כרטיסים לקרנבל (tickets)

רינגו (Ringo) נמצא בקרנבל בסינגפור. יש לו כמה כרטיסי מזל בתיק, שהוא מעוניין להשתמש בהם בדוכן משחק המזל. כל כרטיס מגיע באחד מ-n צבעים ומודפס עליו מספר שלם אי שלילי. המספרים המודפסים על כרטיסים שונים עשויים להיות זהים. עקב תכונה מוזרה בחוקי הקרנבל, מובטח כי n הוא מספר **זוגי**.

לרינגו יש m כרטיסים מכל צבע בתיק, שהם בסך הכל $n \cdot m$ כרטיסים. על הכרטיס ה-j מהצבע ה-i מודפס המספר $x[i][j]$ ($0 \leq i \leq n-1$ וגם $0 \leq j \leq m-1$).

משחק המזל משוחק ב- k סבבים, הממוספרים מ-0 עד $k-1$. כל סבב משוחק באופן הבא:

- מתוך התיק שלו, רינגו בוחר **סט** של n כרטיסים, כרטיס אחד מכל צבע. לאחר מכן הוא נותן את הסט למפעיל המשחק.
- מפעיל המשחק רושם את המספרים $a[0], a[1] \ \ldots \ a[n-1]$ המודפסים על הכרטיסים בסט. הסדר של n המספרים לא משנה.
- מפעיל המשחק שולף קלף מיוחד מקופסת המזל ורושם את המספר השלם b שמודפס עליו.
- מפעיל המשחק מחשב את הערך המוחלט של ההפרש בין $a[i]$ ל-b עבור כל i מ-0 עד $n-1$. נגדיר את S להיות סכום הפרשים מוחלטים אלו.
- עבור סבב זה, מפעיל המשחק נותן לרינגו פרס שערכו שווה ל-S.
- הכרטיסים בסט מושלכים ולא ניתן להשתמש בהם בסבבים עתידיים.

הכרטיסים שנשארו בתיק של רינגו לאחר k סבבי המשחק מושלכים.

על ידי צפייה מקרוב, רינגו מבין כי המשחק מכור! קיימת מדפסת בתוך קופסת המזל. בכל סבב, מפעיל המשחק מוצא מספר b שממזער את ערך הפרס של הזכייה בסבב הזה. הערך הנבחר על ידי מפעיל המשחק מודפס על הקלף המיוחד של אותו הסבב.

בהינתן כל המידע הזה, רינגו רוצה להקצות כרטיסים לסבבים של המשחק. הוא רוצה לבחור את סט הכרטיסים המתאים לשימוש בכל סבב על מנת למקסם את הערך הכולל של הפרסים.

פרטי מימוש

עליכם לממש את הפונקציה הבאה:

```
int64 find_maximum(int k, int[][] x)
```

- k: מספר הסבבים.
- x: מערך בגודל $n \times m$ המתאר את המספרים על כל כרטיס. כרטיסים מכל צבע מסודרים בסדר לא יורד של המספרים שלהם.

- פונקציה זו נקראת בדיוק פעם אחת.
- על הפונקציה לבצע בדיוק קריאה אחת ל-allocate_tickets (ראה למטה), עם תיאור הקצאה של k סטים של כרטיסים, אחד לכל סבב. ההקצאה צריכה למקסם את הערך הכולל של הפרסים.
- על הפונקציה להחזיר את הערך המקסימלי הכולל של שווי הפרסים.

הפונקציה allocate_tickets מוגדרת באופן הבא:

```
void allocate_tickets(int[][] s)
```

- s: מערך בגודל $n \times m$. הערך של $s[i][j]$ צריך להיות r אם הכרטיס ה-j מהצבע ה-i כלול בסט של הסבב ה-r במשחק, או -1 אם הוא לא בשימוש כלל.
- עבור כל $0 \leq i \leq n-1$, מבין $s[i][0], s[i][1], \ldots, s[i][m-1]$ כל ערך $0, 1, 2, \ldots, k-1$ חייב להופיע בדיוק פעם אחת, וכל שאר הרשומות חייבות להיות -1.
- אם ישנן כמה הקצאות שנותנות את הסכום המקסימלי הכולל של שווי הפרסים, מותר לדווח על כל אחת מהן.

דוגמאות

דוגמה 1

הביטו בקריאה הבאה:

```
find_maximum(2, [[0, 2, 5],[1, 1, 3]])
```

משמעותה היא:

- ישנם $k = 2$ סבבים;
- המספרים שמודפסים על הכרטיסים מהצבע ה-0 הם 0, 2 ו- 5, בהתאמה;
- המספרים שמודפסים על הכרטיסים מהצבע ה-1 הם 1, 1 ו- 3, בהתאמה.

