Problem 4 – Towns

You are given **N** towns and you should find a path in them that satisfies few conditions. In each of the towns there are fixed number of **citizens** (between 1 and 1,000,000,000, inclusive).

The towns can be visited **only in the order they are given** on the console. You are not obligated to start the traversing from the first town. You can also **skip some of the towns** but you are not allowed to visit a town more than once.

For example if we have towns Sofia, Plovdiv, Varna and Burgas **you can visit:**

- Sofia, Varna, Burgas (by skipping Plovdiv)
- Plovdiv, Burgas (by starting from Plovdiv and skipping Varna)
- Sofia, Plovdiv, Varna, Burgas (by visiting every town)
- etc.

You are **not allowed** to visit the towns in wrong order. For example:

- Varna, Sofia is **not** allowed because Varna should be visited after Sofia in the given order
- Sofia, Varna, Sofia is **not** allowed because you are visiting Sofia for a second time
- etc.

You should satisfy **one more condition** when traversing towns. If you divide the path into 2 sub-paths that share one common town (part of the original path) such that the last town of the first path is the first town of the second path, then all number of citizens in the first path should be in ascending order and all number of citizens (increasing number of citizens) in the second path should be in descending order (decreasing number of citizens). Yeah.

For example look at the first example below. The path Pleven, Burgas, Varna, Sofia, Ruse, StaraZagora satisfies this condition (in fact, it satisfies all conditions) because we can divide the path into 2 sub-paths (first: Pleven, Burgas, Varna, Sofia; second: Sofia, Ruse, StaraZagora) and the first path has increasing number of citizens and the second path has decreasing number of citizens.

Write a program that finds **the longest path that satisfies the given conditions**.

Input

The input data should be read from the console.

The number of towns **N** will be given on the first input line.

On each of the next **N** lines there will be information for one of the towns in the order they can be traversed. If you split the line by a single space (‘ ‘) the first part of the line will contain the number of citizens in the town, and the second part of the line will contain the name of the town. See the examples below.

The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

The output data should be printed on the console.

On the only output line write the longest path that satisfies the given conditions.

Constraints

- **N** will be an integer between 1 and 1000, inclusive.

- The name of each town will contains only Latin letters (both lowercase (‘a’-‘z’) and uppercase (‘A’-‘Z’) are allowed)
- The length of each town name will be between 1 and 20 characters.
- Each town name will be unique.
- Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

Examples

Example input	Example output	Explanation
8 108214 Pleven 339077 Plovdiv 200612 Burgas 334688 Varna 1241396 Sofia 92162 Sliven 151951 Ruse 137907 StaraZagora	6	The longest path that satisfies all conditions contains these 6 cities: Pleven (first town) Burgas (bigger than Pleven) Varna (bigger than Burgas) Sofia (bigger than Varna) Ruse (smaller than Sofia) StaraZagora (smaller than Ruse) (The path can be cut in 2 paths satisfying citizens count condition. Sofia is the town that divides the path into 2 other paths)
4 19906 NikiTown 19832 EvlogiTown 19894 IvoTown 19896 DonchoTown	3	You can start from EvlogiTown, then go in IvoTown and then DonchoTown. If we divide the path into 2 sub-paths • EvlogiTown, IvoTown and DonchoTown • DonchoTown They satisfy the ascending-descending condition.
3 2 HaxorTown 2 LeetTown 2 RoxxorTown	1	You can visit any of the 3 towns, but cannot move any further because of the citizens count traversing requirements