
	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 practica 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 git 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.

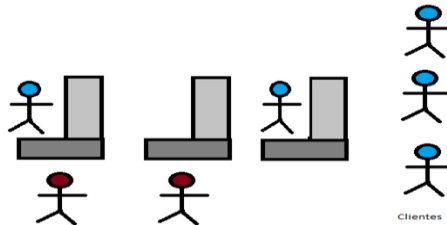
Se parte de una cuenta con 100 euros y se pueden tener procesos que ingresen 100 euros, 50 o 20. También se pueden tener procesos que retiran 100, 50 o 20 euros. Se desean tener los siguientes procesos:

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

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

- 40 procesos que retiran 100
- 20 procesos que retiran 50
- 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:

- Diagrama de Clase 10%
- MVC: 10%
- Técnicas de Programación aplicadas (Java 8, Reflexión y Programación Genérica): 10%
- Hilos 30%
- Sincronización 10%
- Interfaz Gráfica de simulación 20%
- Informe: 10%

	Computación	Docente: Diego Quisi Peralta
	Programacion Aplicada	Período Lectivo: Septiembre 2020 – Febero 2021

2. Informe de Actividades:

- Planteamiento y descripción del problema.
- Diagramas de Clases.
- Patrón de diseño aplicado
- Descripción de la solución y pasos seguidos.
- 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 creacion de sistemas informaticos.
- Los estudiantes implementan soluciones graficas 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: _____

CARRERA: INGENIERIA DE SISTEMAS

ASIGNATURA: PROGRAMACION APLICADA

NRO. PRÁCTICA:

2

TÍTULO PRÁCTICA: PRUEBA 2 - HILOS

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.

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.

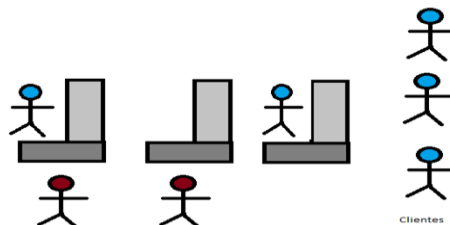
Se parte de una cuenta con 100 euros y se pueden tener procesos que ingresen 100 euros, 50 o 20. También se pueden tener procesos que retiran 100, 50 o 20 euros. Se desean tener los siguientes procesos:

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

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

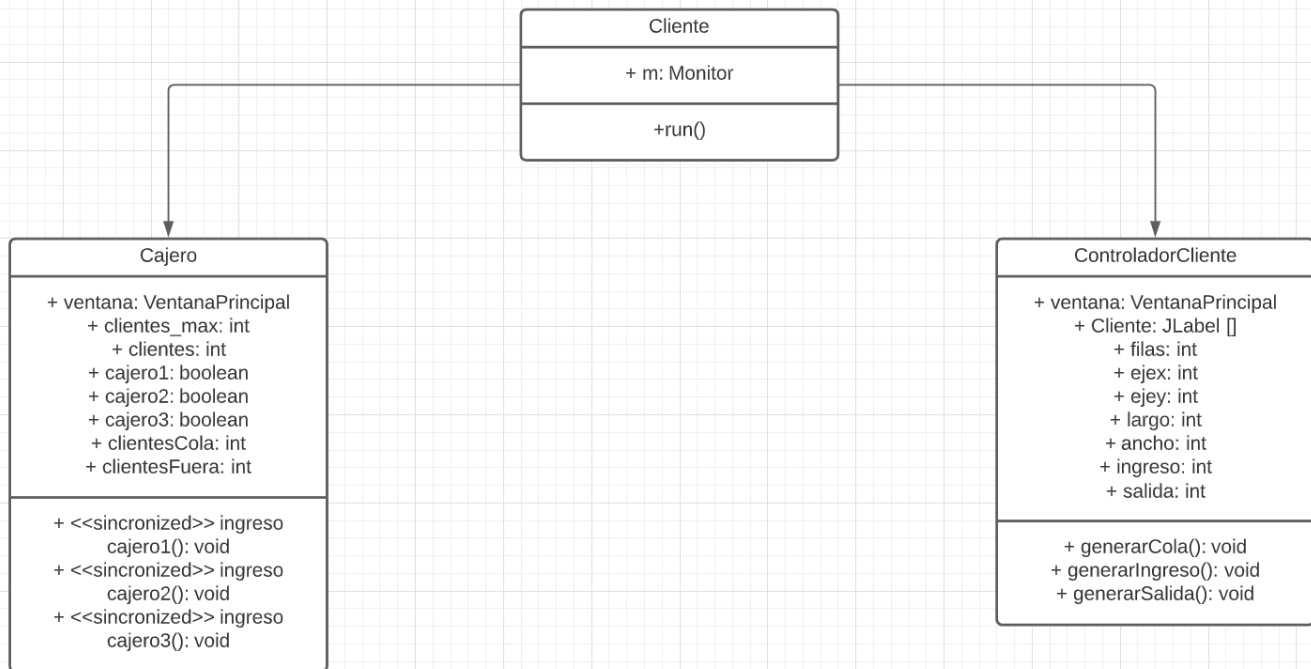
- 40 procesos que retiran 100
- 20 procesos que retiran 50
- 60 que retiran 20.

