Vacaciones de Verano (vacaciones)

Descripción

Pronto llegarán las vacaciones de verano y prometiste invitar helado a tus N amigos.
Para no equivocarte al hacer el pedido, quieres saber, en tu grupo de amigos, qué sabores les gustan a todos.
Cada uno de los M sabores puede gustarle o no gustarle a cada amigo.

De camino a la heladería te cuestionas dos cosas:

1. **¿Qué helados debería comprar para un rango $[l, r]$ de amigos?**
   Necesitas saber qué sabores les gustan a **todos** los amigos dentro de ese rango.
2. **¿Y si un amigo cambia de opinión?**
   El amigo x invierte su gusto por el sabor k.

Implementación

Debes implementar la siguiente función en C++:

```
std::vector<std::string> sabores_comunes(
  int N, int M, std::vector<std::string> preferencias,
  int Q, std::vector<std::tuple<int,int,int>> consultas
);
```

O la siguiente función en Python:

```
def sabores_comunes(
  N: int, M: int, preferencias: list[str],
  Q: int, consultas: list[tuple[int,int,int]]
) -> list[str]:
```

Parámetros

- `N`: número de amigos.
- `M`: número de sabores.
- `preferencias`: lista de cadenas binarias de longitud M, donde el carácter j de la cadena i es `1` si al amigo i le gusta el sabor j, y `0` en caso contrario.
- `Q`: número de consultas.

- `consultas`: lista de tuplas (tipo, a, b) donde:
  - Si $tipo = 1$, $a = l$ y $b = r$, consulta qué sabores gustan a todos los amigos en el rango $[l, r]$.
  - Si $tipo = 2$, $a = x$ y $b = k$, el amigo x invierte su preferencia por el sabor k.

Nota: los índices de amigos y sabores son 1-based, es decir, comienzan en 1.

Retorna

- Una lista de cadenas binarias de longitud M, una por cada consulta de tipo `1`, indicando los sabores que gustan a todos los amigos del rango.
- Para consultas de tipo 2 no se retorna nada.

Ejemplos

En la siguiente llamada:

```
minimo_dias_para_plantar(7, [0, 1, 0, 2, 1, 0, 2], 2, [2, 1])
```

Debe retornar `5`

Ejemplos

Ejemplo 1

En la siguiente llamada:

```
minimo_dias_para_plantar(
    5, 3,
    ["010", "101", "001", "101", "101"],
    4,
    [{1, 1, 1}, {1, 4, 5}, {1, 2, 3}, {1, 1, 5}]
)
```

Debe retornar `["010" ,"101", "001", "000"]`

Explicación: Estado inicial:

- Amigo 1: `010` (le gusta el sabor 2)
- Amigo 2: `101` (le gustan los sabores 1, 3)
- Amigo 3: `001` (le gusta el sabor 3)
- Amigo 4: `101` (le gustan los sabores 1, 3)
- Amigo 5: `101` (le gustan los sabores 1, 3)

Consulta 1: `1 1 1` (sabores que les gustan a todos los amigos del 1 al 1)

- Solo consideramos al amigo 1: `010`

- Resultado: `010`

Consulta 2: `1 4 5` (sabores que les gustan a todos los amigos del 4 al 5)

- Sabor 1: Amigo 4 (SÍ), Amigo 5 (SÍ) → Como a todos les gusta: **1**
- Sabor 2: Amigo 4 (NO), Amigo 5 (NO) → Como no a todos les gusta: **0**
- Sabor 3: Amigo 4 (SÍ), Amigo 5 (SÍ) → Como a todos les gusta: **1**
- Resultado: `101`

Consulta 3: `1 2 3` (sabores que les gustan a todos los amigos del 2 al 3)

- Sabor 1: Amigo 2 (SÍ), Amigo 3 (NO) → Como no a todos les gusta: **0**
- Sabor 2: Amigo 2 (NO), Amigo 3 (NO) → Como no a todos les gusta: **0**
- Sabor 3: Amigo 2 (SÍ), Amigo 3 (SÍ) → Como a todos les gusta: **1**
- Resultado: `001`

Consulta 4: `1 1 5` (sabores que les gustan a todos los amigos del 1 al 5)

- Sabor 1: Amigo 1 (NO), Amigo 2 (SÍ), Amigo 3 (NO), Amigo 4 (SÍ), Amigo 5 (SÍ) → Como no a todos les gusta: **0**
- Sabor 2: Amigo 1 (SÍ), Amigo 2 (NO), Amigo 3 (NO), Amigo 4 (NO), Amigo 5 (NO) → Como no a todos les gusta: **0**
- Sabor 3: Amigo 1 (NO), Amigo 2 (SÍ), Amigo 3 (SÍ), Amigo 4 (SÍ), Amigo 5 (SÍ) → Como no a todos les gusta: **0**
- Resultado: `000`

