FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES

CARRERA: COMPUTACIÓN		**ASIGNATURA**: Programación Aplicada
NRO. PRÁCTICA:	5	**TÍTULO PRÁCTICA**: Hilos en Java

OBJETIVO:
Identificar los cambios importantes de Java
Diseñar e Implementar las nuevas técnicas de programación concurrente
Entender cada una de las características de Thread en Java.

INSTRUCCIONES (Detallar las instrucciones que se dará al estudiante):	**1.** Revisar los conceptos fundamentales de Thread en Java
	2. Establecer como implementar Thread en Java
	3. Implementar y diseñar los nuevos componentes de concurrencia
	4. Realizar el informe respectivo según los datos solicitados.

ACTIVIDADES POR DESARROLLAR

(Anotar las actividades que deberá seguir el estudiante para el cumplimiento de la práctica)

1. Revisar la teoría y conceptos de Teread en Java

En Java la programación concurrente está orientada a través de procesos o tareas, denominados Thread, que en español significa Hilos.

Los Thread (hilos) son procesos ligeros, con línea de flujo de control (métodos) propios, pero que comparte los recursos de la computadora para su ejecución en paralelo o de forma sincronizada, en conclusión, son instrucciones que permiten ejecutar 2 o más acciones (procesos) al mismo tiempo.

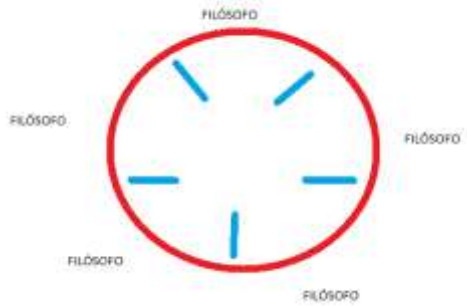
2. Diseñar e implementar las características de Java para generar una simulación 2D del siguiente enunciado:

Problema del Filósofo:

En una mesa hay procesos que simulan el comportamiento de unos filósofos que intentan comer de un plato. Cada filósofo tiene un cubierto a su izquierda y uno a su derecha y para poder comer tiene que conseguir los dos. Si lo consigue, mostrará un mensaje en pantalla que indique «Filosofo 2 (numero) comiendo».
Después de comer, soltará los cubiertos y esperará al azar un tiempo entre 1000 y 5000 milisegundos, indicando por pantalla «El filósofo 2 está pensando».

En general todos los objetos de la clase Filósofo están en un bucle infinito dedicándose a comer y a pensar.

Simular este problema en un programa Java que muestre el progreso de todos sin caer en problemas de sincronización a través de un método gráfico.



- Para esta practica se creo un nuevo proyecto en NetBeans con el nombre de ProblemaDelFilosofo.
- Dentro de esta se crearon tres paquetes:

  modelo: en el cual se implementaron tres clases (Filosofo, Tenedor y Comedor)
  vista: se desarrolló la ventana principal del programa
  imágenes: se guardaros todas las imágenes que mejorarán la interfaz gráfica.

- Comedor:

```
 */
public class Comedor {
    private int comensalesDisponibles = 4;

    public synchronized void alistarse() throws InterruptedException{
        while(comensalesDisponibles==0){
            this.wait();
        }
        comensalesDisponibles--;
    }

    public synchronized void desalistarse() throws InterruptedException{
        comensalesDisponibles++;
        this.notify();
    }
}
```