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.



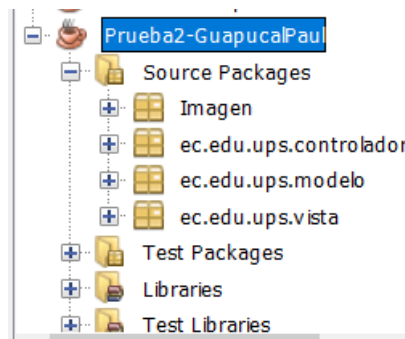
Se desea comprobar que tras la ejecución la cuenta tiene exactamente 100 euros, que era la cantidad de la que se disponía al principio. Realizar el programa Java que demuestra dicho hecho.

2. Diagrama de clase.



3. Patrón de diseño aplicado.

En este caso, se utilizó el patrón de diseño MVC.



4. Descripción de la solución y pasos seguidos.

Para realizar este proyecto se utilizó el siguiente proceso.

- 1- Primero se tiene la creación de paquetes con sus respectivas clases y métodos dentro de ellas.
- 2- Segundo tenemos la implementación de todas estas clases y métodos dentro de programa.
 - Ec.edu.ups.controlador
 - ControladorCliente

```

package ec.edu.ups.controlador;

import ec.edu.ups.vista.VentanaPrincipal;
import ec.edu.ups.modelo.Cajero;
import javax.swing.JLabel;

/**
 *
 * @author paul_
 */
public class ControladorCliente {

    VentanaPrincipal ventana;
    public JLabel [] Clientes ;
    int filas;
    int ejex=10;
    
```

```

int ejey=10;
int largo=60;
int ancho=60;
int ingreso;
int fuera;
public ControladorCliente(VentanaPrincipal ventana, Cajero m) {
    filas= m.getClientesCola();
    this.ventana=ventana;

    ingreso= m.getClientes();
    fuera=m.getClientesFuera();
}

public void generarColaDefault(){

    if(filas>0){
        ventana.getPanelCola().removeAll();
        Clientes= new JLabel [filas];
        for(int i=0;i<filas;i++){

            Clientes[i] =new javax.swing.JLabel();
            Clientes[i].setIcon(new javax.swing.ImageIcon(
Icon(getClass().getResource("/Imagen/iconousuario.png"))));
            Clientes [i].setBounds(ejex, ejey, largo, ancho);
            ventana.getPanelCola().add(Clientes[i]);
            ejex+=50;

        }
    }
}

public void generarIngreso(){
    if(ingreso>0){
        ventana.getPanelIngreso().removeAll();
        Clientes= new JLabel [ingreso];
        for(int i=0;i<ingreso;i++){

            Clientes[i] =new javax.swing.JLabel();
            Clientes[i].setIcon(new javax.swing.ImageIcon(
Icon(getClass().getResource("/Imagen/iconousuario.png"))));
            Clientes [i].setBounds(ejex, ejey, largo, ancho);
            ventana.getPanelIngreso().add(Clientes[i]);
            ejex+=50;

        }
    }
}

public void generarSalida(){

    if (fuera>0){
        ventana.getPanelSalida().removeAll();
        Clientes= new JLabel [fuera];
        for(int i=0;i<filas;i++){

            Clientes[i] =new javax.swing.JLabel();
            Clientes[i].setIcon(new javax.swing.ImageIcon(
Icon(getClass().getResource("/Imagen/iconousuario.png"))));
            Clientes [i].setBounds(ejex, ejey, largo, ancho);
            ventana.getPanelCola().add(Clientes[i]);
            ejex+=50;

