



| | | |
|--|-----------------------|---|
|  | 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 grafica. | |
| | | 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.

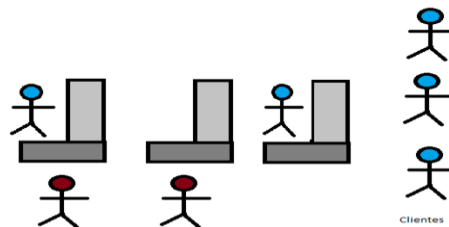
Se parte de una cuenta con 100 euros y se pueden tener procesos que ingresen 100 euros, 50 o 20. También se pueden tener procesos que retiran 100, 50 o 20 euros. Se desean tener los siguientes procesos:

- 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.


Ademas en el banco, existen 3 cajeros que pueden atender y hay un 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 numero 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%

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

- Hilos 30%
- Sincronización 10%
- Interfaz Gráfica de simulación 20%
- 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.
 - 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: COMPUTACIÓN/INGENIERÍA DE
SISTEMAS

ASIGNATURA: PROGRAMACIÓN APLICADA

NRO. PRÁCTICA:

1.1

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. 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 grafica.

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.**

6.

N.

RESULTADO(S) OBTENIDO(S):

- 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 grafica.
- Conclusiones y recomendaciones.
- Resultados.

CONCLUSIONES:

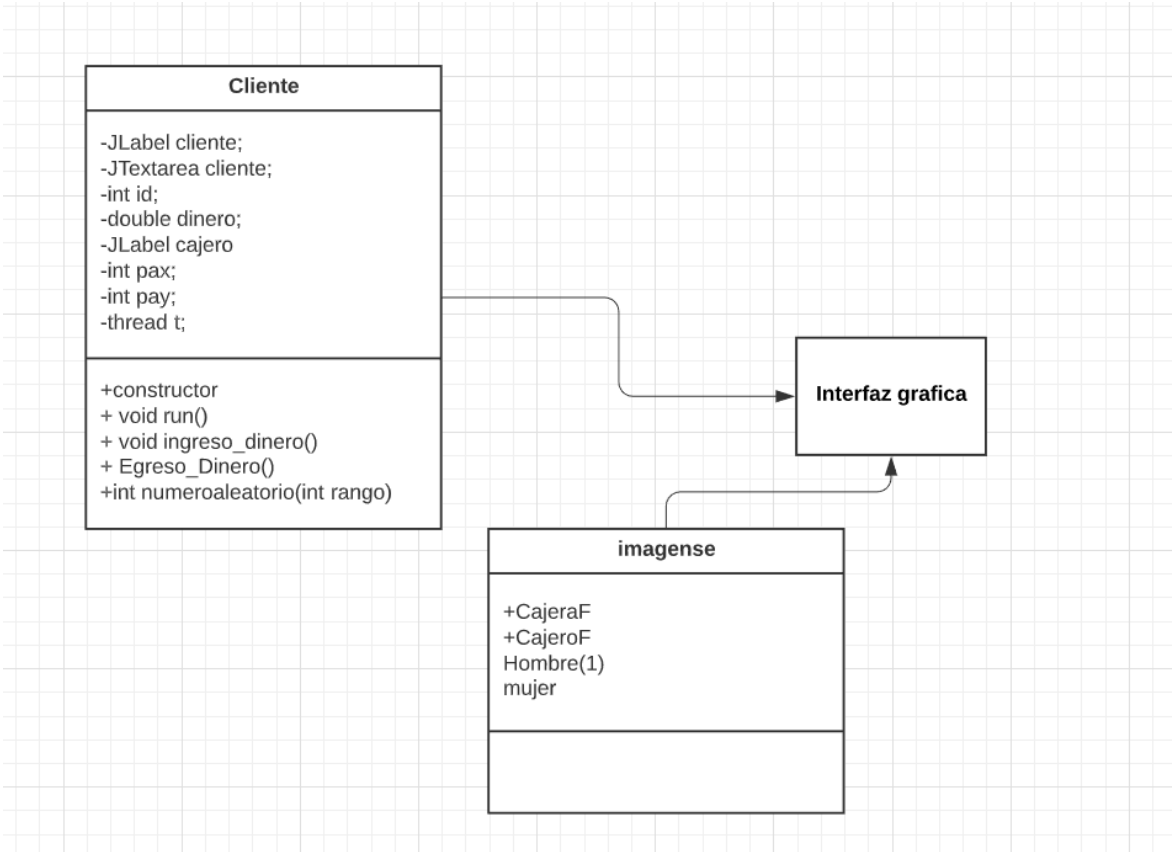
- Los estudiantes identifican las principales estructuras para la creacion de sistemas informaticos.
 - Los estudiantes implementan soluciones graficas en sistemas.
- Los estudiantes están en la capacidad de implementar hilos.

RECOMENDACIONES:

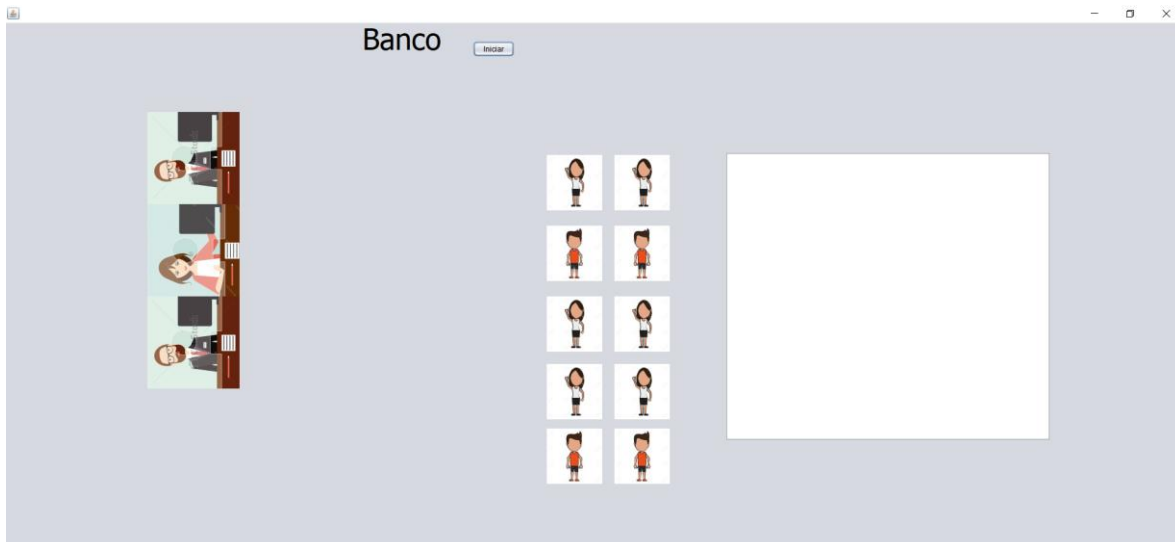
Nombre de estudiante: Adrian Lopez

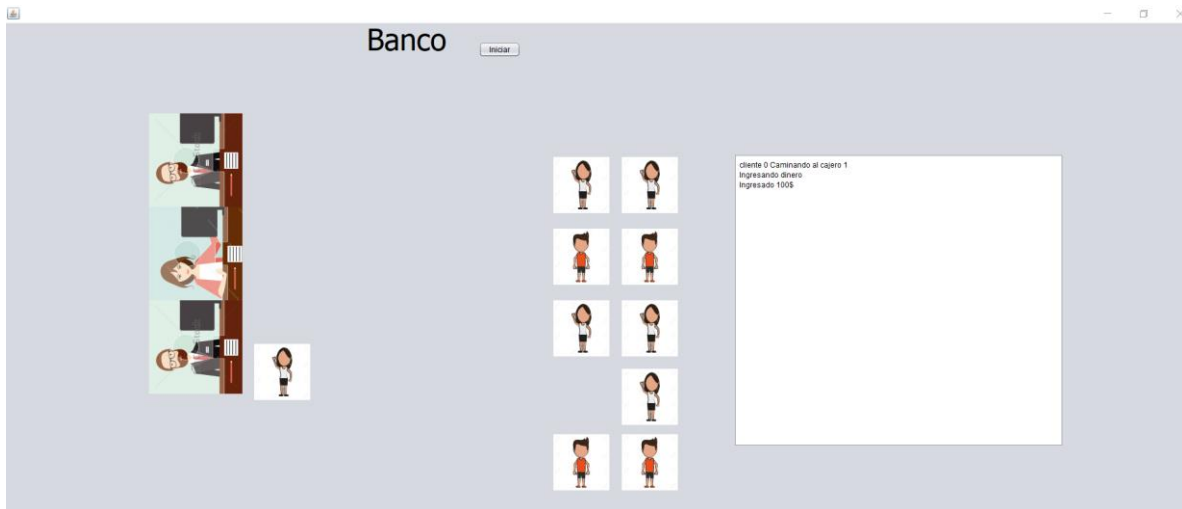
Firma de estudiante: _____

Diagrama de clase



Interfaz





Código

```

Pantalla.java x Cliente.java x Cliente.java x Pantalla.java x
Source Design History
1 package ups.edu.ec.vista;
2 import javax.swing.JLabel;
3 import ups.edu.ec.modelo.Cliente;
4 public class Pantalla extends javax.swing.JFrame {
5     public Pantalla() {
6         initComponents();
7         this.setExtendedState(this.MAXIMIZED_BOTH);
8     }
9
10    @SuppressWarnings("unchecked")
11    Generated Code
12
13
14
15
16
17
18
19    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
20        Cliente cli;
21        JLabel cliente[]=new JLabel[10];
22        JLabel cajero[]=new JLabel[3];
23
24        cliente[0]=c1; cajero[0]=cj1;
25        cliente[1]=c2; cajero[1]=cj2;
26        cliente[2]=c3; cajero[2]=cj3;
27        cliente[3]=c4;
28        cliente[4]=a;
29        cliente[5]=c6;
30        cliente[6]=c5;
31        cliente[7]=c8;
32        cliente[8]=c9;
33        cliente[9]=c10;
34
35
36        int i;
37        for (i = 0; i < 10; i++) {
38
39
40            cli=new Cliente(cliente[i], 90,jTextArea1, cajero[0],i);
41
42
43
44        }
45    }
46
47    private void formMouseClicked(java.awt.event.MouseEvent evt) {
48
49        System.out.println(cli.getLocation());
50        System.out.println("x"+evt.getX()+" "+y"+evt.getY());
51    }
52
53

