Alieni

Il nostro satellite ha appena scoperto una civiltà aliena su un lontano pianeta. Abbiamo già preso una foto a bassa risoluzione di una regione del pianeta, di forma quadrata, che mostra molti segni di vita intelligente. I nostri esperti hanno identificato n punti di interesse nella foto, numerati da 0 a $n-1$. Ora vogliamo scattare delle foto ad alta risoluzione che contengano tutti questi n punti.

Internamente, il satellite ha suddiviso la regione della foto a bassa risoluzione in una griglia m per m di celle quadrate di lato unitario. Le righe e le colonne sono numerate consecutivamente da 0 a $m-1$ (dall'alto e da sinistra, rispettivamente). Usiamo (s,t) per denotare la cella nella riga s e nella colonna t. Il punto numero i è posto all'interno della cella (r_i, c_i). Ogni cella può contenere un arbitrario numero di questi punti.

Il nostro satellite è in un'orbita stabile che passa direttamente sopra la diagonale *principale* della griglia. La diagonale principale è la diagonale che collega il vertice in alto a sinistra con il vertice in basso a destra della griglia.

Il satellite può scattare una foto ad alta risoluzione di ogni regione che soddisfi le seguenti condizioni:

- la regione è un quadrato,
- due vertici opposti di questo quadrato sono sulla diagonale principale della griglia,
- ogni cella della griglia è o completamente all'interno o completamente all'esterno della regione fotografata.

Il satellite può scattare al più k foto ad alta risoluzione.

Dopo aver finito di fotografare, il satellite trasmetterà alla base le foto ad alta risoluzione di ogni cella fotografata (indipendentemente dal fatto che questa cella contenga o meno punti di interesse). I dati per ogni cella fotografata saranno trasmessi *solo una volta*, anche se la cella è stata fotografata più volte.

Dobbiamo scegliere k regioni quadrate da fotografare, assicurandoci che:

- ogni cella contenente almeno un punto di interesse sia fotografata almeno una volta, e
- il numero di celle fotografate almeno una volta sia minimo.

Il tuo compito è di trovare questo minimo numero totale di celle fotografate.

Dettagli di implementazione

Devi implementare la seguente funzione (metodo):

- `int64 take_photos(int n, int m, int k, int[] r, int[] c)`
  - `n`: il numero di punti di interesse,
  - `m`: il numero di righe (e di colonne) nella griglia,
  - `k`: il massimo numero di foto che il satellite può scattare,
  - `r` e `c`: due array di lunghezza n che descrivono le coordinate delle celle della griglia contenenti i punti di interesse. Per ogni indice i compreso tra 0 ed `n` -1, l'i-esimo punto di interesse è nella cella (`r[i]`, `c[i]`),
  - la funzione deve restituire il minor numero possibile di celle fotografate almeno una volta (con la condizione che le foto devono coprire tutti i punti di interesse).

Vedi i template forniti per ulteriori dettagli di implementazione nel tuo linguaggio di programmazione.

Esempi

Esempio 1

```
take_photos(5, 7, 2, [0, 4, 4, 4, 4], [3, 4, 6, 5, 6])
```

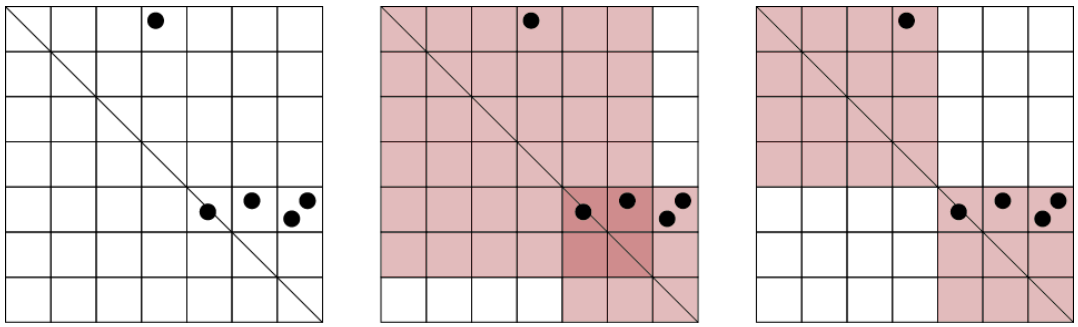
In questo esempio abbiamo una griglia 7×7 con 5 punti di interesse. I punti di interesse sono posti in quattro celle: $(0, 3)$, $(4, 4)$, $(4, 5)$, $(4, 6)$. Puoi scattare al più 2 foto ad alta risoluzione.