```

```

    }
}

}

• Ec.edu.ups.modelo
  ➤ Cliente
package ec.edu.ups.modelo;

import ec.edu.ups.vista.VentanaPrincipal;
import java.util.logging.Level;
import java.util.logging.Logger;
import ec.edu.ups.modelo.Cajero;

/**
 *
 * @author paul_
 */
public class Cliente implements Runnable{

    Cajero m;

    public Cliente( Cajero m) {

        this.m=m;
        m.clientes++;

    }

    public Cliente() {

    }

    @Override
    public void run() {
        while(true){
            if(Thread.currentThread().isAlive()){
                if(!m.cajero1){
                    m.ingresoCajero1();
                    break;
                }else if(!m.cajero2){
                    m.ingresoCajero2();
                    break;
                }else if(!m.cajero3){
                    m.ingresoCajero3();
                    break;
                }
            }
        }
    }
}

➤ Cajero
package ec.edu.ups.modelo;

import ec.edu.ups.controlador.ControladorCliente;
import ec.edu.ups.vista.VentanaPrincipal;

```

```

import javax.swing.JLabel;

/**
 *
 * @author paul_
 */
public class Cajero implements Runnable {

    VentanaPrincipal ventana ;
    int clientes_max;
    int clientes;
    boolean cajero1;
    boolean cajero2;
    boolean cajero3;
    int clientesCola;
    int clientesFuera;

    public Cajero(VentanaPrincipal ventana) {

        cajero1= false;
        cajero2= false;
        cajero3=false;
        clientes=0;
        clientesCola=10;
        clientes_max=90;
        clientesFuera=0;
        this.ventana=ventana;

    }

    public synchronized void ingresoCajero1(){
        JLabel c =new javax.swing.JLabel();
        cajero1=true;
        clientesCola--;
        int r = (int) (Math.random() * 3) + 1;
        if(r==1){
            ingresaDinero();
        }else if(r==2){
            retiraDinero();
        }
        cajero1=false;
        clientesFuera++;
        c.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Imagen/ico-nousuario.png")));
        c.setBounds(-40, -40, 60,60);
        ventana.getPanelCola().add(c);

    }

    try{
        Thread.sleep(4000);
    }catch(InterruptedException e){

    }

        cajero1=false;
        ventana.getPanelCola().remove(c);
    }

    public synchronized void ingresoCajero2(){

        JLabel c =new javax.swing.JLabel();
        cajero2=true;

```



```

        clientesCola--;
        int r = (int) (Math.random() * 3) + 1;
        if(r==1){
            ingresaDinero();
        }else if(r==2){
            retiraDinero();
        }
        try{
            Thread.sleep(4000);
        }catch(InterruptedException e){
        }

        cajero2=false;
        clientesFuera++;
        c.setIcon(new javax.swing.ImageIcon(
            getClass().getResource("/Imagen/iconousuario.png")));
        c.setBounds(-40, -40, 60,60);
        ventana.getPanelCola().add(c);

    }

    public synchronized void ingresoCajero3(){

        JLabel c =new javax.swing.JLabel();
        cajero3=true;
        clientesCola--;
        int r = (int) (Math.random() * 3) + 1;
        if(r==1){
            ingresaDinero();
        }else if(r==2){
            retiraDinero();
        }
        try{
            Thread.sleep(4000);
        }catch(InterruptedException e){
        }

        cajero3=false;
        clientesFuera++;
        c.setIcon(new javax.swing.ImageIcon(
            getClass().getResource("/Imagen/iconousuario.png")));
        c.setBounds(-40, -40, 60,60);
        ventana.getPanelCola().add(c);

    }

    public void ingresaDinero(){

        int randomNumber = (int) (Math.random() * 4) + 1;
        switch (randomNumber) {
            case 1:
                ventana.getjTextArea1().setText("Cliente "+clientes+" ingresa 100
euros");
                break;
            case 2:
                ventana.getjTextArea1().setText("Cliente "+clientes+"ingresa 50 eu-
ros");
                break;
            case 3:
                ventana.getjTextArea1().setText("Cliente "+clientes+"ingresa 20 eu-
ros");
                break;
            default:
                break;
        }
    }

