题目描述

军事演习中分为红蓝两方，我方为蓝方。

地图简化为 n 行 m 列的二维矩阵形式（下标从 0 开始），红方基地在地图中使用 # 表示，蓝方基地在地图中使用 * 表示，中立区域使用 . 表示。

下图为地图示例：

```
.........###............
........###.....**.......
.........###....****.....
........#####.....*****..
.........##........****..
........................
............***.........
......###....***.........
......##................
.....................**..
...##..............***...
........................
```

两方均只使用战斗机进行演练，各个基地停放若干架战斗机并存储若干战备（燃油、导弹）。

战斗机在地图中可向上、下、左、右四个方向进行移动，假定各战斗机在行驶途中有各自飞行高度而不会相撞。所有战斗机的速度均为 1 格 / 帧，空中每经过 1 帧消耗 1 单位油量，攻击距离均为 1 格。

各战斗机具有的属性为：

- 编号
- 初始位置（保证在己方基地）
- 油箱容量
- 最大载弹量

战斗机可以在起飞的基地进行加油和装弹，也可以在途经的己方基地加油和装弹，装填时间不计。初始的油量与载弹量均为 0。

出于隐蔽考虑，战斗机不应经过未被摧毁的敌方基地，而每架战斗机都可以向其上下左右四个方向发射导弹，射程为 1 格，同时不能飞出地图边界。战斗机中途可以在非敌方基地处降落（不发出移动指令则视为降落）。

各个基地具有属性如下：

- 位置
- 燃油储备量
- 导弹储备量
- 防御数值
- 军事价值

其中，防御数值指的是，彻底摧毁该基地所需要的导弹数（可以累加）。当彻底摧毁某个基地后，才可得到对应军事价值的分数。

这是一个简化的问题，所以你只需要考虑作为蓝方如何调度战斗机进攻红方基地，不需要考虑红方对此做出的反击或防御措施，目标为得分最大化。

输入格式

地图
第一行两个正整数 $n, m(1 \leq n, m \leq 200)$ 说明地图表示为一个 n 行 m 列的矩阵。

接下来 n 行，每行 m 个字符，表示地图情况。

如：

```
6 9
......*..
..#......
.##......
.......*.
......***
...#.....
```

基地
先输入蓝方基地信息，再输入红方基地信息。例如上图中有 5 个蓝方基地，4 个红方基地。

对于双方，均先输出一个正整数 N 表示基地数量。每个基地使用 2 行进行描述。

对于一个基地，先输入一行 x, y 表示接下来要描述位于第 x 行第 y 列的基地信息。**注意**：下标从 0 开始。

接下来一行 4 个整数分别表示燃油储备量 $g(0 \leq g \leq 10^3)$、导弹储备量 $c(0 \leq c \leq 2 \cdot 10^3)$、防御数值 $d(1 \leq d \leq 10)$、军事价值 $v(1 \leq v \leq 200)$。

战斗机
接下来一行一个正整数 $k(1 \leq k \leq 10)$，表示我方战斗机数量。

下面 k 行，每行表示一架战斗机的属性。第 i 行 4 个整数 $x, y, g, c(0 \leq x < n, 0 \leq y < m, 1 \leq g \leq 500, 1 \leq c \leq 1000)$，表示编号为 $i - 1(0 \leq i - 1 < k)$ 的战斗机初始位于第 x 行第 y 列（保证为蓝方基地），油箱容量为 g，最大载弹量为 c。

注意：编号从 0 开始。

输出格式

对于每一帧，输出若干行战斗机指令。

指令格式	作用
`move <id> <dir>`	该指令表示战斗机的移动。第一个参数为移动的战斗机编号，第二个参数为移动方向的编号。 `0 1 2 3` 分别表示“上、下、左、右”。若一帧内对同一战斗机输出多条 `move` 指令，只执行第一条合法的 `move` 指令，忽略其余指令。
`attack <id> <dir> <count>`	该指令表示战斗机的进攻。第一个参数为进攻的战斗机编号，第二个参数为攻击方向的编号， `0 1 2 3` 分别表示“上、下、左、右”，第三个参数为投放导弹数量。
`fuel <id> <count>`	该指令表示为战斗机添加燃油。第一个参数为添加燃油的战斗机编号，第二个参数为添加燃油的数量。
`missile <id> <count>`	该指令表示为战斗机装填导弹。第一个参数为装填导弹的战斗机编号，第二个参数为装填导弹的数量。

