Práctica 5

30 de octubre del 2019

Estructura de Datos

Número de Control – Nombre:

18131209 – ADAME SANDOVAL JOSE MISAEL

18131257 – MARTINEZ BLANCO JOSE BRYAN

Descripción:

ESCRIBE UNA APLICACION NETBEANS JAVA QUE LEA Y VISUALICE A DOS GRUPOS DE ALUMNOS SEGUN SE HIZO EN CLASE. ALMACENA A LOS GRUPOS EN SU RESOECTIVA LISTA ENLAZADA.

COMPETENCIA 3

1. Escribe un método en la clase Lista, que elimine al segundo elemento de la lista visto desde la cabecera. Si no existe este segundo elemento, entonces no se efectúa nada. NOTA: puedes usar el atributo \_noNodos. NO PUEDES UTILIZAR OTRAS PROPIEDADES O METODOS. NO LISTAS AUXILIARES NI ARREGLOS AUXILIARES.

2. Escribe un método en la clase Lista, que elimine a todos los nodos de la lista menos al primero y al último. NO PUEDES UTILIZAR OTRAS PROPIEDADES O METODOS.

3. Escribe un método en la clase Lista, que retorne un true, si el primer y último elemento en la lista son iguales. NO PUEDES UTILIZAR OTRAS PROPIEDADES O METODOS.

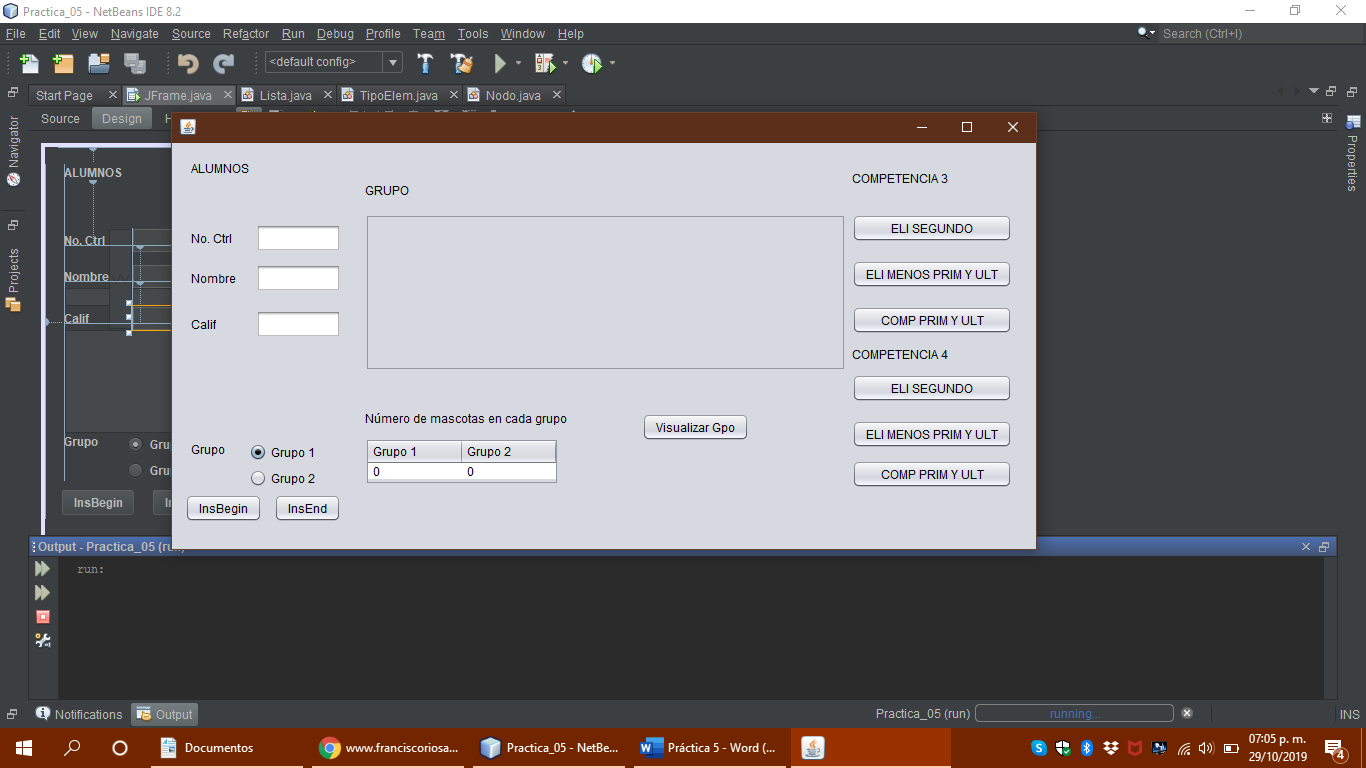
COMPETENCIA 4.

AGREGA 3 BOTONES A TU APLICACION QUE REALICEN LAS TAREAS DE LOS 3 METODOS SOLICITADOS EN LA COMPETENCIA 3.