```

```

    }
    try{
        Thread.sleep(3000);
    }catch(InterruptedException e){

    }
}

public void retiraDinero(){
    int randomNumber = (int) (Math.random() * 4) + 1;
    switch (randomNumber) {
        case 1:
            ventana.getjTextArea1().setText("Cliente "+clientes+" retira 100 eu-
ros");
            break;
        case 2:
            ventana.getjTextArea1().setText("Cliente "+clientes+"retira 50 eu-
ros");
            break;
        case 3:
            ventana.getjTextArea1().setText("Cliente "+clientes+"retira 20 eu-
ros");
            break;
        default:
            break;
    }
    try{
        Thread.sleep(3000);
    }catch(InterruptedException e){

    }
}
@Override
public void run() {

}

public int getClientes_max() {
    return clientes_max;
}

public void setClientes_max(int clientes_max) {
    this.clientes_max = clientes_max;
}

public int getClientes() {
    return clientes;
}

public void setClientes(int clientes) {
    this.clientes = clientes;
}

public int getClientesCola() {
    return clientesCola;
}

public void setClientesCola(int clientesCola) {
    this.clientesCola = clientesCola;
}

```

```

public int getClientesFuera() {
    return clientesFuera;
}

public void setClientesFuera(int clientesFuera) {
    this.clientesFuera = clientesFuera;
}
}

• Ec.edu.ups.vista
  ➤ VentanaPrincipal
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorCliente;
import ec.edu.ups.modelo.Cliente;
import ec.edu.ups.modelo.Cajero;
import java.util.Random;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

/**
 *
 * @author paul_
 */
public class VentanaPrincipal extends javax.swing.JFrame {

    final Cajero m = new Cajero (this);
    final Cliente c = new Cliente(m);
    ControladorCliente controladorCola;
    public VentanaPrincipal() {
        initComponents();
        controladorCola= new ControladorCliente(this,m);
        controladorCola.generarIngreso();
        controladorCola.generarSalida();
        controladorCola.generarColaDefault();
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        new Thread(c).start();
        new Thread(m).start();

        //inicializar clientes
        new Thread(new Runnable() {

            @Override
            public void run() {
                try {

                    Random r = new Random();
                    while(true){
                        Thread.sleep(r.nextInt(10000));
                        if(m.getClientesCola()==0||m.getClientesCola()==10){
                            Cliente nuevo = new Cliente(m);

```

```

        Thread t = new Thread(nuevo);
        t.start();
    }else{
        System.out.println("Llego un cliente y se fue");
    }
    }
} catch (InterruptedException ex) {

}

}).start();
}
public JPanel getjPanel1() {
    return jPanel1;
}

public JScrollPane getjScrollPane() {
    return jScrollPane1;
}

public JTextArea getjTextArea1() {
    return jTextArea1;
}

public JLabel getjLabel1() {
    return jLabel1;
}

public void setjLabel1(JLabel jLabel1) {
    this.jLabel1 = jLabel1;
}

public JLabel getjLabel2() {
    return jLabel2;
}

public void setjLabel2(JLabel jLabel2) {
    this.jLabel2 = jLabel2;
}

public JLabel getjLabel3() {
    return jLabel3;
}

public void setjLabel3(JLabel jLabel3) {
    this.jLabel3 = jLabel3;
}

public JPanel getPanelCajero1() {
    return panelCajero1;
}

public JPanel getPanelCajero2() {
    return panelCajero2;
}

public JPanel getPanelCajero3() {
    return panelCajero3;
}

public JPanel getPanelCola() {

```

```

    return panelCola;
}

public JPanel getPanelIngreso() {
    return panelIngreso;
}

public JPanel getPanelSalida() {
    return panelSalida;
}

