimal solutions print any of them. You can print the segments in any order.

If there are no correct partitions of the row print the number "`-1`".

Examples

input	Copy
5 1 2 3 4 1	
output	**Copy**
1 1 5	

input	Copy
5 1 2 3 4 5	
output	**Copy**

```
-1
```

input Copy

```
7
1 2 1 3 1 2 1
```

output Copy

```
2
1 3
4 7
```