CS2023 - Aula de Ejercicios Nº 7

Brenner H. Ojeda Rios

Semestre 2024-0

Se sugiere que cada estudiante trate de resolver los ejercicios de forma **individual** y luego los discuta en grupo.

Ejercicios

1. (5 pts) Se le proporciona una matriz binaria `mat` de $m \times n$ de 1's (que representan a los soldados) y 0's (que representan a los civiles). Los soldados se colocan delante de los civiles. Es decir, todos los 1 aparecerán a la izquierda de todos los 0 en cada fila.

   Una fila i es más débil que una fila j si se cumple una de las siguientes condiciones:

   - El número de soldados en la fila i es menor que el número de soldados en la fila j.
   - Ambas filas tienen el mismo número de soldados e $i < j$.

   Devuelve los índices de las k filas más débiles de la matriz ordenadas del más débil al más fuerte. Vea el siguiente ejemplo:

   Input:

   ```
   mat = [
           [1,1,0,0,0],
           [1,1,1,1,0],
           [1,0,0,0,0],
           [1,1,0,0,0],
           [1,1,1,1,1]
         ]
   k = 3
   ```

   Output:

   ```
   [2,0,3]
   ```

   Explicación:

   El número de soldados en cada fila es:

   - Fila 0: 2
   - Fila 1: 4
   - Fila 2: 1
   - Fila 3: 2
   - Fila 4: 5

   Las filas ordenadas de más débil a más fuerte son [2,0,3,1,4].

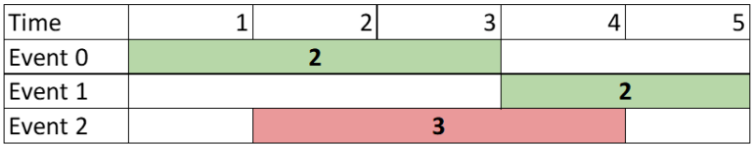
   Restricciones:

   - $2 \leq n, m \leq 100$
   - $1 \leq k \leq m$
   - `matriz[i][j]` es 0 o 1

2. (6 pts) Se le proporciona un arreglo 2D indexado en 0 denominado `eventos` donde eventos[i] = $[startTime_i, endTime_i, value_i]$. El i-ésimo evento comienza en $startTime_i$ y termina en $endTime_i$, y si asiste a este evento, recibirá un valor de $value_i$. Puede elegir como máximo dos eventos que no se superpongan para asistir de modo que se maximice la suma de sus valores.

   Su tarea es devolver esta suma máxima.

   Ten en cuenta que la hora de inicio y la hora de finalización son inclusivas: es decir, no puedes asistir a dos eventos donde uno de ellos comienza y el otro termina al mismo tiempo. Más específicamente, si asiste a un evento con hora de finalización t, el siguiente evento debe comenzar en $t + 1$ o después.



   Input:

   ```
   [[1,3,2],[4,5,2],[2,4,3]]
   ```

   Output: 4

   Explicación: Se escoge los eventos, 0 y 1 por sumar 2 + 2 = 4.

   Restricciones:

   - $2 \leq$ events.length $\leq 10^5$
   - $1 \leq startTime_i \leq endTime_i \leq 10^9$
   - $1 \leq value_i \leq 10^6$

3. (9 pts) La mediana es el valor medio en una lista de enteros ordenados. Si el tamaño de la lista es par, no hay un valor medio y la mediana es la media de los dos valores medios.

   - Por ejemplo, para `arr = [2,3,4]`, la mediana es 3.
   - Por ejemplo, para `arr = [2,3]`, la mediana es $(2 + 3)/2 = 2,5$.

   Implemente la clase `MedianFinder`:

   - `MedianFinder()` inicializa el objeto MedianFinder.
   - `void addNum(int num)` agrega el número entero del flujo de datos a la estructura de datos.
   - `double findMedian()` devuelve la mediana de todos los elementos hasta el momento. Se aceptarán respuestas dentro de 10^{-5} de la respuesta real.

   Input:

   ```
   ["MedianFinder", "addNum", "addNum", "findMedian", "addNum", "findMedian"]
   [[], [1], [2], [], [3], []]
   ```

   Output

   ```
   [null, null, null, 1.5, null, 2.0]
   ```

   Explicación:

   ```
   MedianFinder medianFinder = new MedianFinder();
   medianFinder.addNum(1);    // arr = [1]
   medianFinder.addNum(2);    // arr = [1, 2]
   medianFinder.findMedian(); // return 1.5 (i.e., (1 + 2) / 2)
   medianFinder.addNum(3);    // arr[1, 2, 3]
   medianFinder.findMedian(); // return 2.0
   ```

   Restricciones:

   - $-10^5 \leq num \leq 10^5$
   - Habrá al menos un elemento en la estructura de datos antes de llamar a findMedian.
   - Como máximo se realizarán 5×10^4 llamadas para `addNum` y `findMedian`.