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

Diseño y Análisis de Algoritmos - IIC2283
Tarea 2
Entrega: lunes 8 de noviembre (hasta las 11:59pm)

La solución de cada problema debe ser un programa en Python 3. Estos programas deben ser subidos a su repositorio de git creado para este propósito. Sus archivos deben llamarse `p1.py` y `p2.py` para las preguntas 1 y 2 respectivamente, y estar en la carpeta `T2` de su repositorio. Cualquier pregunta de la tarea hágala en el foro del curso (`https://github.com/marceloarenassaavedra/IIC2283-2-21/issues`). Si quiere usar alguna librería en sus soluciones debe preguntar en el foro si esta librería está permitida. El foro es el canal de comunicación oficial para todas las tareas.

Problema 1

Usted ha sido contratado por un laboratorio de investigación del genoma humano como parte del equipo de diseño, análisis e implementacion de algoritmos genómicos. En particular, a usted se le ha encargado desarrollar un algoritmo de detección de cadenas de ADN humano dentro de otra cadena más larga. Esto puede verse como un problema de *strings*, donde el alfabeto consiste en los caracteres `A`, `T`, `G` y `C`, y donde el problema se reduce a encontrar las ocurrencias de un string dentro de otro.

No obstante, hay una complicación adicional. Debido a la posibilidad de mutaciones en el ADN, se le pide considerar un umbral de tolerancia de error k. Concretamente, diremos que el string T ocurre en el string S en la posición i si es que al alinear T junto a S a partir de esta posición, cada caracter de T es igual a algún caracter en S ubicado a una distancia de no más de k posiciones. De manera formal, para cada $j \in [1, |T|]$ existe $p \in [1, |S|]$ tal que $|p - i + 1 - j| \leq k$ y $S[p] = T[j]$. Notar que con $k = 0$ esto es equivalente a la definición tradicional de ocurrencia de un string dentro de otro.

Dados un par de strings S y T, y un umbral de error k, usted deberá retornar el número de ocurrencias de T en S considerando dicho umbral.

Formato

La primera línea del input contiene tres enteros $|T|, |S|$ y k, después vienen dos líneas, la primera contiene a T y la segunda contiene a S.

El output debe ser una línea con el número de ocurrencias del string T en S considerando el umbral de error k.

Límites

$1 \leq |T| \leq |S| \leq 2 \cdot 10^5$
$0 \leq k \leq 2 \cdot 10^5$

Tiempo de ejecución

1 segundo

Complejidad esperada

$O(|S| \log |S|)$

Ejemplos

Input 1

```
10 4 0
GGCAATTCAT
GCAT
```

Output 1

```
0
```

Input 2

```
10 4 1
GGCAATTCAT
GCAT
```

Output 2

```
2
```

Hint

Notar que el alfabeto contiene 4 letras. Descomponer el problema por letra, resolver por separado y luego combinar resultados puede ser útil. Usar FFT puede que sea útil también. ☺

Problema 2

Martín está muy interesado en las criptomonedas, y está organizando una junta con sus n amigos para hablar de ellas. Pero él sabe que sus amigos no están interesados en las mismas, y por lo tanto quiere saber cuál es el **subconjunto más grande** de criptomonedas tal que al menos $\lceil \frac{n}{2} \rceil$ de sus amigos estén interesados en **todas** las criptomonedas del subconjunto.

Martín sabe qué criptomonedas le interesan a cada amigo, y sabe que a cada amigo le interesan a lo más p criptomonedas de las m que existen.

Formato

La primera línea del input contiene tres números n, m y p separados por un espacio, donde n es la cantidad de amigos de Martín, m es la cantidad total de criptomonedas y cada amigo está interesado en a lo más p criptomonedas. Después vienen n líneas, donde la i-ésima contiene un string binario s_i de largo m, donde el j-ésimo carácter de s_i es 1 si al i-ésimo amigo le interesa la j-ésima criptomoneda, y 0 en caso contrario.

El output es una sola línea que corresponde a un string binario que representa el subconjunto más grande. Nótese que si hay múltiples soluciones, entonces puede entregar cualquiera de ellas.

Límites

$1 \leq n \leq 2 \cdot 10^5$

$1 \leq p \leq m \leq 60$

$1 \leq p \leq 15$

Tiempo de ejecución

1 segundo

Complejidad y error esperado

Se espera una solución con un algoritmo aleatorizado de tipo Monte Carlo, donde la probilidad de retornar una respuesta incorrecta sea menor a 10^{-15} (i.e. probabilidad prácticamente nula). La complejidad esperada es $O(iter \cdot p \cdot (2^p + n) + n \cdot m)$, donde $iter$ es la cantidad de iteraciones. Además, se espera que $iter \in O(\log(1/\varepsilon))$ dado una probabilidad de error $\varepsilon \in (0, 1)$.

Ejemplos

Input 1

```
3 4 3
1000
0110
1001
```

Output 1

```
1000
```

Hint

Elegir subconjuntos al azar puede ser útil.