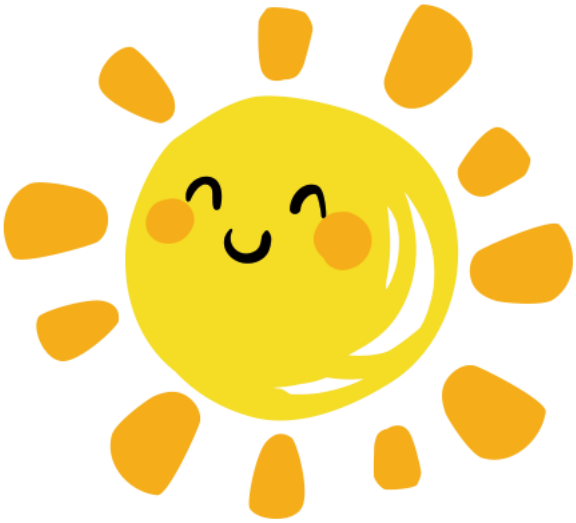

TAREA 5

Estructuras de Datos Concurrentes

Profesor: Salvador Gonzalez Arellano.
email: salvador_gonzalez_a@ciencias.unam.mx

Ayudante de teoría: Rogelio Alcantar Arenas.
email: rogelio-aa@ciencias.unam.mx

Ayudante de laboratorio: Kassandra
email: luis_angel_howke@ciencias.unam.mx

FACULTAD DE CIENCIAS, UNAM
Computación Concurrente

Introducción

Zeratun estaba muy contento pues ya pudo recuperar su tanque de gas y sus macetas, pero se dio cuenta que le roban mucho, por lo que decidio aprender computación concurrente para que no le pasara de nuevo y asi estar prevenido, ayudemos a zeratun a aprender computo concurrente y pueda por fin descansar.

TEORÍA

1. ¿Para que sirve el método Yield? (yield())
2. ¿Qué es un atributo átomico? (En la biblioteca atomic de Java)
3. Ventajas de usar atributos atomicos
4. Desventajas de usar atributos atómicos
5. Da 2 ejemplos en donde se puedan aplicar este tipo de atributos (no pongan en resolver la tarea plox)
6. Algun uso que creas que se da en estructuras de datos concurrentes.

Ademas a eso, quire ver como se comportan los hilos, por lo que te pide que le ayudes escribiendo las historias de ejecución del siguiente bloque de código:

```
public class Counter implements Runnable {
    public static final int ROUNDS = 5;
    private int counter = 0;

    @Override
    public void run() {
        for(int i=0; i< Counter.ROUNDS; i++) {
            counter++;
        }
    }
}
```

Muestra historias donde se puedan obtener lo siguientes valores con la cantidad de hilos 2,4 y 5 hilos, en caso de que no se pueda, explica el porque no:

- 0
- 2

- 3
- 5
- 10
- 13
- 15
- 20
- 26

Ademas, Zeratun le marco a su amix el Poncho para que le ayudara en la parte de estructuras de datos Concurrentes, este tiene los pseudocódigos pero se le olvido documentarlos (y en algunos caso implementarlos), ayudales a documentar e implementar el codigo explicando que hace cada linea.

- TAS
- TTAS
- BackoffLock

```
1  public class Backoff {
2    final int minDelay, maxDelay;
3    int limit;
4    final Random random;
5    public Backoff(int min, int max) {
6      minDelay = min;
7      maxDelay = max;
8      limit = minDelay;
9      random = new Random();
10   }
11   public void backoff() throws InterruptedException {
12     int delay = random.nextInt(limit);
13     limit = Math.min(maxDelay, 2 * limit);
14     Thread.sleep(delay);
15   }
16 }
```

```
1  public class BackoffLock implements Lock {
2    private AtomicBoolean state = new AtomicBoolean(false);
3    private static final int MIN_DELAY = ...;
4    private static final int MAX_DELAY = ...;
5    public void lock() {
6      Backoff backoff = new Backoff(MIN_DELAY, MAX_DELAY);
7      while (true) {
8        while (state.get()) {};
9        if (!state.getAndSet(true)) {
10         return;
11       } else {
12         backoff.backoff();
13       }
14     }
15   }
16   public void unlock() {
17     state.set(false);
18   }
19   ...
20 }
```

