Иванчо и приятелите му обичат да играят на странни игри. Една от тях се нарича въртелянка.

В тази игра, **N2** приятели на Иванчо се нареждат в квадрат със страна **N** (така че всеки приятел на Иванчо заема едно квадратче с измерения **1x1**) и Иванчо ги инструктира да се въртят по всевъзможни начини, например по редове, колони и диагонали. Вашата задача е да напишете програма, която да отговаря на запитвания от вида „Кой от приятелите на Иванчо стои на позиция **(i, j)** след като са направени всички завъртания?“. Тъй като Иванчо има прекалено много приятели, че да им запомни имената, той ги нарича на галено с естествените числа от **1** до **N2**. Например когато в играта участват **22** приятели на Иванчo тяхната начална наредба е следната:

|  |  |
| --- | --- |
| 1 | 2 |
| 3 | 4 |

Иванчо задава своите инструкции по следния начин:

<*вид\_въртене*> <*позиция*> <*брой завъртания*>

Вид въртене се задава по следния начин:

1. ‚r’ обозначава завъртане по ред
2. ‘c’ обозначава завъртане по колона
3. ‘d’ обозначава завъртане по прав диагонал1
4. ‘x’ обозначава завъртане по опак диагонал2

Позицията се задава с:

1. При ‘r’ или ‘c’ – с единствено число **T**, обозначаващо редa/колоната, която трябва да бъде завъртяна (0 ≤ Т ≤ n-1)

2. При ‘d’ или ‘x’ – с две числа **R** и **C** задаващи определен диагонал, съответно прав или опак, който съдържа клетката (R,C) (така зададеният диагонал е единствен)

(0 ≤ R, C ≤ n-1)

 Брой завъртания се задава:

 1. С единствено цяло число **k** (-1000 ≤ *k* ≤ 1000)

Когато някой от приятелите на Иванчо трябва да се завърти от крайна позиция (т.е. от (n-1)-ва към следващата) приемаме, че редът „превърта“ и съответният човек отива на 0-лева позиция.

Например:

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 6 | 4 | 5 |
| 7 | 8 | 9 |

r 1 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 6 | 4 | 5 |
| 7 | 8 | 9 |

|  |  |  |
| --- | --- | --- |
| 6 | 2 | 3 |
| 7 | 4 | 5 |
| 1 | 8 | 9 |

c 0 -1

|  |  |  |
| --- | --- | --- |
| 6 | 2 | 3 |
| 8 | 4 | 5 |
| 1 | 7 | 9 |

|  |  |  |
| --- | --- | --- |
| 6 | 2 | 3 |
| 7 | 4 | 5 |
| 1 | 8 | 9 |

d 2 1 -1

|  |  |  |
| --- | --- | --- |
| 6 | 2 | 1 |
| 8 | 3 | 5 |
| 4 | 7 | 9 |

|  |  |  |
| --- | --- | --- |
| 6 | 2 | 3 |
| 8 | 4 | 5 |
| 1 | 7 | 9 |

x 1 1 1

**Вход**

От първия ред на файла rotate.in се въвеждат 1 цяло число, N – съответно страната на квадрата, които образуват приятелите на Иванчо.

На следващия ред се въвежда цяло число R - броя на завъртанията, които ще бъдат извършени.

На следващите R реда се задават завъртанията, по гореописания начин.

На следващия ред се въвежда цялото числото Q - броят на заявки от типа „Кой от приятелите на Иванчо стои на позиция **(i, j)** след като са направени всички завъртания?“.

Всеки от следващите Q реда съдържа две цели числа **i** и **j**

**Изход**

В изходния файл rotate.out отпечатайте по едно цяло число – номерът на позиция **(i, j)** след изпълняване на указаните завъртания за всяка от Q-те заявки.

**Ограничения**

1 ≤ *N* ≤ 1000

1 ≤ R ≤ 104

1 ≤ *Q* ≤ 105

0 ≤ Т ≤ n-1

0 ≤ R, C ≤ n-1

-1000 ≤ *k* ≤ 1000

0 ≤ i, j ≤ n-1

**Ограничение за време: 0.7 сек**

**Ограничение за памет: 256 MB**

**Примерен тест**

|  |  |
| --- | --- |
| **Вход (rotate.in)** | **Изход (rotate.out)** |
| 34r 1 1c 0 -1d 2 1 -1x 1 1 130 01 12 1 | 637 |

 **Пояснения**

1. „Главен диагонал“ в една матрица е диагоналът, определен от всички клетки с еднакви индекси (т.е. 0,0 1,1 … n-1,n-1). „Прав диагонал“ ще наричаме диагонал, успореден на главния. Завъртане става в посока „надолу“, когато k > 0 и обратно, когато k < 0, т.е. след d 0 0 1, а00 отива на мястото на а11, а11 отива на мястото на а22 и т.н. След d 0 0 -1 а00 отива на мястото на аn-1,n-1 и т.н.

2. „Обратен диагонал“ в една матрица е диагоналът, определен от всички клетки с индекси (i, n-i-1), например клетките (0,3),(1,2),(2,1),(3,0) представляват обратния диагонал в матрица 4x4. В задачата опак диагонал ще наричаме диагонал, успореден на обратния. Завъртане става в посока „надолу“, когато k > 0 и обратно, когато k < 0.