Tarea 2.

Profesor. Alejandro Hernández Mora
Ayudante. Pablo Camacho González
Estructuras de Datos 2019-1
Fecha de entrega : 13 de septiembre del 2018

30 de agosto de 2018

"Pensar es olvidar diferencias, es generalizar, abstraer."
. *- Jorge Luis Borges*

Listas

Ejercicio 1. Modele el siguiente problema usando Listas circulares

Según cuenta la historia que un judío J, junto con cuarenta soldados camaradas fueron capturados por los romanos después de la caída de Jotapata. Antes que rendirse, decidieron acabar ellos mismos con sus vidas. Para hacerlo, se dispusieron en un círculo y acordaron que irían contando de tres en tres, de forma que cada tercer soldado sería ejecutado por la persona de su izquierda. El último hombre que quedara con vida tendría que suicidarse. Según cuenta la leyenda, el judío J calculó rápidamente cuál sería la posición del último hombre en morir para colocarse allí, y una vez hubieron muerto sus compatriotas, se entregó a los romanos.

Problema a resolver (Versión 1)

Se tienen n personas numeradas entorno a un círculo esperando ser ejecutadas. Empezando por la persona número 1, se saltan $m-1$ personas y se mata a la m-ésima. A continuación se saltan otras $m-1$ personas y se ejecuta a la siguiente, y así hasta que sólo quede una. El objetivo es encontrar el lugar inicial en el círculo para sobrevivir $J(n,m)$,dados n y m

Modela tu programa usando listas circulares, de tal manera que cuando se inicie el programa se pregunte al usuario el número de personas que hay en el círculo y el número de lugares que se tiene que saltar, para ejecutar a un persona.

Problema a resolver (Versión 2)

Para la segunda versión, el número de lugares que se tienen que saltar para ejecutar a un persona será de manera aleatoria, es decir que cada vez que se ejecuta una persona el número de lugares que se tendrán que saltar para la próxima ejecución será aleatoria.

De igual manera que en la primera versión, se necesita el número de personas que hay en el círculo.

Requerimientos

- Debes de utilizar listas circulares, se proporciona el jar con la implementación de una lista circular.
- Debes de mostrar cómo se ejecuta el programa, es decir, como se van ejecutando las personas en el círculo. A lo largo del programa deberás guardar las ejecuciones en una lista simple y al final muestra el contenido, además deberás indicar el lugar en donde estaba la persona sobreviviente. Para la segunda versión, después de ejecutar a una persona tienes que mostrar el nuevo valor para el número de saltos,que será calculado aleatoriamente.
- Crea un menú para poder decidir cuál versión del problema ejecutar.
- El sistema debe de ser robusto (resitente a fallos, es decir que se ejecute sin arrojar excepciones de ningún tipo).
- El sistema nunca debe de cerrarse, solo cuando el usuario quiera dejar de hacer experimentos.
- Cada operación que puede ser realizada por el sistema debe estar encapsulada en un método.

Ejemplo de ejecución:

```
Numero de persona en el circulo:
10
Numero de pasos a dar para ejecutar a una persona:
2
--------------------
En la ejecucion 1 murio la persona en el lugar 2
--------------------
En la ejecucion 2 murio la persona en el lugar 4
--------------------
En la ejecucion 3 murio la persona en el lugar 6
--------------------
En la ejecucion 4 murio la persona en el lugar 8
--------------------
En la ejecucion 5 murio la persona en el lugar 10
--------------------
En la ejecucion 6 murio la persona en el lugar 3
--------------------
En la ejecucion 7 murio la persona en el lugar 7
--------------------
En la ejecucion 8 murio la persona en el lugar 1
--------------------
En la ejecucion 9 murio la persona en el lugar 7
--------------------
La persona en el lugar 5 es la que sobrevivio
--------------------
Permutaciones resultantes:
(1,2), (2,4), (3,6), (4,8), (5,10), (6,3), (7,7), (8,1), (9,7)


Para sobrevivir se necesita esta en el lugar 5
```

Punto extra

1. Crea nombres de manera aleatoria para las personas en el círculo y muestra su nombre en la ejecución, no se vale repetir nombre.

Información importante

Métodos que contiene la clase ListaCircular:

```
/**
 * Constructor que no recibe parámetros.
 * Inicializa a inicio con null
 * Inicializa nElementos con cero
 */
public ListaCircular()

 /**
 * Inserta el primer elemento de la lista
 * @param obj - Elemento que será insertado en la lista
 */
public void insertar(Object obj)

 /**
 * Devuelve el tamaño de la lista
 * @return int - valor del tamaño de la lista
 */
public int getSize()

/**
 * Metodo que determina si la lista esta o no vacia
 * @return boolean true si la lista está vacía y false en otro caso
 */
public boolean estaVacia()


/**
 * Método que devuelve un Iterador con todos los elementos de la Lista.
 * @return Iterador con todos los elementos de la Lista.
 */
public java.util.Iterator elementos(int inicio)

 /**
  * Método que determina si el Iterador tiene o no elementos que aún falten por iterar
  * @return true - Si todavía hay elementos por iterar, -faslse, en otro caso.
  */

 @Override
 public boolean hasNext()

 /**
  * Método que obtiene el siguiente elemento en nuestro Iterador
  * @return Object - El siguiente elemento en el iterador
  */

 @Override
 public Object next()
```

```
/**
 * Método para eliminar un elemento de Iterador
 */

@Override
public void remove()
```

Entrega de Tarea

- La entrega de la tarea se hace bajo los mismo lineamientos que las prácticas.
- Envía la tarea al correo **pablopcg1@ciencias.unam.mx** con el asunto [**EDD-TAREA02**]
- Se tiene como límite las **23:59:59** de la fecha de entrega, cada día de retraso se penaliza con un punto menos, después de tres días de retraso la tarea ya no se califica.