Un modo di catturare tutti i cinque punti di interesse è di scattare due foto: una con il quadrato 6×6 contenente le celle $(0, 0)$ e $(5, 5)$ come vertici, e un'altra del quadrato 3×3 contenente le celle $(4, 4)$ e $(6, 6)$ come vertici. Se il satellite scatta queste foto, dovrà trasmettere dati riguardo a 41 celle. Questa strategia non è ottima.

La soluzione ottima scatta una foto del quadrato 4×4 contenente le celle $(0, 0)$ e $(3, 3)$, e un'altra foto del quadrato 3×3 contenente le celle $(4, 4)$ e $(6, 6)$. In questo modo sono fotografate solo 25 celle, e questa è la soluzione ottima. Quindi `take_photos` deve restituire 25.

Nota che è sufficiente fotografare la cella $(4, 6)$ una sola volta, anche se contiene due punti di interesse.

Questo esempio è rappresentato nelle figure qui di seguito. Al centro, la soluzione subottimale, con 41 celle fotografate. A destra, la soluzione ottima.

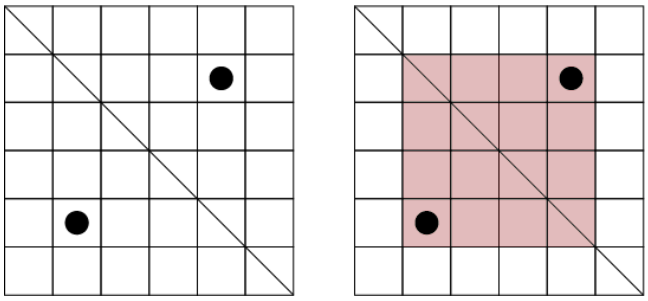


Esempio 2

```
take_photos(2, 6, 2, [1, 4], [4, 1])
```

In questo caso abbiamo 2 punti di interesse posti simmetricamente, nelle celle $(1, 4)$ e $(4, 1)$. Ogni foto valida che contiene uno dei due conterrà anche l'altro. Quindi, è sufficiente scattare una sola foto.

La figura qui sotto rappresenta l'esempio e la sua soluzione ottima. Con questa soluzione, il satellite cattura una sola foto di 16 celle.



Subtask

In ogni subtask, $1 \leq k \leq n$.

1. (4 punti) $1 \leq n \leq 50$, $1 \leq m \leq 100$, $k = n$,
2. (12 punti) $1 \leq n \leq 500$, $1 \leq m \leq 1000$, $r_i = c_i$ per ogni $0 \leq i \leq n - 1$,
3. (9 punti) $1 \leq n \leq 500$, $1 \leq m \leq 1000$,
4. (16 punti) $1 \leq n \leq 4000$, $1 \leq m \leq 1\,000\,000$,
5. (19 punti) $1 \leq n \leq 50\,000$, $1 \leq k \leq 100$, $1 \leq m \leq 1\,000\,000$,
6. (40 punti) $1 \leq n \leq 100\,000$, $1 \leq m \leq 1\,000\,000$.

Grader di esempio

Il grader di esempio legge l'input nel formato seguente:

- riga 1: gli interi n, m e k,
- riga 2 + i ($0 \leq i \leq n - 1$): gli interi r_i e c_i.