除 `move` 指令每架战斗机只能执行一次外，其余合法指令每架战斗机均可执行多次，非法指令将被忽略并给出警告信息。按照输出顺序执行各合法指令，可以移动后再进行攻击、加油、装弹等操作。

指令输出结束后，输出一行 `OK` 表示该帧指令输出结束，并刷新输出。

每一帧判题器将会打印当前帧数、得分以及该帧的错误指令警告信息。

例如：

```
[INFO] Frame: 999 Score: 4859
[WARNING] move 2 4: Invalid parameter(s)
[WARNING] fuel 2 10000: No enough supplies
[WARNING] move 1 0: Fighter will be out of bound
[WARNING] missile 4 99999: No enough supplies
[WARNING] attack 1 -1 999: Invalid parameter(s)
...
```

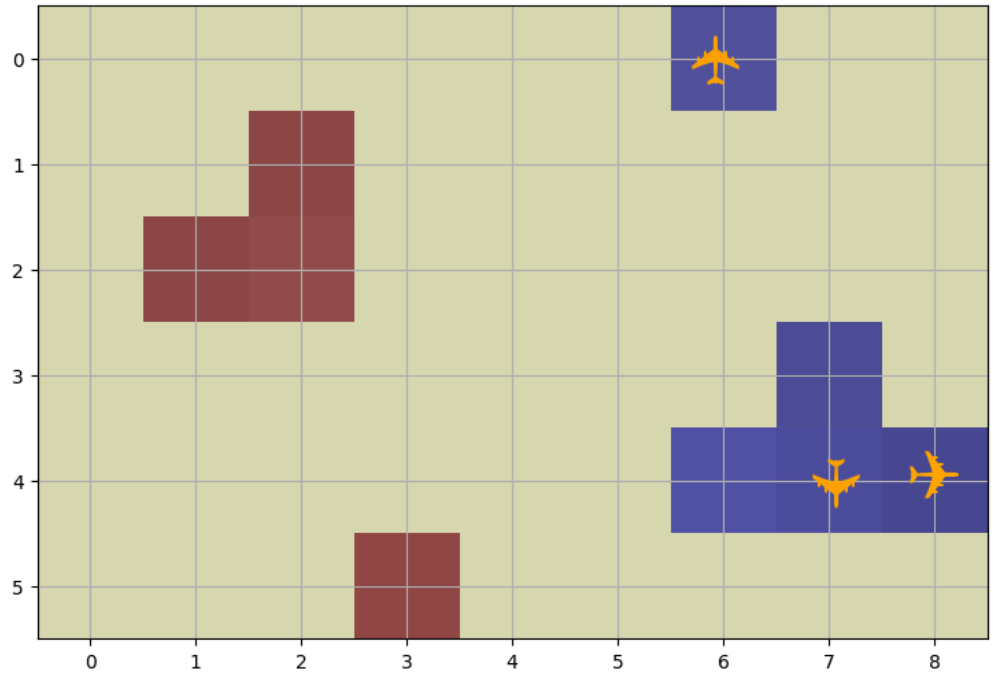
总共 15000 帧，每一帧的处理时间不受限，但需要保证程序在 2 分钟内结束，超时后做出的指令无效。

样例输入

```
6 9
......*..
..#......
.##......
.......*.
......***
...#.....
5
0 6
100 20 5 50
3 7
100 15 4 40
4 6
150 25 3 60
4 7
120 18 6 45
4 8
90 10 2 30
4
1 2
80 10 1 20
2 1
100 20 2 25
2 2
110 15 4 35
5 3
95 12 3 30
3
0 6 1000 500
4 7 800 300
4 8 900 400
```

