
	Computación	Docente: Diego Quisi Peralta
	Programación Aplicada	Período Lectivo: Septiembre 2020 – Febrero 2021

		FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES	
CARRERA: COMPUTACIÓN/INGENIERÍA DE SISTEMAS		ASIGNATURA: PROGRAMACIÓN APLICADA	
NRO. PROYECTO:	1.1	TÍTULO PROYECTO: Prueba Practica 2 Desarrollo e implementación de un sistema de simulación de acceso y atención bancaria	
OBJETIVO: Reforzar los conocimientos adquiridos en clase sobre la programación en Hilos en un contexto real.			
INSTRUCCIONES:		1. Revisar el contenido teórico y práctico del tema	
		2. Profundizar los conocimientos revisando los libros guías, los enlaces contenidos en los objetos de aprendizaje Java y la documentación disponible en fuentes académicas en línea.	
		3. Deberá desarrollar un sistema informático para la simulación y una interfaz gráfica.	
		4. Deberá generar un informe de la práctica en formato PDF y en conjunto con el código se debe subir al GitHub personal y AVAC.	
		5. Fecha de entrega: El sistema debe ser subido al jit hasta 17 de enero del 2021 – 23:55.	
ACTIVIDADES POR DESARROLLAR			

1. Enunciado:

Realizar un sistema de simulación de acceso y atención a través de colas de un banco.

Problema: Un banco necesita controlar el acceso a cuentas bancarias y para ello desea hacer un programa de prueba en Java que permita lanzar procesos que ingresen y retiren dinero a la vez y comprobar así si el resultado final es el esperado.

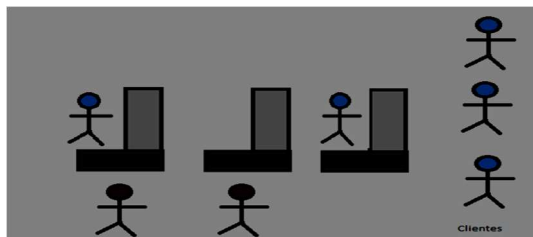
Se parte de una cuenta con 100 euros y se pueden tener procesos que ingresen 100 euros, 50 o 20. También se pueden tener procesos que retiren 100, 50 o 20 euros. Se desean tener los siguientes procesos:

- ⑩ 40 procesos que ingresan 100
- ⑩ 20 procesos que ingresan 50
- ⑩ 60 que ingresen 20.

De la misma manera se desean lo siguientes procesos que retiran cantidades.

- ⑩ 40 procesos que retiran 100
- ⑩ 20 procesos que retiran 50
- ⑩ 60 que retiran 20.


Además en el banco, existen 3 cajeros que pueden atender y hay una cola inicial de 10 clientes para ser atendidos, el proceso de atención es de 20 – 15 segundos y los clientes llegan constantemente cada 30 - 50 segundos. Ningún cajero puede atender simultáneamente, adicionalmente el tiempo de moverme de la cola al estante del cajero es de 2 - 5 segundos, esto deberán ser generados aleatoriamente entre los 100 clientes que disponen una cuenta, estos pueden volver a ingresar el número de veces que sea necesario.



Se desea comprobar que tras la ejecución la cuenta tiene exactamente 100 euros, que era la cantidad de la que se disponía al principio. Realizar el programa Java que demuestra dicho hecho.

Calificación:

- ⑩ Diagrama de Clase 10%
- ⑩ MVC: 10%
- ⑩ Técnicas de Programación aplicadas (Java 8, Reflexión y Programación Genérica): 10%

	Computación	Docente: Diego Quisi Peralta
	Programación Aplicada	Período Lectivo: Septiembre 2020 – Febrero 2021

- ⑩ Hilos 30%
- ⑩ Sincronización 10%
- ⑩ Interfaz Gráfica de simulación 20%
- ⑩ Informe: 10%

2. Informe de Actividades:

- Planteamiento y descripción del problema.
- Diagramas de Clases.
- Patrón de diseño aplicado
- Descripción de la solución y pasos seguidos.
 - ⑩ Comprobación de las cuentas bancarias e interfaz gráfica.
- Conclusiones y recomendaciones.
- Resultados.

RESULTADO(S) OBTENIDO(S):

- Interpreta de forma correcta los algoritmos de programación y su aplicabilidad.
- Identifica correctamente qué herramientas de programación se pueden aplicar.

CONCLUSIONES:

- Los estudiantes identifican las principales estructuras para la creación de sistemas informáticos.
- Los estudiantes implementan soluciones gráficas en sistemas.
- Los estudiantes están en la capacidad de implementar hilos.

RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la práctica.
- Haber asistido a las sesiones de clase.
- **Consultar con el docente las dudas que puedan surgir al momento de realizar la prueba.**

BIBLIOGRAFIA:

[1]: <https://www.ups.edu.ec/evento?calendarBookingId=98892>

Docente / Técnico Docente: Ing. Diego Quisi Peralta Msc.

Firma: _____



FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES

CARRERA: COMPUTACION

ASIGNATURA: Programación Aplicada

NRO. PRÁCTICA:

TÍTULO PRÁCTICA: Prueba Practica 2

Desarrollo e implementación de un sistema de simulación de acceso y atención bancaria

OBJETIVO ALCANZADO:

Reforzar los conocimientos adquiridos en clase sobre la programación en Hilos en un contexto real.

ACTIVIDADES DESARROLLADAS

1. Planteamiento y descripción del problema

Para resolver el problema se necesita conocer lo que es la práctica de hilos, y como crearlos, en este case se utilizara 2 hilos de diferentes clases, el uno para llenar lo que es la cola y el otro para ir utilizando, como productor consumidor.

2. Diagrama de clases.

Para realizar el diagrama de clases se utilizó la herramienta lucidchart.

Adjunto el link de el diagrama:

<https://lucid.app/lucidchart/invitations/accept/eccd78ae-6d70-45ba-a8e9-dbb238101dfc>.

3. Patrón de diseño aplicado

El patrón de diseño utilizado fue el singleton y una parte del flyweight para generar de forma random si se desea retirar o depositar y cuál sería la cantidad.

```
public class ControladroRandomProcesos {  
  
    private String[] accion = {"Ingresar", "Retirar"};  
    private int[] valor = {100, 50, 20};  
    private Random randomico = new Random();  
  
    public String generarAccion() {  
  
        int i = randomico.nextInt(accion.length);  
        return accion[i];  
    }  
  
    public int generarValor() {  
  
        int i = randomico.nextInt(valor.length);  
        return valor[i];  
    }  
  
}
```

EL patrón singleton se utilizó para tener una misma instancia de la cola y así no generar errores de null.

```
public class ControladorCola {

    private static ControladorCola instance = new ControladorCola();

    private int MAXIMO = 10;
    private List<Cliente> lista;
    LinkedList<Cliente> cola;

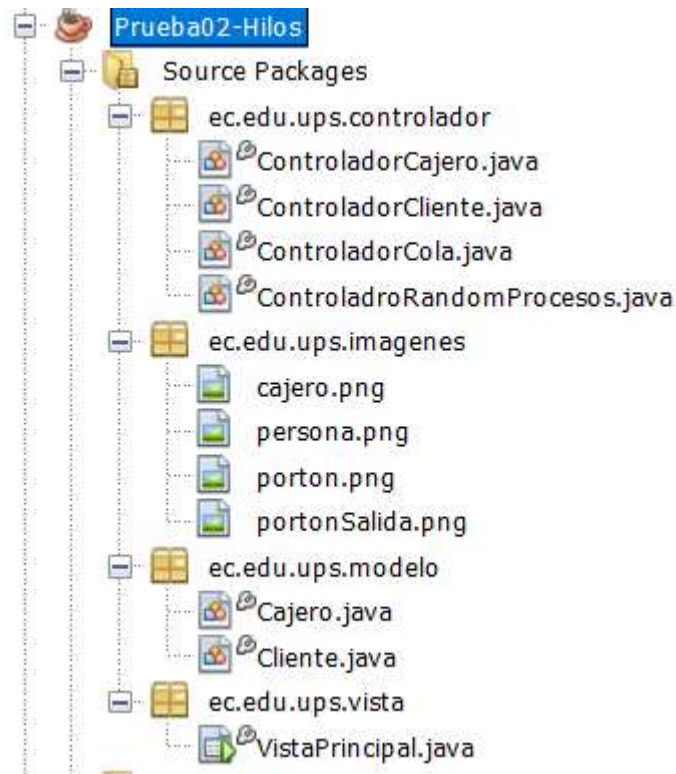
    public static ControladorCola getInstance(){

        return instance;

    }
}
```

4. Descripción de la solución y pasos seguidos.

Primero se creó 2 clases modelos la una de cliente que sería la productora y los cajeros que serían los consumidores.