- CLHLock

```
1  public class CLHLock implements Lock {
2    AtomicReference<Qnode> tail;
3    ThreadLocal<QNode> myPred;
4    ThreadLocal<QNode> myNode;
5    public CLHLock() {
6    tail = new AtomicReference<QNode>(null);
7      myNode = new ThreadLocal<QNode>() {
8        protected QNode initialValue() {
9          return new QNode();
10       }
11     };
12     myPred = new ThreadLocal<QNode>() {
13       protected QNode initialValue() {
14         return null;
15       }
16     };
17   }
18   ...
19 }
```

```
20    public void lock() {
21      QNode qnode = myNode.get();
22      qnode.locked = true;
23      QNode pred = tail.getAndSet(qnode);
24      myPred.set(pred);
25      while (pred.locked) {}
26    }
27    public void unlock() {
28      QNode qnode = myNode.get();
29      qnode.locked = false;
30      myNode.set(myPred.get());
31    }
32  }
```

Qnode tiene por unico elemento un booleano volatil, llamado looked.

- MCSLock

Una vez documentado el codigo, realiza pruebas de tiempo de ejecución usando distintos número de Hilos (2,3,7,15,21,30,50), de ahí realiza una tabla como la siguiente:

Hilos	TAS	TTAS	BackOff	CLH
3	6.68	6.44	5.12	8.24
7	12.67	11.28	6.01	193.08

Posteriormente realiza una grafica para mostrar visualmente la comparación de estos.

***NOTA: ESTA PARTE LA TIENEN QUE HACER TODOS LOS INTEGRANTES DEL EQUIPO, ES DECIR, CADA INTEGRANTE TIENE QUE TOMAR LOS CODIGOS Y EJECUTARLOS, HACER SU TABLA Y POSTERIORMENTE SU GRÁFICA ASI MISMO, DEBEN DE AGREGAR LAS ESPECIFICACIONES DE SU COMPUTADORA.**

Una vez hecho esto, explica el porque se obtienen esos tiempos, compara el resultado con el de tus compañeros, y si la diferencia es muy grande explica el porque

EXTRA

Ayudales escribiendo el mismo codigo pero sin usar operaciones atomicas (solo volatile), escribe el porque “Falla”, haz una pequeña historia de ejecución donde no falle, en todo caso, dejalos correr entre 2 a 10 minutos.

Implementa el Alock

```
1  public class ALock implements Lock {
2    ThreadLocal<Integer> mySlotIndex = new ThreadLocal<Integer> (){
3      protected Integer initialValue() {
4        return 0;
5      }
6    };
7    AtomicInteger tail;
8    volatile boolean[] flag;
9    int size;
10   public ALock(int capacity) {
11     size = capacity;
12     tail = new AtomicInteger(0);
13     flag = new boolean[capacity];
14     flag[0] = true;
15   }
16   public void lock() {
17     int slot = tail.getAndIncrement() % size;
18     mySlotIndex.set(slot);
19     while (! flag[slot]) {};
20   }
21   public void unlock() {
22     int slot = mySlotIndex.get();
23     flag[slot] = false;
24     flag[(slot + 1) % size] = true;
25   }
26 }
```

ENTREGABLE

Zeratún que las respuestas se hagan a computadora, en el editor de su elección. Deben de poner las referencias bibliográficas en donde consultaron la información, esta debe de ir en formato APA y en formato PDF, si no se llevaran los dados de su carro.

Se les dará 0.5 extra si la realizan en LaTeX.

NOTA: SE PUEDE LLEGAR A CALENTAR BASTANTE SU COMPUTADORA CON LAS PRUEBAS EJERCIDAS.

Debe de llevar el siguiente formato:

[NOMBRE DEL EQUIPO.[pdf]

EJEMPLO:

SPECIALOPERATIONSCONCURRENT.pdf