# 02. Beaver at Work

*The Beaver got married and he needs to build a new home for his family. Help him to collect as many wood branches as possible for he****'****s new home.*

You will receive an integer **n** for the **size** of the **pond** with a **square** shape. On the next **n lines,** you will receive the **matrix**, which represents the **pond.**

The Beaver will be placed in a **random** **cell**, marked with the letter '**B**'. Each cell stands for a place where the beaver can move. If the cell is marked with а **lowercase character**, that means there is a **wood** **branch**. If thecell is **marked** with **'F'**, the beaver catch and eats a **fish**. **All of the empty positions** will be marked with a **'-' (dash).**

The Beaver can move "**up**", "**down**", "**left"**, and "**right**". These will be the **commands** that you**'**ll receive. Anytime the beaver moves, change the value of the cell of his new position to '**B**' and the cell it left to **'-' (dash)**.

* If the **beaver** **moves to a wood** **branch** it puts it away for his new home.
* If the beaver moves outside of the pond (field), he drops **his last collected wood branch** (if there are any), **without changing its current position**.
* If the beaver moves to **'F'**, he eats a **fish** and gains the **strength to swim underwater for a very long time.**
  + **If the fish is NOT located in the last index, the beaver swims to the last index in the direction it received.**

**Еxample**: If the beaver is located on **[0;0]and** moves **right**, eats a **fish in [0;1]**, he swims all the way **right** to the **column's last position**, **[0;lastIndex]** **(row remains the same)**.

* + **If the braver is eats a fish on the last index – it swims in the opposite direction**. Opposite of "**up**" is "**down**", opposite of "**left"** is"**right"** and vice versa. Set the value of the fish**'**s cell to **'-' (dash)**. If there is a **wood branch at the cell the beaver swims to,** it **collects** the wood branch. While swimming underwater, the beaver **does not** collect any wood branches.

**Еxample**: If the beaver moves **up**, and eats a **fish in [0;0]**, he **can't swim all the way up**, so **he swims all the way down** to the **row's last position**, **[lastIndex;0]** **(column remains the same)**.

When the command"**end**" is received or the beaver **manages** to **collect** **all the wood** **branches**, **stop** the program, and **print** the result in the format below.

### Input

* On the first line – integer **n** – the size of the pond (field).
* The **next n lines** hold the values for every **row.**
* On each of the next lines, you will receive a command.

### Output

* If the Beaver manages to collect **all wood branches** in the pond, print:

**"The Beaver successfully collect {numberOfbranches} wood branches: {branch1}, {branch2}, {branch3}(…)."**

* If the Beaver could not collect every branch in the pond, print:

**"The Beaver failed to collect every wood branch. There are {numberOfbranches} branches left."**

* Then print the last state of the pond.

### Constraints

* The size of the square matrix will be between [2…15].
* The Beaver's position will be marked with **'B'**.
* The fish will be marked with **'F'**.
* **Wood branches** will be smaller case letters from the English **alphabet (a - z)**.
* There will be no case where the **Beaver** will **move** to **a fish two consecutive times**.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 4  - F e -  - B F y  - - - q  - - z x  up  right  right  right  up  end | The Beaver failed to collect every wood branch. There are 2 branches left.  - - e -  - - F y  - - - B  - - - -  5  F - - z -  - - F B F  - - - - -  - - - - -  - - - - -  up  down  right  end | The first command is **up** so **B (beaver)** moves up.  **On turn 1**: the beaver moves **to fish**. It should **swim up**, but the fish is placed at the **end of the pond**, so the braver swims in the **opposite direction, which is down**. **[3;1]**  **On turn 2:** thecommand is **right**, the beaver moves right and collects a **wood branch**. Now he has one wood branch ('z').  **On turn 3:** the command is **right**, the beaver moves right and collects another **wood branch**. Now he has **two wood** branches ('z', 'x').  **On turn 4**: command is **right**, the beaver moves right, which is an **invalid index**, so he **loses he's last wood branch** ('x'). Now he has only one wood branch ('z').  **On turn 5**: the command is **up**, the beaver moves up and collects another branch. Now he has two branches ('z', 'q').  We receive command **end** so we stop the program. There are **two branches left in the pond**. We print the output and the last state of the pond.  **Turn 1 Turn 2 Turn 3 Turn 4 Turn 5**  - - e - - - e - - - e - - - e - - - e -  - - F y - - F y - - F y - - F y - - F y  - - - q - - - q - - - q - - - q - - - B  - B z x - - B x - - - B - - - B - - - - |
| 3  - - -  B F -  d b m  down  left  right  right  right | The Beaver successfully collect 2 wood branches: b, m.  - - -  - F -  - - B |  |

3

- - -

B - -

d b m

right

left

left

end