PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACION

Diseño y Análisis de Algoritmos - IIC2283
Tarea 2
Entrega: 9 de octubre (hasta las 23:59pm)

La solución de cada problema debe ser un programa en Python 3. Estos programas deben ser entregadas a través del Siding. El formato de entrega será informado en el foro del curso en el Siding. En este foro además puede hacer preguntas sobre los problemas de la tarea. Si quiere usar alguna librería en sus soluciones debe preguntar en el foro del curso si esta librería está permitida. El foro es el canal de comunicación oficial para todas las tareas.

Nota: Para esta tarea se permite el uso de la libreria `cmath`.

Problema 1

Se tiene tres conjuntos $S = \{1, \ldots, n\}$, A y B tales que $A \cup B \subseteq S$. Para un conjunto $T \subseteq S$ definimos $\alpha(T) = |T \cap A|$ y $\beta(T) = |T \cap B|$. Un conjunto T se dice A-*estable* si $\alpha(T) \geq \frac{|T|}{2}$ y se dice B-*estable* si $\beta(T) \geq \frac{|T|}{2}$. Además, se tiene las funciones $f_A : S \to \mathbb{N}$ y $f_B : S \to \mathbb{N}$, y partir de ellas se define las funciones F_A y F_B como:

$$F_A(T) = \begin{cases} \sum_{s \in T} f_A(s) & \text{si } T \text{ es } A\text{-estable} \\ 0 & \text{en otro caso} \end{cases}$$

$$F_B(T) = \begin{cases} \sum_{s \in T} f_B(s) & \text{si } T \text{ es } B\text{-estable} \\ 0 & \text{en otro caso.} \end{cases}$$

En esta pregunta usted debe recibir los conjuntos S, A, B y las funciones f_B, f_A, y debe encontrar un $T \subseteq S$ que maximice el valor de $F(T) = F_A(T) + F_B(T)$.

Formato

Sean $n_A = |A|$, $n_B = |B|$, $A = \{a_1, \ldots, a_{n_A}\}$ y $B = \{b_1, \ldots, b_{n_B}\}$. El input consistirá de cinco líneas separadas por espacios:

$n\ n_A\ n_B$
$a_1\ \cdots\ a_{n_A}$
$b_1\ \cdots\ b_{n_B}$
$f_A(1)\ \cdots\ f_A(n)$
$f_B(1)\ \cdots\ f_B(n)$

Su output debe ser una sóla linea con el valor $M = \text{máx}\{F(T) \mid T \subseteq S\}$.

Límites

$1 \leq n_A, n_B \leq n \leq 10^5$,
$1 \leq a_i \leq n$ para cada $i \in [1, n_A]$,
$1 \leq b_i \leq n$ para cada $i \in [1, n_B]$,
$0 \leq f_A(i), f_B(i) \leq 10^9$ para cada $i \in [1, n]$.

Tiempo de ejecución

2 segundos

Complejidad esperada

$O(n \log n)$.

Ejemplos

Input 1

```
4 1 1
1
3
2 1 1 1
1 1 2 2
```

Output 1

```
6
```

Input 2

```
4 1 1
1
3
2 1 1 1
1 1 2 5
```

Output 2

```
7
```

Input 3

```
4 2 2
1 2
2 3
5 5 5 5
5 5 5 5
```

Output 3

```
40
```

Problema 2

Como fue explicado en clases, un polinomio $p(x)$ de grado $n \geq 0$ se puede representar a través de un arreglo $A = [a_0, \ldots, a_n]$ de largo $n + 1$ de tal manera que:

$$p(x) \quad = \quad \sum_{i=0}^{n} a_i x^i.$$

En este problema su programa recibirá dos arreglos A y B que representan los polinomios $p(x)$ y $q(x)$ respectivamente, y deberá retornar un arreglo C que representa el polinomio $p(x) \cdot q(x)$.

Formato

El input consistirá de 2 líneas. La primera línea contiene un entero n indicando el grado del primer polinomio $p(x)$, seguido de $n+1$ enteros correspondientes a los coeficientes de $p(x)$. La segunda línea contiene un entero m indicando el grado del segundo polinomio $q(x)$, seguido de $m+1$ enteros correspondientes a los coeficientes de $q(x)$.

Su output debe ser una única línea, consistente en un entero k indicando el grado del polinomio $p(x) \cdot q(x)$, seguido de $k + 1$ enteros correspondientes a los coeficientes de $p(x) \cdot q(x)$.