```

```
Pantalla.java x Cliente.java x Cliente.java x Pantalla.java x
Source History
1 package ups.edu.ec.modelo;
2
3 import javax.swing.*;
4
5 public class Cliente implements Runnable {
6
7     private JLabel cliente;
8     private JTextArea textArea;
9     private int id;
10    private Thread t;
11    private double dinero;
12    private JLabel cajero;
13    private int pax;
14    private int pay;
15
16    public Cliente(JLabel cliente, double dinero, JTextArea texto, JLabel cajero, int id) {
17        this.cliente = cliente;
18
19        this.textArea = texto;
20        this.dinero = dinero;
21        this.cajero = cajero;
22
23        this.id = id;
24        t = new Thread(this);
25        t.start();
26    }
27
28    public void run() {
29        pax = cliente.getX();
30        pay = cliente.getY();
31        for (int i = 0; i < 1; i++) {
32
33            synchronized (this.cajero) {
34                textArea.append("cliente " + id + " Caminando al cajero 1\n");
35                cliente.setLocation(531, 320);
36
37                try {
38                    Thread.sleep(5000);
39
40                } catch (InterruptedException e) {
41                }
42                cliente.setLocation(399, 510);
43                int op = numeroaleatorio(2);
44                if (op == 1) {
45                    Ingreso_Dinero();
46                } else {
47                    Egreso_Dinero();
48                }
49            }
50        }
51    }
52}
```

```
Pantalla.java x Cliente.java x Cliente.java x Pantalla.java x
Source History
43 int op = numeroaleatorio(2);
44 if (op == 1) {
45     Ingreso_Dinero();
46 } else {
47     Egrese_Dinero();
48 }
49 cliente.setName("asd");
50 synchronized (this.cajero) {
51
52     textArea.append("cliente " + id + " Caminando al cajero 2\n");
53     cliente.setLocation(531, 320);
54
55     try {
56         Thread.sleep(5000);
57     } catch (InterruptedException e) {
58     }
59     cliente.setLocation(412, 350);
60     op = numeroaleatorio(1);
61     if (op == 1) {
62         Ingreso_Dinero();
63     } else {
64         Egrese_Dinero();
65     }
66     synchronized (this.cajero) {
67         textArea.append("cliente " + id + " Caminando al cajero 3\n");
68         cliente.setLocation(531, 320);
69
70         try {
71             Thread.sleep(5000);
72         } catch (InterruptedException e) {
73         }
74         cliente.setLocation(412, 200);
75         op = numeroaleatorio(1);
76         if (op == 1) {
77             Ingreso_Dinero();
78         } else {
79             Egrese_Dinero();
80         }
81     }
82 }
83
84 }
85
86 }
87
88 }
89
```



```
Pantalla.java x Cliente.java x Cliente.java x Pantalla.java x
Source History
82      }
83
84      }
85
86      }
87
88      }
89
90      }
91
92      }
93
94      public void Ingreso_Dinero() {
95          if (dinero == 100) {
96              textArea.append("cliente" + id + " su cuenta tiene 100 o mas\n");
97          } else {
98
99              textArea.append("Ingresando dinero\n");
100              int opcion = numeroaleatorio(3);
101
102              if (opcion == 0) {
103                  textArea.append("Ingresado 100$\n");
104                  dinero = dinero + 100;
105
106                  try {
