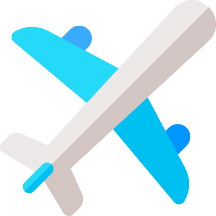
# War Planes



Write a program that receives a state of a **battlefield**, reads some **attack commands** and changes the state of the field. On the **first line** you will be given the **size** of the field (**N**). On the next **N** lines, you will be given the **state** of the field represented as some **symbols** separated by **single space**.

Here are the **possible symbols** and their meaning:

* **"."** - **empty** field
* **"p"** - the starting **position** of the **plane**
* **"t"** - a **target** that the plane wants to destroy

After the field state you will be given another **number** (**M**). On the next **M** lines, you will be given some **commands**:

* **move {right/left/up/down} {steps}** - the plane **moves** in the given **direction** with the given **steps**. **Move** the player **only** if the filed he wants to step on is marked with **"."**
* **shoot {right/left/up/down} {steps}** - the plane **shoots** in the given direction with the given **steps** (from his **current position without moving**), the field gets **destroyed** and marked with a **"x"**. If the plane **shoots** at a target, it also gets **destroyed**. When a **field** is **destroyed**, the plane can **no longer** **step** on it.
* **Validate** the positions, since they can be **outside** the field

Keep track of all of the **killed targets**. If at any point there are **no targets left**, **end** the program and print **"Mission accomplished! All {count\_targets} targets destroyed."**. If after **all the commands,** there are still **targets left** print **"Mission failed! {count\_left\_targets} targets left."**.

Finally, print the **state of the field** as shown in the examples

### Input

* **N** - count of the **rows and columns** on the field (NxN matrix)
* On the next **N lines** - the **field** itself (symbols **separated by a single space**)
* **M** - count of **commands**
* On the next **M lines** - the commands in the format described above

### Output

* On the **first line** print one of the following:
  + If all the **targets** were **destroyed**

**Mission accomplished! All {count\_targets} targets destroyed.**

* + Otherwise, if the **commands have ended** and there are still **targets left**

**Mission failed! {count\_left\_targets} targets left.**

* Finally, **print the field** as shown in the examples

### Constrains

* There will only be **one position** of the plane on the field
* All the **commands** will be **valid**
* There will **always be at least one target**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 4  . . p .  . . . .  . . t .  t . . .  4  shoot down 2  move down 2  move left 2  shoot down 3 | Mission accomplished! All 2 targets destroyed.  p . . .  . . . .  . . x .  x . . . | The plane **shoots** below his position with **step 2** and **kills** the first target  He wants to **move** down, but he **cannot**, because the **field is destroyed**  He moves **2 positions** to the **left**  He **shoots** **below** him again but with **step 3** and **kills** the final target |
| 5  . . . t .  t . . . .  . . . . .  . p t . .  . . . . .  6  move right 1  move right 2  shoot up 3  shoot left 1  move up 2  shoot left 2 | Mission failed! 1 targets left.  . . . x .  t x . p .  . . . . .  . . x . .  . . . . . |  |
| 2  . t  p .  6  move right 2  move right 1  shoot down 1  shoot up 1  move up 3  move left 1 | Mission accomplished! All 1 targets destroyed.  . x  . p |  |

*My grandfather never threw anything away, bless him.  
He died in the war holding on to a hand grenade…*