样例解释

- 地图大小为 6 行 9 列。
- 接下来 6 行描述了地图的布局，`.` 表示中立区域，`*` 表示蓝方基地，`#` 表示红方基地。
- 蓝方有 5 个基地，每个基地的信息包括位置、燃油储备量、导弹储备量、防御数值、军事价值。
- 红方有 4 个基地，每个基地的信息也以同样的格式给出。
- 有 3 架蓝方战斗机，每架战斗机的信息包括初始位置、油箱容量、最大载弹量。



> 图中战斗机朝向仅为示例，实际飞行方向由指令中的参数决定。

错误信息

`move` 的错误警告格式：

错误类型	警告消息
参数不合法（如 `id` 小于 0 或 `dir` 大于 3 或参数数量不为 2）	`[WARNING] move <id> <dir>: Invalid parameter(s)`
缺乏燃油	`[WARNING] move <id> <dir>: No fuel to fly`
尝试越界	`[WARNING] move <id> <dir>: Fighter will be out of bound`
该帧已经移动	`[WARNING] move <id> <dir>: Fighter has already moved`
尝试进入未摧毁的红方基地	`[WARNING] move <id> <dir>: Can't pass a non-destroyed red base`

`attack` 的错误警告格式：

错误类型	警告消息
参数不合法	`[WARNING] attack <id> <dir> <count>: Invalid parameter(s)`
没有足够的导弹	`[WARNING] attack <id> <dir> <count>: No enough missile(s)`
尝试打击中立位置、己方基地或边界外位置	`[WARNING] attack <id> <dir> <count>: Invalid target`

`fuel` 的错误警告格式：

错误类型	警告消息
参数不合法	`[WARNING] fuel <id> <count>: Invalid parameter(s)`
当前位置不为己方基地	`[WARNING] fuel <id> <count>: Not at a blue base`
当前位置没有足够的燃油	`[WARNING] fuel <id> <count>: No enough supplies`
剩余容量不足	`[WARNING] fuel <id> <count>: No enough capacity`

`missile` 的错误警告格式：

错误类型	警告消息
参数不合法	`[WARNING] missile <id> <count>: Invalid parameter(s)`
当前位置不为己方基地	`[WARNING] missile <id> <count>: Not at a blue base`
当前位置没有足够的导弹	`[WARNING] missile <id> <count>: No enough supplies`
剩余容量不足	`[WARNING] missile <id> <count>: No enough capacity`

所有的非法指令但被忽略，但仍建议程序对指令的合法性进行检查。

判题器使用方式

判题器接受两个参数： `<input_file>` 和 `<output_file>` 。

例如，当输入文件为 `testcase.in` ，输出文件为 `testcase.out` 时，Windows 环境下可在命令行中输入：

```
checker.exe testcase.in testcase.out
```

判题信息将打印在命令行中，最后一行输出 `Total score: xxxxx` 。

可将判题信息重定向至文件，以加速判题过程。

`demo` 文件夹中的 `demo.exe` （Windows）和 `demo` （Linux）使用标准输入输出读取输入和输出答案，最后使用标准错误输出打印期望分数以避免和答案混淆。对其进行判题的示例如下：

```
# Windows
# In the demo directory
demo.exe < ..\data\testcase1.in > ..\data\testcase1.out
Expected score: 110
# Now in the checker directory
checker.exe ..\data\testcase1.in ..\data\testcase1.out > report.log

# Linux
# In the demo directory
./demo < ../data/testcase2.in > ../data/testcase2.out
Expected score: 97676
# Now in the checker directory
./checker ../data/testcase2.in ../data/testcase2.out > report.log
```

判题信息重定向到判题器所在文件夹的文件 `report.log` 中：

```
14995    [INFO] Frame: 14995 Score: 110
14996    [INFO] Frame: 14996 Score: 110
14997    [INFO] Frame: 14997 Score: 110
14998    [INFO] Frame: 14998 Score: 110
14999    [INFO] Frame: 14999 Score: 110
15000    [INFO] Frame: 15000 Score: 110
15001    Total score: 110
```