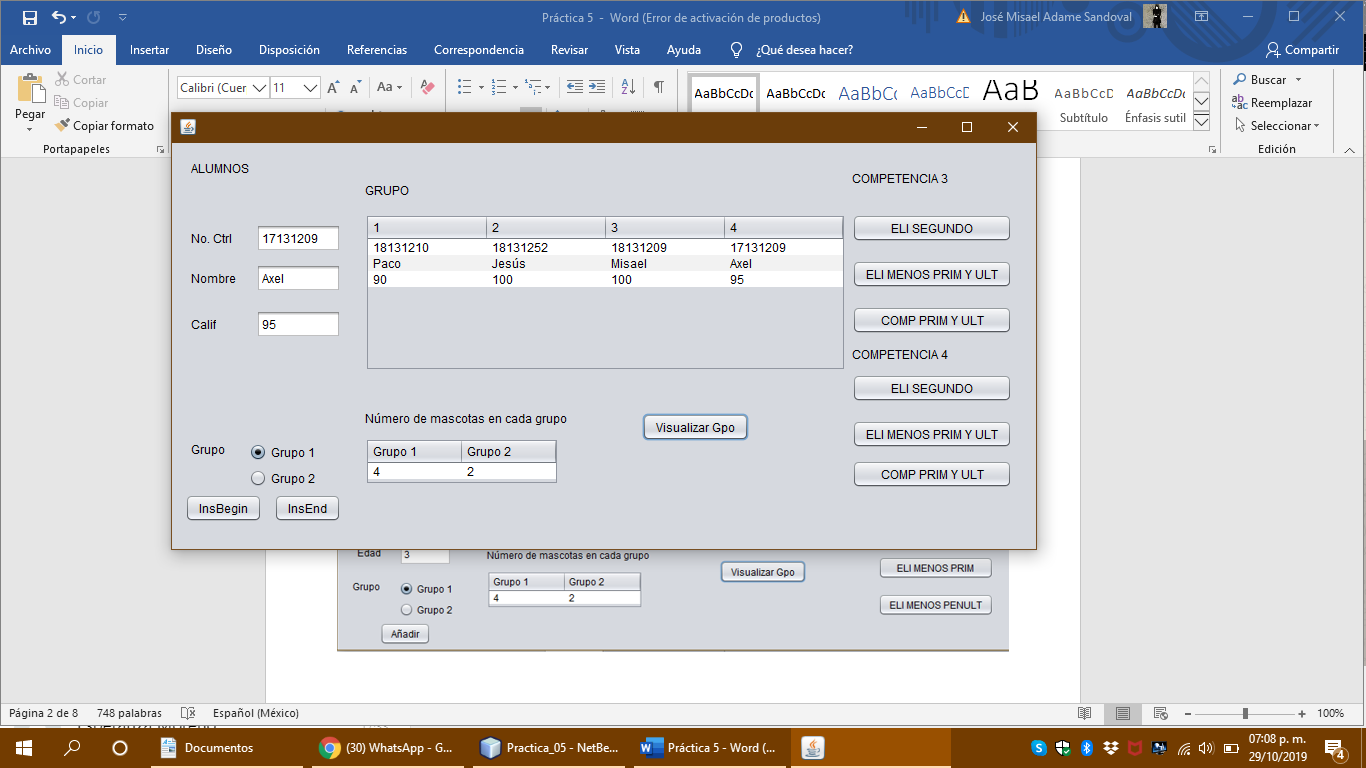
**DESARROLLO:**

Interfaz Gráfica

al inicio



Con registros (se muestra el grupo 1)



**DEFINICIÓN DE LOS OBJETOS:**

public class JFrame extends javax.swing.JFrame {

Lista grupo1 = new Lista();

Lista grupo2 = new Lista();

public JFrame() {

initComponents();

}

**CLASES:**

**TipoElem**

public class TipoElem {

private String \_noCon;

private String \_nombre;

private int \_calif;

public TipoElem(String noCon,String nombre,int calif) {

\_noCon = noCon;

\_nombre = nombre;

\_calif = calif;

}

public TipoElem() {

// solo reservo espacio de memoria

}

public String NoCon() {

return \_noCon;

}

public String Nombre() {

return \_nombre;

}

public int Calif() {

return \_calif;

}

public boolean compararAlums(TipoElem elem){

return \_noCon.equals(elem.\_noCon) && \_nombre.equals(elem.\_nombre) &&

\_calif == elem.\_calif;

}

}

**Nodo**

public class Nodo {

private TipoElem \_info;

private Nodo \_sig;

public Nodo(TipoElem elem) {

\_info = elem;

\_sig = null;

}

public TipoElem Info() {

return \_info;

}

public Nodo Sig() {

return \_sig;

}

public void Info(TipoElem elem) {

\_info = elem;

}

public void Sig(Nodo sig) {

\_sig = sig;

}

}

**Lista**

public class Lista {

private Nodo \_cab;

private int \_noNodos;

public Lista() {

\_cab = null;

\_noNodos = 0;

}

public void Clear() {

\_cab = null;

\_noNodos = 0;

}

public boolean Empty() {

return \_cab == null;

}

public int Length() {

return \_noNodos;

}

public void Length(int valor) {

\_noNodos = valor;

}

public Nodo Cab() {

return \_cab;

}

public void Cab(Nodo cab) {

\_cab = cab;

}

public void InsBegin(TipoElem elem) {

Nodo p = new Nodo(elem);

p.Sig(\_cab);

Cab(p);

\_noNodos++;

}

public void InsEnd(TipoElem elem) {

Nodo p = new Nodo(elem);

Nodo q = \_cab;

if (q == null)

\_cab = p;

else {

while(q.Sig() != null)

q = q.Sig();

q.Sig(p);

}

\_noNodos++;

}

public void EliminarSegundo() {

if(\_noNodos > 1) {

\_cab.Sig(\_cab.Sig().Sig());

\_noNodos--;

}

}

public void EliminarMenosPrimYUlt() {

if(\_noNodos > 2) {

Nodo p = \_cab;

while(p.Sig() != null)

p = p.Sig();

\_cab.Sig(p);

\_noNodos=2;

}

}

public boolean CompararPrimYUlt() {

if(\_noNodos > 1) {

Nodo p = \_cab;

while(p.Sig() != null)

p = p.Sig();

return \_cab.Info().compararAlums(p.Info());

}

return false;

}

}

**MÉTODOS DE LA APLICACIÓN:**

private void formWindowOpened(java.awt.event.WindowEvent evt) {

jTabNoAlumnos.setValueAt(grupo1.Length(), 0, 0);

jTabNoAlumnos.setValueAt(grupo2.Length(), 0, 1);

}

private void actualizarNoAlumnosGpos() {

jTabNoAlumnos.setValueAt(grupo1.Length(), 0, 0);

jTabNoAlumnos.setValueAt(grupo2.Length(), 0, 1);

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

if(jTextNoCtrl.getText().isEmpty() || jTextNombre.getText().isEmpty()

|| jTextCalif.getText().isEmpty()) {

JOptionPane.showMessageDialog(this, "NO SE PUDO GUARDAR LA MASCOTA"

+ " , DEBE LLENAR TODOS LOS CAMPOS",

"ERROR", JOptionPane.ERROR\_MESSAGE); }

else {

TipoElem elem = new TipoElem(jTextNoCtrl.getText(), jTextNombre.getText(),

Integer.parseInt(jTextCalif.getText()));

grupoSel.InsBegin(elem);

actualizarNoAlumnosGpos();

}

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

if(jTextNoCtrl.getText().isEmpty() || jTextNombre.getText().isEmpty()

|| jTextCalif.getText().isEmpty()) {

JOptionPane.showMessageDialog(this, "NO SE PUDO GUARDAR LA MASCOTA"

+ " , DEBE LLENAR TODOS LOS CAMPOS",

"ERROR", JOptionPane.ERROR\_MESSAGE); }

else {

TipoElem elem = new TipoElem(jTextNoCtrl.getText(), jTextNombre.getText(),

Integer.parseInt(jTextCalif.getText()));

grupoSel.InsEnd(elem);

actualizarNoAlumnosGpos();

}

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

DefaultTableModel modelo = (DefaultTableModel) jTabAlumnos.getModel();

modelo.setColumnCount(0);

for(int i=1; i<= grupoSel.Length(); i++)

modelo.addColumn(i);

int col = 0;

Nodo p = grupoSel.Cab();

while (p != null) {

jTabAlumnos.setValueAt(p.Info().NoCon(),0,col);

jTabAlumnos.setValueAt(p.Info().Nombre(),1,col);

jTabAlumnos.setValueAt(p.Info().Calif(),2,col++);

p = p.Sig();

}

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

grupoSel.EliminarSegundo();

actualizarNoAlumnosGpos();

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

grupoSel.EliminarMenosPrimYUlt();

actualizarNoAlumnosGpos();

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

boolean c = grupoSel.CompararPrimYUlt();

if(c)

JOptionPane.showMessageDialog(null, "Verdadero");

else

JOptionPane.showMessageDialog(null, "Falso");

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

if(grupoSel.Length() > 1) {

grupoSel.Cab().Sig(grupoSel.Cab().Sig().Sig());

grupoSel.Length(grupoSel.Length()-1);

actualizarNoAlumnosGpos();

}

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

if(grupoSel.Length() > 2) {

Nodo p = grupoSel.Cab();

while(p.Sig() != null)

p = p.Sig();

grupoSel.Cab().Sig(p);

grupoSel.Length(2);

actualizarNoAlumnosGpos();

}

}

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

Lista grupoSel = rb1.isSelected()? grupo1 : grupo2;

if(grupoSel.Length() > 1) {

Nodo p = grupoSel.Cab();

while(p.Sig() != null)

p = p.Sig();

boolean c = grupoSel.Cab().Info().compararAlums(p.Info());

if(c)

JOptionPane.showMessageDialog(null, "Verdadero");

else

JOptionPane.showMessageDialog(null, "Falso");

}

else

JOptionPane.showMessageDialog(null, "Falso");

}