Prueba de Cátedra 2

EII147 – Introducción a las Tecnologías de la Información
Profesores: Rafael Mellado (paralelo 1) – Katherine Valencia (paralelo 2)
II Semestre 2019 – Tiempo: 90 Minutos

ESCUELA DE INGENIERÍA INDUSTRIAL
PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

1. Instrucciones generales

1. Debe descargar el proyecto de Netbeans disponible en el aula virtual llamado SistemaControlMarchas.
2. Sólo puede trabajar con las clases y atributos definidos en el proyecto descargado. Cualquier cambio sobre el diseño de la estructura será evaluado con nota mínima.
3. Deberá entregar el mismo proyecto con el código desarrollado en las clases correspondientes.
4. Cualquier indicio de copia entre proyectos/desarrollos será evaluado con nota mínima. No puede ser desarrollado de forma grupal.
5. En el caso que los profesores lo estimen necesario o conveniente, la nota de esta evaluación podrá quedar sujeta a una interrogación oral de verificación de conocimientos.
6. Cuide el tiempo y la hora de entrega, ya que el único medio de entrega es aula virtual.

2. Enunciado

El gobierno ha decidido automatizar el proceso de control de personas que asisten a las marchas a nivel nacional. Para ello, define las siguientes principales clases:

El **País** tiene el nombre del presidente y una referencia a la lista de las regiones del país. La lista de regiones es un arreglo compacto manejado con pLibre.

Una **Región** tiene un número, nombre, nombre del gobernador y la lista de manifestantes. Los manifestantes son organizados a través de una matriz en niveles, en donde el primer nivel (fila cero) corresponde a aquellos encapuchados con mayor peligrosidad, la peligrosidad baja a medida que aumentan los niveles. La matriz no utiliza compactación ni pLibre.

Cada **Manifestante** tiene un rut, nombre, edad y un indice de peligrosidad, el cual es un valor real que tiene valores 0-1 e indica la probabilidad que el manifestante genere destrozos o daños a la policía.

Con lo señalado, se le pide: escribir una aplicación en Java junto a las clases que implementan las entidades descritas. Para ello, escriba su código en base a las siguientes preguntas:

1. `public boolean agregarManifestante (Manifestante nuevo, int numeroRegion)`

   Este método recibe como parámetro una referencia a un objeto tipo clase Manifestante a ser agregado y el número de la región a la cual se debe añadir. Una vez encontrada la región a la cual se debe agregar, validando que el manifestante no esté repetido (por rut) y si queda espacio en la colección de manifestantes, el nuevo objeto deberá ser agregado en el nivel correspondiente, para ello deberá considerar:

   - Si su peligrosidad total es mayor al **promedio del índice de peligrosidad de todos los manifestantes** de la primera fila entonces se debe agregar en la primera fila.
   - Si no se cumple la condición anterior, entonces deberá ser agregado en la primera fila que tenga espacio disponible desde la segunda en adelante.

   Además, debe implementar el cálculo de la peligrosidad total de cada manifestante de la forma:

   1. Si el manifestante cuenta con un índice de peligrosidad: la peligrosidad total del manifestante es igual al índice de peligrosidad * 0,2.
   2. Si el manifestante no cuenta con un índice de peligrosidad: la peligrosidad total es la edad * 0,08.

   Finalmente, deberá retornar true en caso de éxito y false en caso de fracaso.

2. **`public Manifestante[] manifestantesPeligrosos (int numeroRegion, double indicePeligrosidad)`**

   El gobernador de la región le indica por parámetro el número de región y el índice de peligrosidad a comparar, para ello ha dado la orden de detener de inmediato a aquellos manifestantes pertenecientes a la región y que su índice de peligrosidad es igual al indicado por parámetro. Para lo anterior, deberá quitar todos los manifestantes que cumplan con la condición anterior y retornarlos en un arreglo de tamaño exacto y compacto con las referencias a las instancias que se quitaron. En el caso de no existir región o manifestantes que cumplan con las condiciones antes descritas deberá retornar null.

3. Consideraciones

1. Recuerde ser ordenado en el desarrollo, indicando claramente la pregunta que está respondiendo.
2. No debe hacer uso de valores literales dentro de los problemas resueltos a menos que se pida explícitamente.
3. Respete las convenciones y nomenclatura vistas en el curso/presentes en el material del curso.
4. Respete el encapsulamiento y las buenas prácticas.
5. Comente su código para comprender mejor su resolución.
6. Debe implementar aquellos get y set que va a utilizar.
7. El main no puede ser modificado.
8. No puede agregar/eliminar clases y tampoco modificar los atributos y constructores ya definidos.