Además se utilizó la arquitectura MVC modelo, vista, controlador.
En los controladores se implementó la clase runnable.

```
public class ControladorCliente implements Runnable{

public class ControladorCajero implements Runnable {
```

Para poner en cola los hilos se utilizó la clase LinkedList que es una clase propia de java para esos usos. y en su código se genero las 4 formas de control de las colas.

```
public boolean encolar(Cliente cliente){
    if(colaLlena()){
        Cliente c = clienteRemplazado(cliente);
        cola.addLast(c);
        return true;
    }
    return false;
}

public Cliente desencolar(){
    if(!colaVacia()){
        return cola.removeFirst();
    }
    return null;
}

public boolean colaLlena(){
    if(cola.size() <= MAXIMO-1){
        return true;
    }
    return false;
}

public boolean colaVacia(){
    if(cola.size()==0){
        return true;
    }
    return false;
}
```

En la clase Controlador Cliente se implemento el método run de la siguiente forma:

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

        synchronized(cola){

            if(cliente.getContador()!=1){
                System.out.println("Cuenta llena");
            }else{
                if(!cola.comprobarCola(cliente)){
                    esperar();

                    try {
                        llegarBanco();
                    } catch (InterruptedException ex) {
                        Logger.getLogger(ControladorCliente.class.getName()).log(Level.SEVERE, null, ex);
                    }
                }
            }
        }
    }
}
```

En el cual el método llegar al banco es el método para cargar los clientes en la cola.

En la clase Controlador Cajero se implementó el método run de la siguiente forma:

```
@Override
public void run() {

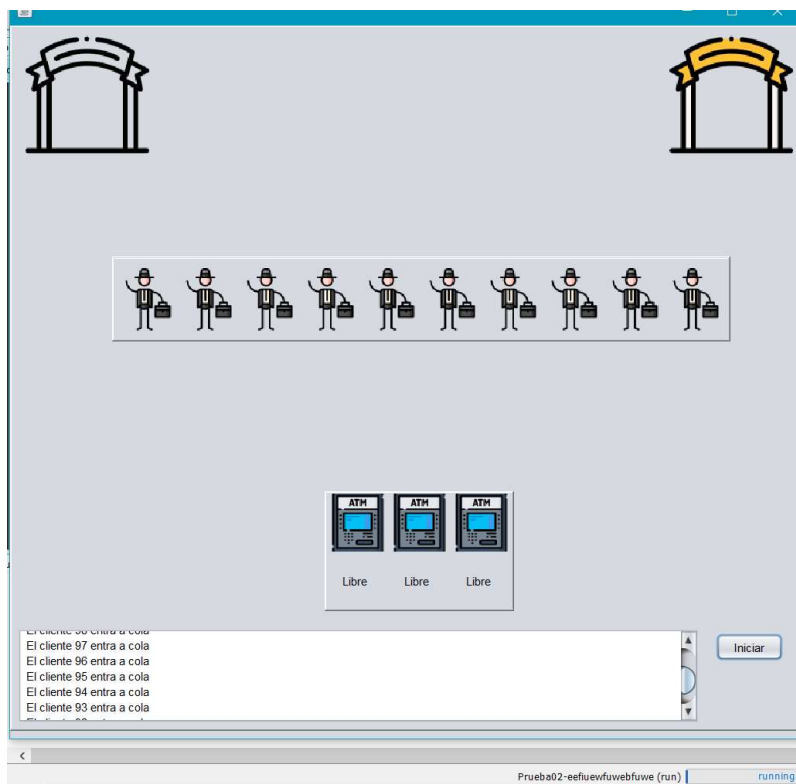
    while (centinela) {

        while(cent){
            System.out.println(cola.colaLlena());
            if(cola.colaLlena()==false){
                cent=false;
                break;
            }
        }
        cobrar();
    }
}
```

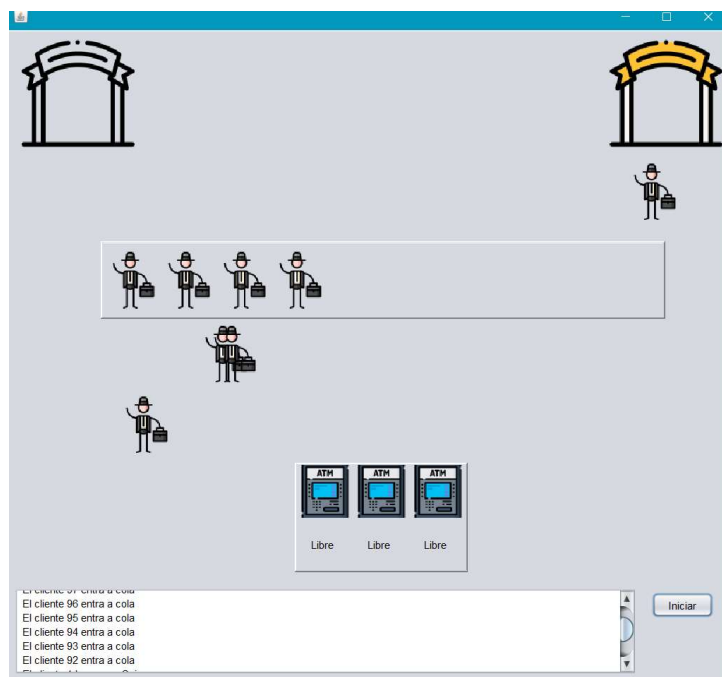
En el cual se espera a que la cola se llene, ya que el enunciado pide que se comienza con una cola de 10 personas o clientes.


Y el método cobrar es en el cual se controla la cuenta y los valores que son generados de forma random.

Pantalla de los 10 procesos en cola:

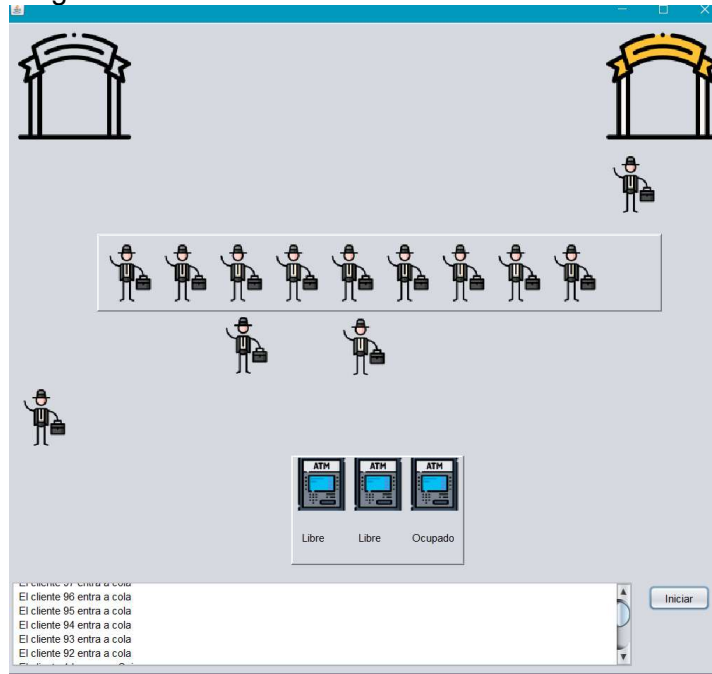


En la siguiente imagen se puede apreciar que los clientes que estan en la cola se mueven hacia los cajeros mientras que llega un nuevo cliente.

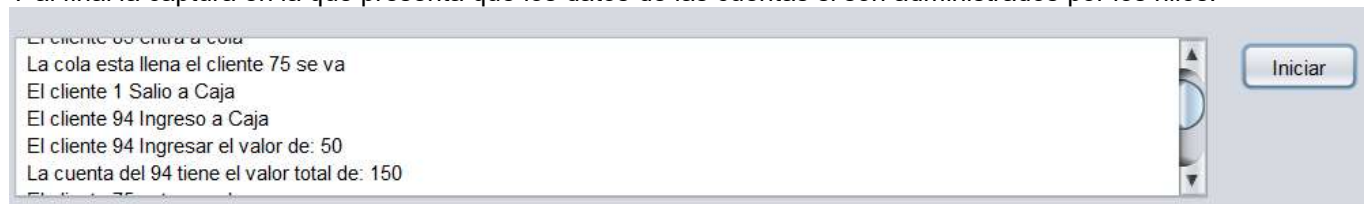


	Computación	Docente: Diego Quisi Peralta
	Programación Aplicada	Período Lectivo: Septiembre 2020 – Febrero 2021

En la siguiente imagen se puede apreciar que un cliente se termina mientras que dos se dirigen hacia el cajero y un nuevo cliente llega



Y al final la captura en la que presenta que los datos de las cuentas si son administrados por los hilos.



RESULTADO(S) OBTENIDO(S):

- Interpreta de forma correcta los algoritmos de programación y su aplicabilidad.
- Identifica correctamente qué herramientas de programación se pueden aplicar.

CONCLUSIONES:

- Los estudiantes identifican las principales estructuras para la creación de sistemas informáticos.
- Los estudiantes implementan soluciones gráficas en sistemas.
- Los estudiantes están en la capacidad de implementar hilos.

RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la práctica.
- Haber asistido a las sesiones de clase.
- Consultar con el docente las dudas que puedan surgir al momento de realizar la prueba.

Nombre de estudiante: Sebastián Uyaguari

Firma de estudiante:

