INF-221 Algoritmos y Complejidad, 2023-1

Tarea 3

Profesor: Diego Arroyuelo
Ayudantes: Javier Pérez, Bayron Valenzuela
javier.perezp@usm.cl
bayron.valenzuela@sansano.usm.cl

Fecha de Inicio: 15 de junio, 2023
Fecha de Entrega: 30 de junio, 2023
Plazo máximo de entrega atrasada: 5 horas

Reglas del Juego

La presente tarea debe hacerse en grupos de 3 personas. Toda excepción a esta regla debe ser conversada con los ayudantes **ANTES** de comenzar la tarea. No se permiten de ninguna manera grupos de más de 3 personas. Pueden usarse los lenguajes de programación C, C++, Python, y Java.

Un Trabajador Ordenado

Un trabajador muy ordenado anota cada hora la tarea en la que está trabajando. Su profesión le obliga a concentrarse en una única tarea a la vez. Algunas tareas le toman más tiempo que otras, y si durante una hora trabaja en varias tareas, sólo anota en su bitácora aquella a la que le haya dedicado más tiempo (el resto se desecha y no importa para los registros). Después de todo, sus tareas son largas y es extraño que pueda trabajar en más de dos tareas distintas durante la misma hora. A cada tarea se le asigna un identificador entero consecutivo, comenzando con el identificador 1. De esta manera, la bitácora de tareas del trabajador es un arreglo que almacena los identificadores de las tareas correspondientes. Cada hora, un nuevo identificador es agregado al final del arreglo. Cada cierto tiempo, el trabajador es evaluado por su jefa, quien revisa la cantidad de tareas distintas en las que trabajó durante un período de tiempo consecutivo (sólo importa contar las tareas registradas en la bitácora del trabajador). La tarea consiste en desarrollar un algoritmo de tipo *decrecer y conquistar* que le permita a la jefa contar esa cantidad de tareas de forma eficiente: En un arreglo que tiene datos correspondientes a n horas, si la consulta de la jefa corresponde a un período en que realizó $k \geq 1$ tareas, el tiempo de ejecución de su algoritmo debe ser $\Theta(k \lg n)$. Note que ese valor k es desconocido, es lo que su algoritmo debe descubrir en el tiempo indicado.

Ejemplo: Si la secuencia de tareas de un trabajador es $\langle 1, 1, 1, 2, 3, 3, 3, 3, 3, 4, 5, 6, 6, 6, 6, 7, 7, 7, 8, 8, 8, 8, 9, 10\rangle$, y la jefa consulta por el período de horas $[7, 16]$, la respuesta son 5 tareas.

Formato de Entrada

Los datos serán leídos desde la entrada standard, en donde la primera línea contiene un único valor entero n ($1 \leq n \leq 10^9$) indicando la cantidad de horas registradas por el trabajador. La siguiente línea de la entrada contiene n valores enteros, separados entre sí por un único espacio. Dichos valores corresponden a las tareas desarrolladas por el trabajador en cada hora, tal como se indica en el enunciado más arriba. Luego, le sigue una línea que contiene un único valor entero m ($1 \leq m \leq 10^6$). A continuación le siguen m líneas, cada una

conteniendo un par de valores enteros $1 \leq i \leq j \leq n$, separados entre sí por un único espacio Un ejemplo particular de entrada es el siguiente:

```
24
1 1 1 2 3 3 3 3 3 4 5 6 6 6 6 7 7 7 8 8 8 8 9 10
4
7 16
1 24
6 9
18 23
```

Hint: para probar su programa de una mejor manera, ingrese los datos de entrada con el formato indicado en un archivo de texto (por ejemplo, el archivo `input-1.dat`). Luego, ejecute su programa desde la terminal, redirigiendo la entrada standard como a continuación:

```
./problema1 < input-1.dat
```

De esta manera evita tener que entrar los datos manualmente cada vez que prueba su programa, y evita errores.

Formato de Salida

La salida del programa debe mostrarse a través de la salida standard. La salida consiste de m líneas, cada una conteniendo un único entero correspondiente al intervalo de consulta dado en la entrada. La salida para el ejemplo anterior debería ser:

```
5
10
1
3
```

Entrega de la Tarea

La entrega de la tarea debe realizarse enviando un archivo comprimido llamado

`tarea3-apellido1-apellido2-apellido3.tar.gz`

(reemplazando sus apellidos según corresponda), o alternativamente usando formato zip, en el sitio Aula USM del curso, a más tardar el día 30 de junio, 2023, a las 23:59:00 hrs (Chile Continental), el cual contenga:

- Los archivos con el código fuente necesarios para el funcionamiento de la tarea.
- `NOMBRES.txt`, Nombre y ROL de cada integrante del grupo. También se debe indicar qué hizo cada integrante del grupo.
- `README.txt`, Instrucciones de compilación en caso de ser necesarias.
- `Makefile`, Instrucciones para compilación automática, en caso de ser necesarias.

El plazo máximo de entrega es de a lo más 5 horas desde la fecha original de entrega (30 de junio, 2023). Por cada hora (o fracción) de atraso se descontarán 20 puntos de la nota de la tarea.