107                      Thread.sleep(5000);
108                  } catch (InterruptedException e) {
109                  }
110                  cliente.setLocation(792, 787);
111              }
112
113              if (opcion == 1) {
114                  textArea.append("Ingresando 50$\n");
115                  dinero = dinero + 50;
116                  try {
117                      Thread.sleep(5000);
118                  } catch (InterruptedException e) {
119                  }
120                  cliente.setLocation(792, 787);
121              }
122
123              if (opcion == 2) {
124                  textArea.append("Ingresando 20$\n");
125                  dinero = dinero + 20;
126                  try {
127                      Thread.sleep(5000);
128                  }
```

```
Pantalla.java x Cliente.java x Cliente.java x Pantalla.java x
Source History
124         textArea.append("Ingresando 20$\n");
125         dinero = dinero + 20;
126         try {
127             Thread.sleep(5000);
128         } catch (InterruptedException e) {
129         }
130         cliente.setLocation(792, 787);
131     }
132 }
133
134
135 }
136
137 public void Egrese_Dinero() {
138     if (dinero == 100) {
139         textArea.append("cliente" + id + " su cuenta tiene 100$\n");
140     } else {
141
142         textArea.append("Sacando dinero\n");
143         int opcion = numeroaleatorio(3);
144
145         if (opcion == 0) {
146             textArea.append("Sacado 100$\n");
147             dinero = dinero - 100;
148
149             try {
150                 Thread.sleep(5000);
151             } catch (InterruptedException e) {
152             }
153             cliente.setLocation(792, 787);
154         }
155         if (opcion == 1) {
156             textArea.append("Sacado 50$\n");
157             dinero = dinero - 50;
158
159             try {
160                 Thread.sleep(5000);
161             } catch (InterruptedException e) {
162             }
163             cliente.setLocation(792, 787);
164         }
165         if (opcion == 2) {
166             textArea.append("Sacado 20$\n");
167             dinero = dinero - 20;
168         }
169     }
170 }
```

```
Pantalla.java x Cliente.java x Cliente.java x Pantalla.java x
Source History
151
152         } catch (InterruptedException e) {
153         }
154         cliente.setLocation(792, 787);
155     }
156     if (opcion == 1) {
157         textArea.append("Sacado 50$\n");
158         dinero = dinero - 50;
159
160         try {
161             Thread.sleep(5000);
162
163         } catch (InterruptedException e) {
164         }
165         cliente.setLocation(792, 787);
166     }
167     if (opcion == 2) {
168         textArea.append("Sacado 20$\n");
169         dinero = dinero - 20;
170
171         try {
172             Thread.sleep(5000);
173
174         } catch (InterruptedException e) {
175         }
176         cliente.setLocation(792, 787);
177     }
178 }
179 try {
180     Thread.sleep(5000);
181     textArea.append("Cajero libre\n");
182 } catch (InterruptedException e) {
183 }
184 cliente.setLocation(pax, pay);
185
186 }
187
188 public int numeroaleatorio(int rango) {
189     int n = (int) (Math.random() * rango);
190     return n;
191 }
192
193 }
194
```