Límites

$0 \leq n, m \leq 4 \times 10^4$
$-10^9 \leq a_i \leq 10^9$, donde a_i es el i-ésimo coeficiente de $p(x)$, $i \in [0, n]$
$-10^9 \leq b_i \leq 10^9$, donde b_i es el i-ésimo coeficiente de $q(x)$, $i \in [0, m]$
$a_n \neq 0$
$b_m \neq 0$

Tiempo de ejecución

2 segundos

Complejidad esperada

$O(\text{máx}(n, m) \log(\text{máx}(n, m)))$

Ejemplos

Input 1

```
2 1 0 5
1 0 -2
```

Output 1

```
3 0 -2 0 -10
```

Input 2

```
5 1 2 3 4 5 6
3 -1 1 -3 2
```

Output 2

```
8 -1 -1 -4 -5 -6 -7 -1 -8 12
```

Problema 3

Se tiene un arreglo de enteros $A = [a_1, \ldots, a_n]$. Considere el siguiente arreglo de 3-tuplas:

$$C = \{(i, j, k) \mid a_i + a_j = a_k \text{ e } i, j, k \text{ son distintos}\}.$$

Su programa debe recibir A y retornar $|C|$.

Formato

El input consistirá de dos líneas separadas por espacios:

n
$a_1 \ \cdots \ a_n$

Su output debe ser una sola línea con el valor $|C|$.

Límites

$1 \leq n \leq 10^4$
$|a_i| \leq n$ para cada $i \in [1, n]$

Tiempo de ejecución

2 segundos

Complejidad esperada

$O(n \log n)$

Ejemplos

Input 1

```
5
1 2 3 4 5
```

Output 1

```
8
```

Input 2

```
10
0 0 1 1 1 2 2 2 -1 -1
```

Output 2

```
150
```

Problema 4

Hay una tienda con un total de T objetos, los cuales están distribuidos en N estantes. El $i-$ésimo estante tiene a K_i objetos apilados verticalmente, de manera tal que $T = \sum_{i=1}^{N} K_i$. Cada objeto tiene un precio base asociado. Usted tiene como objetivo comprar todos los objetos de la tienda, pero lamentablemente hay ciertas reglas para comprar que usted debe respetar y que pueden determinar que el monto final que usted pague no sea fijo:

1. Para comprar un objeto, usted debe retirarlo antes de su estante (no puede comprar objetos que todavía están en un estante).
2. Usted sólo puede comprar una vez en su vida en dicha tienda, y sólo está permitido hacer una sola canasta de compra. Por lo tanto, para comprar los T objetos usted debe primero sacarlos todos de sus estantes, agregarlos a una única canasta de compra y finalmente comprarlos.
3. Usted sólo tiene permitido sacar un objeto que en ese instante **se encuentre en el tope de algún estante**. Al sacarlo, dicho objeto es removido del tope del estante y el objeto inmediatamente abajo pasa a estar en el tope (a menos que el estante se vacíe).
4. Cuando usted saca un objeto del tope de un estante, debe esperar **por lo menos 1 minuto** para sacar otro objeto (de cualquier estante, si es que quedan) o bien dar por terminada su canasta de compra y proceder a comprar. Además, cuando usted saca un objeto de su estante se asume que lo va a comprar (no puede devolverlo).
5. Si el objeto $j-$ésimo es sacado en el instante t y usted paga su canasta de compra definitiva en el instante F (donde claramente $F > t$), para la tienda esto implica que dicho objeto estuvo en posesión suya sin comprar por un período de $M = F - t$ minutos, por lo cual tendrá que pagar un precio de $B_j \times I^M$, donde B_j es el precio base del objeto $j-$ésimo e I es el coeficiente de interés de la tienda, de manera tal que I^M es el interés por los M minutos que tuvo el objeto j-ésimo en su canasta.

Dadas estas reglas, ¿cuál es el mínimo monto que usted tendría que pagar por una canasta de compra con todos los objetos de la tienda, si usted sacara los objetos de los estantes de manera óptima?

Formato

El input consta de varias líneas. La primera línea contiene dos enteros N e I separados por un espacio, correspondientes al número de estantes y el coeficiente de interés de la tienda, respectivamente. Luego vienen N líneas describiendo el contenido de los N estantes de la tienda. De manera precisa, la linea $i \in [1, N]$, que describe al estante i, contiene un entero K_i seguido por K_i enteros $B_{i,1}, B_{i,2}, \ldots, B_{i,K_i}$, donde K_i es el número de objetos en el estante y $B_{i,j}$ es el precio base del $j-$ésimo objeto en dicho estante (los precios base de los objetos son dados en el orden en que están apilados en el estante, de abajo hacia arriba).
Su output debe ser una única línea con el mínimo monto a pagar por todos los objetos, pero dado que este monto puede ser muy grande, se pide su valor **módulo** $10^9 + 7$.

Límites

$1 \leq N \leq 10^4$
$2 \leq I \leq 500$
$1 \leq K_i$ para cada $i \in [1, N]$
$T = \sum_{i=1}^{N} K_i \leq 4 \times 10^4$
$1 \leq B_{i,j} < I$ para cada $i \in [1, N]$ y $j \in [1, K_i]$

Tiempo de ejecución

2 segundos

Complejidad esperada

$O(T \log(T))$

Ejemplos

Input 1

```
4 400
1 1
1 2
1 3
1 4
```

Output 1

```
728481425
```

Input 2

```
3 500
3 7 2 5
4 10 6 4 4
4 8 2 1 3
```

Output 2

```
916715981
```