Ejemplo 2

En la siguiente llamada:

```
minimo_dias_para_plantar(
    5, 5,
    ["01011", "11001", "11111", "00101", "10101"],
    4,
    [{1, 1, 3}, {2, 2, 5}, {1, 1, 3}]
)
```

Debe retornar `["01001", "01000"]`

Explicación: Estado inicial:

- Amigo 1: `01011` (le gustan los sabores 2, 4, 5)
- Amigo 2: `11001` (le gustan los sabores 1, 2, 5)
- Amigo 3: `11111` (le gustan todos los sabores)
- Amigo 4: `00101` (le gustan los sabores 3, 5)
- Amigo 5: `10101` (le gustan los sabores 1, 3, 5)

Consulta 1: `1 1 3` (sabores que les gustan a todos los amigos del 1 al 3)

- Sabor 1: Amigo 1 (NO), Amigo 2 (SÍ), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Sabor 2: Amigo 1 (SÍ), Amigo 2 (SÍ), Amigo 3 (SÍ) → Como a todos les gusta: **1**
- Sabor 3: Amigo 1 (NO), Amigo 2 (NO), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Sabor 4: Amigo 1 (SÍ), Amigo 2 (NO), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Sabor 5: Amigo 1 (SÍ), Amigo 2 (SÍ), Amigo 3 (SÍ) → Como a todos les gusta: **1**
- Resultado: `01001`

Operación 2: `2 2 5` (amigo 2 cambia de opinión sobre sabor 5)

- Amigo 2 tenía: `11001` (le gustaba el sabor 5)
- Amigo 2 ahora: `11000` (ya no le gusta el sabor 5)

Consulta 3: `1 1 3` (sabores que les gustan a todos los amigos del 1 al 3)

- Sabor 1: Amigo 1 (NO), Amigo 2 (SÍ), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Sabor 2: Amigo 1 (SÍ), Amigo 2 (SÍ), Amigo 3 (SÍ) → Como a todos les gusta: **1**
- Sabor 3: Amigo 1 (NO), Amigo 2 (NO), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Sabor 4: Amigo 1 (SÍ), Amigo 2 (NO), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Sabor 5: Amigo 1 (SÍ), Amigo 2 (NO), Amigo 3 (SÍ) → Como no a todos les gusta: **0**
- Resultado: `01000`

Límites

- $1 \leq N \leq 10^6$
- $1 \leq M \leq 64$
- $1 \leq Q \leq 10^5$
- $1 \leq l \leq r \leq N$ (para consultas tipo 1)
- $1 \leq x \leq N$, $1 \leq k \leq M$ (para consultas tipo 2)

Subtareas

Subtarea 1 (10 puntos)

"A todos les gustan todos los helados"

- $N \leq 1000$
- $M \leq 30$
- $Q \leq 1000$

- Solo consultas tipo 1
- **Garantía especial:** Inicialmente, todos los amigos tienen preferencias `111...1` (todos los sabores)

Subtarea 2 (10 puntos)

"Amor u odio total"

- $N \leq 1000$
- $M \leq 30$
- $Q \leq 1000$
- Solo consultas tipo 1
- **Garantía especial:** Cada amigo tiene preferencias `000...0` (odia todos) o `111...1` (ama todos)

Subtarea 3 (10 puntos)

- $N \leq 100$
- $M \leq 10$
- $Q \leq 100$
- Solo consultas tipo 1

Subtarea 4 (20 puntos)

- $N \leq 10^6$
- $M \leq 32$
- $Q \leq 1000$
- Solo consultas tipo 1

Subtarea 5 (10 puntos)

- $N \leq 1000$
- $M \leq 32$
- $Q \leq 1000$
- Ambos tipos de consulta

Subtarea 6 (40 puntos)

- **No hay restricciones adicionales.**

Entrada

- Solo debes subir el archivo `vacaciones.cpp` con tu solución.
- La entrada y salida serán manejadas por el archivo `grader.cpp` o `grader.py`.
- **No debes imprimir nada por la salida estándar** ni leer de la entrada estándar en tu archivo `vacaciones.cpp`.

- No modifiques los archivos `grader.cpp` o `grader.py`, ya que el juez solo tomará en cuenta tu solución en `vacaciones.cpp`.