L09: Todos al jardín

Estructuras de Datos
Facultad de Informática - UCM

Para realizar este ejercicio, descarga la plantilla que se proporciona en este enunciado (icono del clip al lado del título) o a través del Campus Virtual.

La entrega de este ejercicio consiste en un único fichero `.cpp` que se subirá a *DOMjudge*. Podéis subir tantos intentos como queráis. Se tendrá en cuenta el último intento con el veredicto `CORRECT` que se haya realizado antes de la hora de entrega por parte de alguno de los miembros de la pareja.

No olvidéis poner el nombre de los componentes de la pareja en el fichero `.cpp` que entreguéis. Solo es necesario que uno de los componentes de la pareja realice la entrega.

Evaluación: La práctica se puntuará de 0 a 10 si ha obtenido el veredicto `CORRECT` por parte del juez. En caso contrario, la calificación será de 0. No se realizará evaluación por pares en esta práctica.

En mi empresa son muy metódicos a la hora de asignar los despachos al personal. Los empleados, uno a uno, van escogiendo sus despachos en estricto orden de escalafón, comenzando por la presidenta y finalizando por los becarios. Cada cierto tiempo, la presidenta organiza una reorganización radical de los espacios, que ella misma denomina *Todos al jardín*. Consiste en sacar a todo el personal de sus despachos, dejándolos vacíos, para volver a asignarlos desde cero.

Este año nos toca un *Todos al jardín*. No sé si será por estrés o por aburrimiento, pero la presidenta ha ideado un nuevo sistema para el reparto de despachos que no me llega a convencer del todo. En este sistema, el personal escoge en orden de escalafón inverso (es decir, de menor a mayor categoría). Para cada empleado, se hace lo siguiente: Supongamos que escoge el despacho N. Si quedan puestos libres, se instala allí. Si no, la persona de menor categoría que esté en ese despacho tiene que recoger sus bártulos e irse al despacho $N + 1$. Si este último también esta lleno, tiene que ir al $N + 2$, y así sucesivamente hasta encontrar un despacho con algún puesto libre.

La peor parte viene cuando escogen los directivos, que siempre escogen en último lugar. Son bastante reacios a compartir despacho, por lo que si eligen un despacho N que esté ocupado, *todas* las personas que estén allí tendrán que abandonarlo y buscar refugio en el despacho $N+1$. Si este último puede acoger solamente a algunas de ellas, las restantes deberán probar suerte en el $N + 2$, y así sucesivamente hasta que estén todas redistribuidas. En este proceso hay que tener en cuenta que los directivos nunca comparten despacho. Por supuesto, un directivo de mayor rango en el escalafón puede desalojar a otro de menor rango. En este caso, el de menor rango se tendrá que mudar al siguiente despacho que no esté ocupado por otro directivo, y desalojar a todos los empleados que estén allí.

La presidenta acaba de encomendarme la tarea de llevar a cabo este nuevo sistema de asignación. Espero no equivocarme, por la cuenta que me trae.

Entrada

La entrada consiste en varios casos de prueba, cada uno de ellos consistente en tres líneas. La primera línea contiene tres números:

- C, que es el número máximo de personas que caben en cada despacho. Todos los despachos tienen la misma capacidad ($1 \leq C \leq 10\,000$).
- N, que es el número de empleados en la empresa ($0 \leq N \leq 15\,000$).
- M, que es el número de directivos en le empresa ($0 \leq M \leq 15\,000$)..

La segunda línea contiene N números comprendidos entre 1 y 10^9, que son los números de despacho escogidos por cada empleado, desde el de menor rango hasta el de mayor rango. La tercera línea contiene M números, que son las elecciones de despachos por parte de los directivos (de nuevo, de menor a mayor rango).

La entrada finaliza con tres ceros, que no se procesan.

Salida

Para cada caso de prueba, debe imprimirse un listado con la información de los despachos que estén ocupados tras el proceso de elección. La lista debe estar ordenada de modo ascendente según el número de despacho. Para cada despacho:

- Si acoge a un directivo, debe imprimirse una línea con el número de despacho seguido de la cadena `JEFE`.
- Si solamente contiene empleados, debe imprimirse una línea con el número de despacho seguido del número de empleados que lo ocupan.

El listado de despachos ha de ir finalizado con una línea con tres guiones (`---`).

Entrada de ejemplo

```
2 7 3
4 6 3 4 3 3 8
1 6 3
2 3 2
100 500 200
500 500
0 0 0
```

Salida de ejemplo

```
1 JEFE
3 JEFE
4 2
5 2
6 JEFE
7 2
8 1
---
100 1
200 1
500 JEFE
501 JEFE
502 1
---
```

Notas sobre la implementación

Para realizar este ejercicio puedes utilizar la clase `MapTree` vista en clase, o la clase `std::map` de la STL. Si decides utilizar `std::map`, ten en cuenta lo siguiente:

- En las librerías de la STL se introduce el método `contains()` a partir del estándar C++20. El compilador del juez aún no soporta este estándar. En su lugar, puedes utilizar el método `count()`, que devuelve `1` si la clave está en el diccionario, o `0` en caso contrario.
- Los iteradores de la clase `MapTree<K, V>` recorren sus entradas, que son objetos de tipo `MapEntry<K, V>` con dos atributos: `key` y `value`. En los iteradores de la clase `std::map<K, V>`, las entradas son objetos de tipo `std::pair<K, V>` con dos atributos: `first` y `second`. En cualquiera de los dos casos, puedes utilizar la sintaxis de C++17 para obtener ambas componentes a la vez:

  ```
  for (auto [k, v] : diccionario) {
    // ...
  }
  ```