הקצאה אפשרית שנותנת את הסכום המקסימלי הכולל של שווי הפרסים היא:

- בסבב 0, רינגו בוחר בכרטיס 0 מהצבע ה-0 (שמודפס עליו המספר 0) וכרטיס 2 מהצבע 1 (שמודפס עליו המספר 3). ערך הזכייה הקטן ביותר האפשרי בסבב זה הוא 3. למשל: מפעיל המשחק יכול לבחור $b = 1$: $|1 - 0| + |1 - 3| = 1 + 2 = 3$.
- בסבב 1, רינגו בוחר בכרטיס 2 מהצבע ה-0 (שמודפס עליו המספר 5) וכרטיס 1 מהצבע 1 (שמודפס עליו המספר 1). ערך הזכייה הקטן ביותר האפשרי בסבב זה הוא 4. למשל: מפעיל המשחק יכול לבחור $b = 3$: $|3 - 1| + |3 - 5| = 2 + 2 = 4$.
- לכן, הסכום הכולל של שווי הפרסים יהיה $3 + 4 = 7$.

כדי לדווח על ההקצאה הזאת, הפונקציה find_maximum צריכה לבצע את הקריאה הבאה ל allocate_tickets:

- allocate_tickets([[0, -1, 1], [-1, 1, 0]])

לבסוף, הפונקציה `find_maximum` צריכה להחזיר 7.

דוגמה 2

הביטו בקריאה הבאה:

```
find_maximum(1, [[5, 9], [1, 4], [3, 6], [2, 7]])
```

משמעותה היא:

- יש רק סבב אחד,
- המספרים שמודפסים על הכרטיסים מהצבע ה- 0 הם 5 ו- 9, בהתאמה;
- המספרים שמודפסים על הכרטיסים מהצבע ה- 1 הם 1 ו- 4, בהתאמה;
- המספרים שמודפסים על הכרטיסים מהצבע ה- 2 הם 3 ו- 6, בהתאמה;
- המספרים שמודפסים על הכרטיסים מהצבע ה- 3 הם 2 ו- 7, בהתאמה.

הקצאה אפשרית שנותנת את הסכום המקסימלי הכולל של שווי הפרסים היא:

- בסבב 0, רינגו בוחר בכרטיס 1 מהצבע ה-0 (שמודפס עליו המספר 9) כרטיס 0 מהצבע 1 (שמודפס עליו המספר 1), כרטיס 0 מהצבע 2 (שמודפס עליו המספר 3) וכרטיס 1 מהצבע 3 (שמודפס עליו המספר 7). ערך הזכייה הקטן ביותר האפשרי בסבב זה הוא 12. למשל: מפעיל המשחק יכול לבחור $b = 3$: $|3 - 9| + |3 - 1| + |3 - 3| + |3 - 7| = 6 + 2 + 0 + 4 = 12$.

כדי לדווח על הפתרון, הפונקציה `find_maximum` צריכה לבצע את הקריאה הבאה ל `allocate_tickets`:

- `allocate_tickets([[-1, 0], [0, -1], [0, -1], [-1, 0]])`

לבסוף, הפונקציה `find_maximum` צריכה להחזיר 12.

מגבלות

- $2 \leq n \leq 1500$ ו- n הוא זוגי.
- $1 \leq k \leq m \leq 1500$
- $0 \leq x[i][j] \leq 10^9$ (לכל $0 \leq i \leq n - 1$ ו- $0 \leq j \leq m - 1$)
- $x[i][j - 1] \leq x[i][j]$ (לכל $0 \leq i \leq n - 1$ ו- $1 \leq j \leq m - 1$)

תתי משימות

1. (11 נקודות) $m = 1$
2. (16 נקודות) $k = 1$
3. (14 נקודות) $0 \leq x[i][j] \leq 1$ (לכל $0 \leq i \leq n - 1$ ו- $0 \leq j \leq m - 1$)
4. (14 נקודות) $k = m$
5. (12 נקודות) $n, m \leq 80$

6. $n, m \leq 300$ (23 נקודות)
7. (10 נקודות) ללא מגבלות נוספות.

גריידר לדוגמה

הגריידר לדוגמה קורא את הקלט לפי הפורמט הבא (השורות נקראות משמאל לימין):

- שורה 1: n m k
- שורה $2 + i$ $(0 \leq i \leq n - 1)$: $x[i][0]$ $x[i][1]$ $\ldots$ $x[i][m-1]$

הגריידר לדוגמה מדפיס את התשובה שלכם לפי הפורמט הבא (השורות נקראות משמאל לימין):

- שורה 1: ערך החזרה של `find_maximum`
- שורה $2 + i$ $(0 \leq i \leq n - 1)$: $s[i][0]$ $s[i][1]$ $\ldots$ $s[i][m-1]$