```

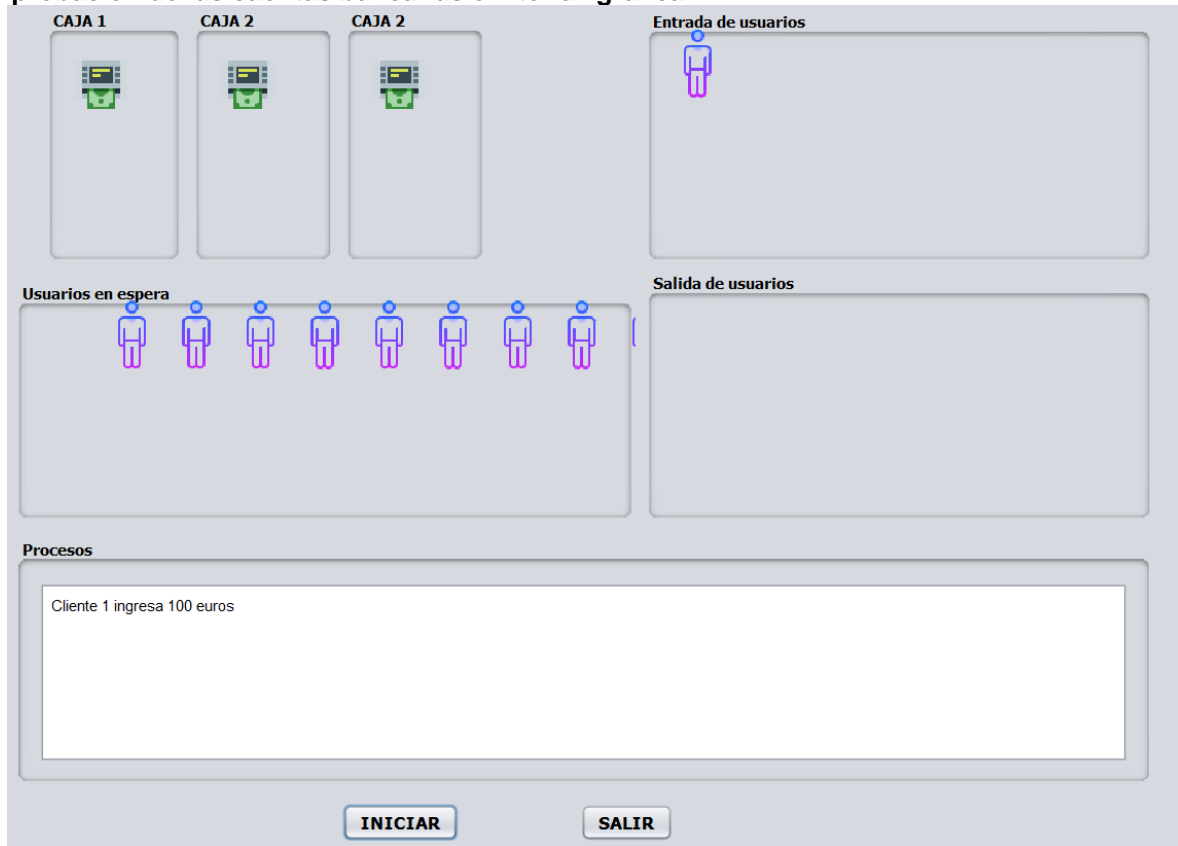
3- Como tercero y último la ejecución del programa.



The application window displays the following components:

- Top Section:** Three panels labeled "CAJA 1" and "CAJA 2" (repeated), each containing an icon of a cash register.
- Right Section:** Two large empty rectangular panels labeled "Entrada de usuarios" (top) and "Salida de usuarios" (bottom).
- Left Section:** A large empty rectangular panel labeled "Usuarios en espera".
- Bottom Section:** A large empty rectangular panel labeled "Procesos".
- Buttons:** Two buttons at the bottom center labeled "INICIAR" and "SALIR".

5. Comprobación de las cuentas bancarias e interfaz gráfica.



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 graficas 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:

PAUL ALEXANDER GUAPUCAL CARDENAS

Firma de estudiante: