Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2283 - Diseño y Análisis de Algoritmos - 2do semestre 2022

TAREA 4

Publicación:	Martes 18 de octubre.
GitHub Classroom:	https://classroom.github.com/a/FtD-2MFl
Entrega:	**Jueves 27 de octubre 23:59 horas**.

Indicaciones

- La tarea es estrictamente individual.
- La solución puede ser entregada en el archivo `t4.py` del repositorio privado asignado mediante GitHub Classroom para esta tarea. Se revisará el último *commit* subido antes de la entrega al repositorio. Se usará Python 3.10.X para la revisión.
- El *input* para el programa debe ser obtenido desde *standard input*. El *output* debe ser entregado mediante *standard output*.
- La corrección se realizará mediante *tests* automatizados acordes al formato de *input* y *output* especificado. Cada *test* tendrá un *timeout* según lo que se especifica como tiempo esperado.
- Un *test* se considerará **reprobado** en caso de que 1) dado el *input* el *output* sea incorrecto, 2) exista un error de *runtime* durante su ejecución, o 3) el *timeout* se cumpla durante su ejecución. En otro caso, el *test* se considerará **aprobado**.
- No se permite el uso de librerías externas a la librería estándar de Python *a priori*. Consultar en las *issues* del repositorio oficial del curso en caso de requerir una.
- **Para esta tarea sí aplica la política de atrasos descrita en el programa del curso.**

Problema

Renata quiere abrir un nuevo cine en la ciudad, y por lo tanto debe decidir cuál será la cartelera que presentará durante el año. Para decidir esto, Renata consiguió *legalmente* datos de usuarios de Netflix que viven en la ciudad. Estos datos son tal que, para cada usuario, se asocia una lista de a lo más p películas aún sin estrenar que a este le interesan, basado en el historial de películas que este ya ha visto.

Renata entonces quiere conocer, dado un conjunto M películas que se van a estrenar este año y un estudio de n usuarios de Netflix con las preferencias de estos para estas M películas, un subconjunto maximal[1] $S \subseteq M$ de películas tal que al menos $\lceil \frac{n}{2} \rceil$ usuarios de Netflix estén interesados en todas las películas presentes en S según el estudio.

[1] Con conjunto maximal nos referimos a un conjunto para el cuál no exista un conjunto de mayor tamaño que cumpla las condiciones que se piden. Nótese que pueden existir varios subconjuntos maximales.

Input

La primera línea contiene un tres números enteros: n, m y p, separados por un espacio. Así, n ($1 \leq n \leq 2 \cdot 10^5$) es la cantidad de usuarios de Netflix, m ($1 \leq m \leq 60$) es la cantidad total de películas que se estrenarán en el año y cada usuario está interesado en a lo más p películas ($1 \leq p \leq \min\{m, 15\}$).

Después vienen n líneas de largo m que representan la matriz de $n \times m$ que codifica los intereses de los n usuarios para las m películas. Así, la i-ésima línea es un string $b_1 \cdots b_m$ de ceros y unos tal que

$$b_j = \begin{cases} 1 & \text{al } i\text{-ésimo usuario le interesa la } j\text{-ésima película} \\ 0 & \text{al } i\text{-ésimo usuario NO le interesa el } j\text{-ésima película} \end{cases}$$

Output

Entrega un string $s = s_1 \cdots s_m$ de ceros y unos que representa un subconjunto maximal S según lo pedido. Es decir, si $P = \{p_1 \dots p_m\}$ es el conjunto de películas que se estrenarán este año, entonces $s_i = 1$ si, y solo si, $p_i \in S$.

Si hay múltiples conjuntos maximales, basta con entregar cualquiera de ellos.

Tiempo esperado

Se espera que la solución implemente un algoritmo aleatorizado de Monte Carlo, tal que la probabilidad de que se retorne una respuesta incorrecta sea menor a $\frac{1}{1000}$. Se espera que la solución se ejecute en un tiempo **menor o igual a 3 segundos**.

Complejidad esperada

Se espera que la solución posea una complejidad de $\mathcal{O}(p \cdot (2^p + n) + m \cdot n)$.

Hint: Considere la posibilidad de elegir subconjuntos de manera aleatorizada para constuir el algoritmo de Monte Carlo.

Ejemplo

Los siguientes tests están ya cargados a GitHub Classroom con corrección automática mediante GitHub Actions. **Los tests para la corrección serán distintos a estos.**

Input
3 4 3 1000 0110 1001
Output
1000

Input
5 5 4 11001 10101 10010 01110 11011
Output
10001