Problem Wall

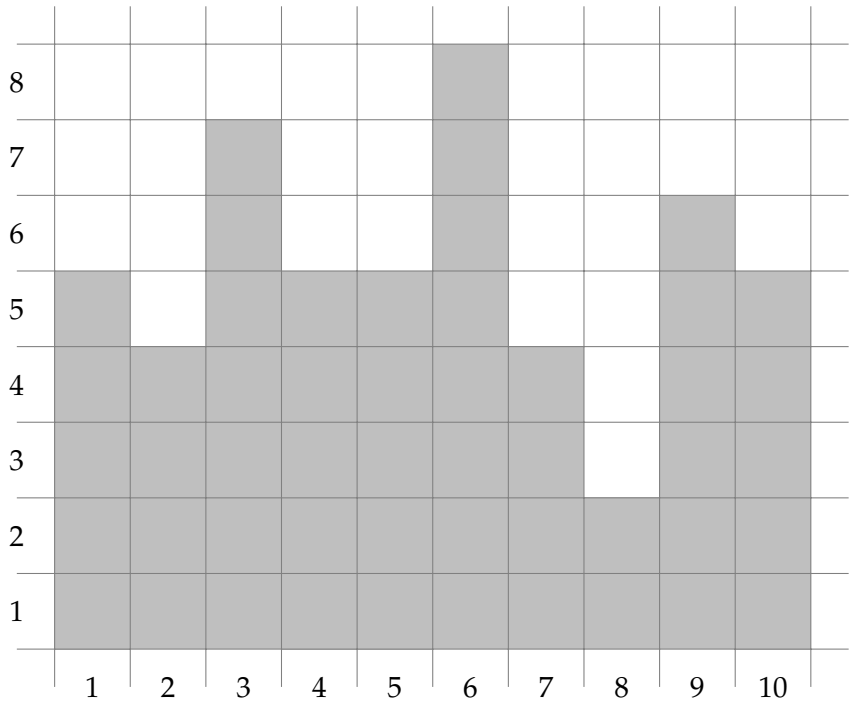
Input data	`stdin`
Output data	`stdout`

The Fortress of Suceava, built by Petru Mușat during the glory days of medieval Moldova at the end of the 14th century and consolidated in the 15th century by Ștefan cel Mare, is best known for never being conquered by the Ottoman Empire.

The Fortress of Suceava

The fortress's medieval system of fortifications was comprised of various constructions (royal courts, high-walled monasteries, and significant strategic points) designed for defensive purposes that were surrounded by high stone walls.

We represent a fragment of the fortress's wall similarly to the figure displayed below. It is easy to identify the stone blocks that make up the wall. The wall is comprised of adjacent towers built through layering identical cubic stone blocks. Thus, for the given example the wall contains 10 towers, out of which the first one contains 5 blocks, the second one contains 4 blocks, the third one contains 7 blocks, and so on. Note that the wall does not have a constant height across its width as some of the original blocks had been destroyed long ago.



Example of a Wall

The Romanian restorers were able to recover S stone blocks and they wish to restore a fragment of the wall that is as large as possible. In other words, they would like to repair a contiguous sequence of towers by adding blocks in such a way that all of the towers in the sequence would have the same height. Because of historical reasons, the height of the restored fragment must not exceed the highest tower from the fragment (before restoration).

Given the configuration of the wall before restoration, comprised of N towers, indexed from left to right using the natural numbers between 1 and N, and for each tower given the number of stone blocks it contains, find the maximum width of the wall fragment that can be restored, such that the restorers would have to use *all* of the S recovered stone blocks for the fragment. The width of the fragment is defined as the number of towers contained in it.

Input Data

The input consists of two lines. The first line contains two space separated positive integers N and S (previously defined in the task's statement). The second line contains N space separated positive integers, the i-th of which denotes the number of stone blocks contained in the i-th tower of the wall.

Output Data

Output a single line containing two space separated integers L_{max} and Pos with the following meaning:

- L_{max} - the maximum width of the restored fragment
- Pos - the index of the leftmost tower in the optimal solution

We guarantee that at least one fragment can be restored by using *all* of the S recovered stone blocks. If there are multiple fragments with the same maximum width, output the starting position of the fragment with the greatest height. If there are still several such fragments, output the starting position of the leftmost one.

Restrictions

- $1 \leq N, S \leq 200\,000$
- $1 \leq$ *the number of blocks in any tower* $\leq 10\,000$
- This problem has individual test scoring. See the notice for more details.

#	Points	Restrictions
1	20	$1 \leq N \leq 500$ and $1 \leq S \leq 1\,000$
2	24	$1 \leq N, S \leq 10\,000$
3	40	$1 \leq N, S \leq 100\,000$
4	16	No further constraints.

Examples

Input data	Output data
11 7 5 4 7 6 4 7 6 8 7 5 1	5 6

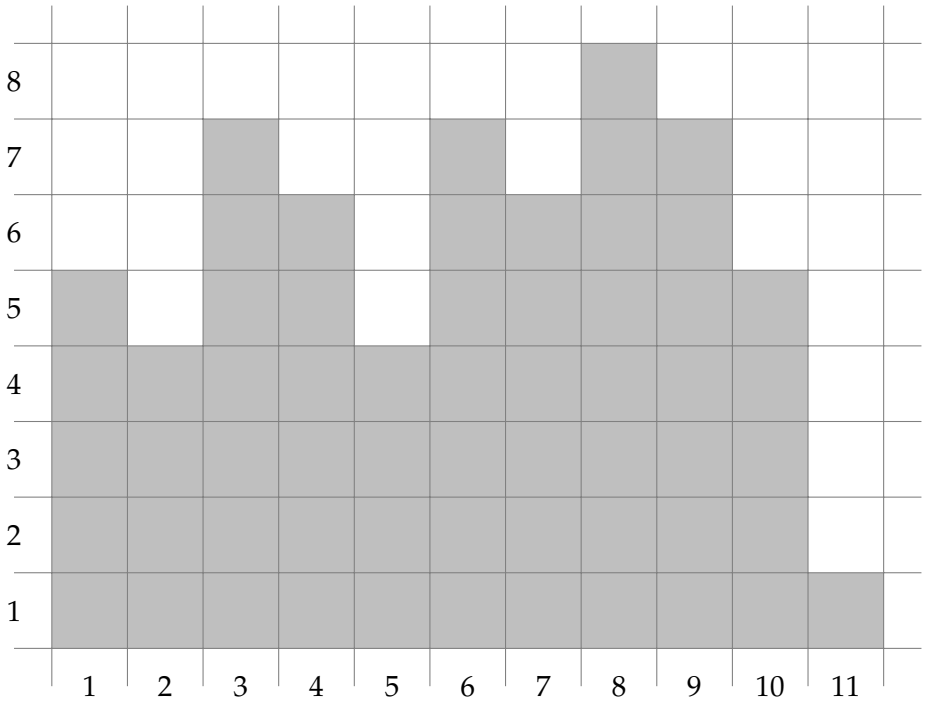
Explanations

We notice that there are two maximum-width (equal to 5) fragments that can be restored using exactly $S = 7$ stone blocks.

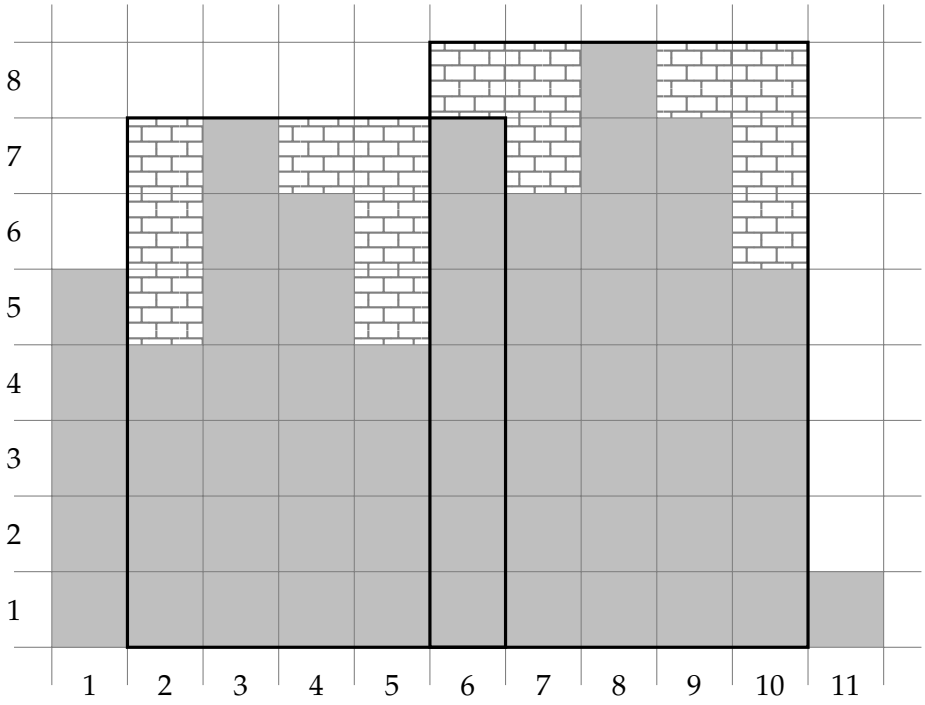
The first fragment is comprised of the towers from index 2 to 6. Its height after restoration would be equal to 7.

The second fragment is comprised of the towers from index 6 to 10. Its height after restoration would be equal to 8. Since after restoration this fragment would be taller than the previous one, we have to

output the index of its leftmost tower, i.e. 6.



The Unrestored Wall Fragment



The Restored Wall Fragment