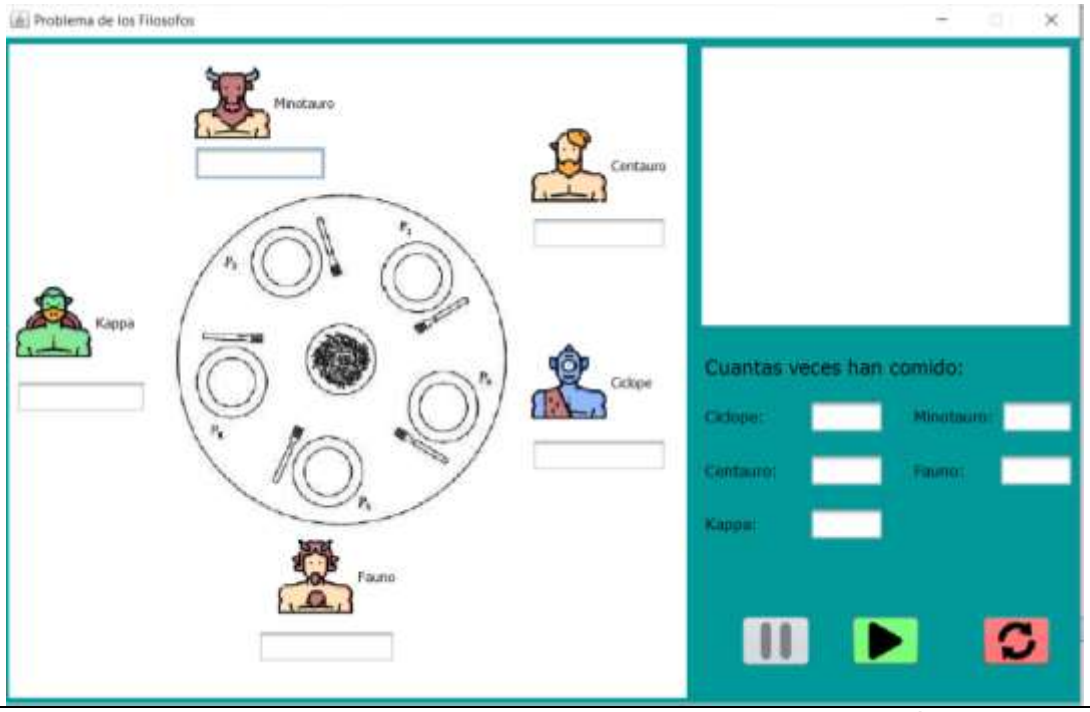
- Filosofo (Método run):

```
@Override
public void run() {
    while (true) {

        try {
            comedor.alistarse();
            this.tenDerecha.cogerTenedor();

            if (!tenIzquieda.cogerTenedorIzqdo()) {

                tenDerecha.soltarTenedor();
                comedor.desalistarse();

                try {
                    Filosofo.sleep(random.nextInt(1000) + 100);
                } catch (InterruptedException ex) {
                    System.out.println("Ha ocurrido un error: " + ex.toString());
                }
                continue;
            }

            this.jTextField_Estado.setText("Comiendo.....");
            this.jTextArea_Mensajes.setText(this.jTextArea_Mensajes.getText() + "\n" + nombre + ":  Esta comiendo...");
            this.contador++;
            this.jTextField_Contador.setText(contador + "");

            try {
                sleep(random.nextInt(1000) + 500);
            } catch (InterruptedException ex) {
                System.out.println("Ha ocurrido un error:" + ex.toString());
            }

            tenDerecha.soltarTenedor();
            tenIzquieda.soltarTenedor();
            comedor.desalistarse();

            this.jTextField_Estado.setText("Pensando.....");
            this.jTextArea_Mensajes.setText(this.jTextArea_Mensajes.getText() + "\n" + nombre + ":  Esta Pensando...");

            try {
                Filosofo.sleep(random.nextInt(4000) + 1000);
            } catch (InterruptedException ex) {
                System.out.println("Ha ocurrido un error: " + ex.toString());
            }

        } catch (InterruptedException ex) {
            ex.printStackTrace();
        }

        if (finalizado) {
            this.jTextField_Estado.setText("Pensando.....");
            this.jTextArea_Mensajes.setText(this.jTextArea_Mensajes.getText() + "\n" + nombre + ":  Esta Pensando...");
            break;
        }
    }
```

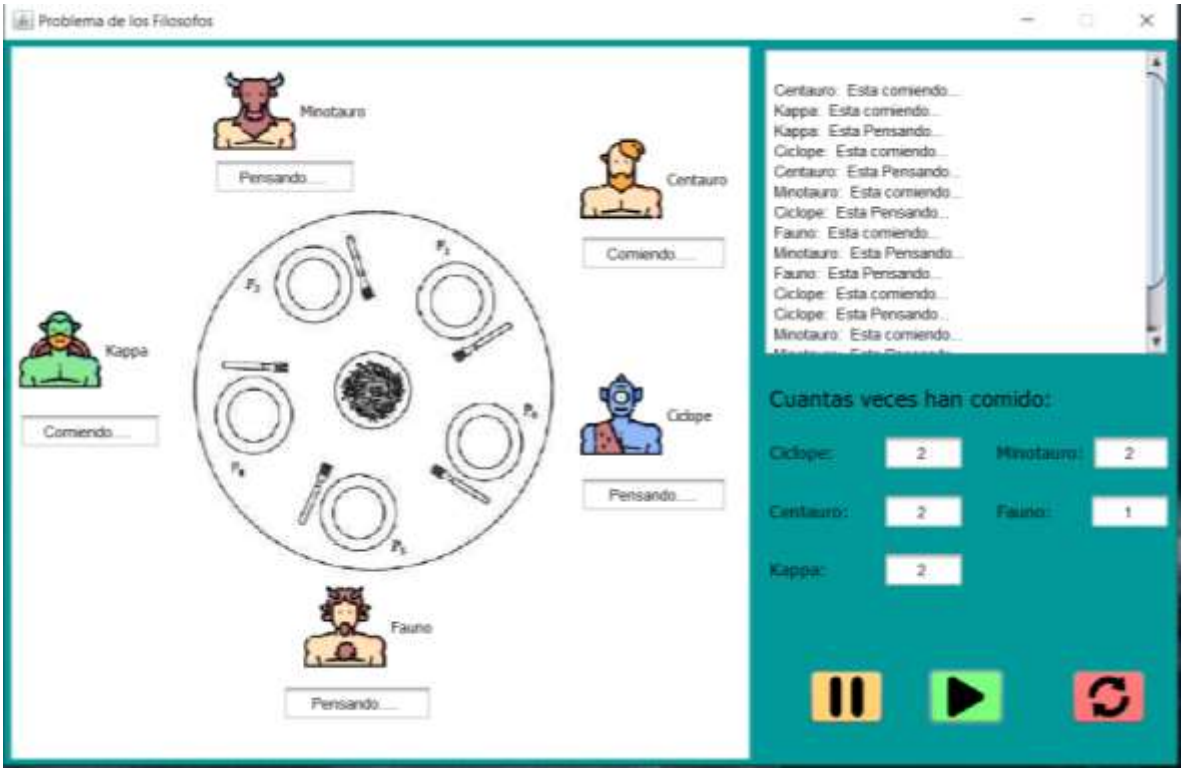
- Tenedor:

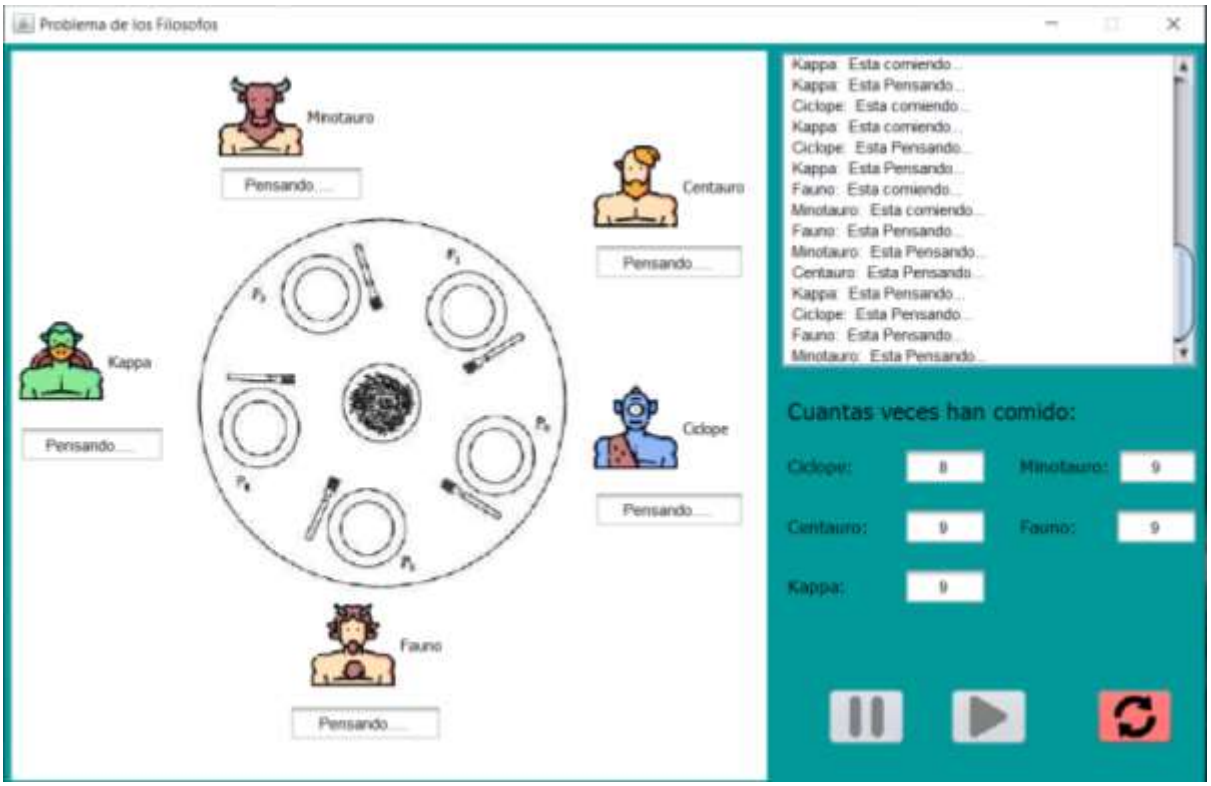
```
public class Tenedor {
    private Random random = new Random();
    private int id;
    private boolean libre;

    public Tenedor(int id){
        this.id = id;
        this.libre = true;
    }

    public synchronized void cogerTenedor() throws InterruptedException{
        while(!libre)
            this.wait();
        libre = false;
    }
    public synchronized boolean cogerTenedorIzqdo() throws InterruptedException{
        while(!libre){
            this.wait(random.nextInt(1000) + 500);
            return false;
        }
        libre = false;
        return true;
    }
    public synchronized void soltarTenedor() throws InterruptedException {
        libre = true;
        this.notify();
    }
}
```

- Luego se procedió con la creación de la interfaz gráfica:



- Funcionamiento:





- Repositorio del Proyecto:

  https://github.com/VazquezAriel/FilosofosComensales

RESULTADO(S) OBTENIDO(S):
Realizar procesos de Hilos en Java.
Entender las aplicaciones de codificación de las nuevas características de concurrencia.
Entender las funcionalidades de sincronización y manejo de grupo de Thread dentro de Java.

CONCLUSIONES:
Aprenden a trabajar en grupo dentro de plazos de tiempo establecidos, manejando el lenguaje de programación de Java.

RECOMENDACIONES:
Realizar el trabajo dentro del tiempo establecido.

***Nombre de estudiante*:** Ariel Vazquez

***Firma de estudiante*:**