Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela Ciencias y sistemas

MANUAL TÉCNICO

Introducción a la programación y computación 1
Rubén Alejandro Ralda Mejia
202111835

Guatemala 8 de marzo del 2022

Iniciar sesión

Buscamos el usuario y lo comparamos con la caja de texto desplegando un mensaje si no coincide el usuario o este no existe.

```
        boolean existe=true;
        boolean coincide=true;
        for (int i = 0; i < 50; i++) {
            if (usuario[i]!=null) {
                if (usuario[i].getUser().equals(txt_usuario.getText())) {
                    existe=true;
                    if (usuario[i].getPass().equals(txt_contra.getText())) {
                        if (usuario[i].getUser().equalsIgnoreCase("admin")) {
                            Admin admin= new Admin(usuario,libros);
                            admin.setVisible(true);
                            this.dispose();
                            break;
                        }else{
                            Usuario_comun usu= new Usuario_comun(usuario,libros,i);
                            usu.setVisible(true);
                            this.dispose();
                            break;
                        }

                    }else{
                        JOptionPane.showMessageDialog(null,"El usuario y contraseña "
                            + "no coinciden por favor revise sus datos",
                            "Error",JOptionPane.ERROR_MESSAGE);
                    }
                }else{
                    existe=false;
                }
            }
        }
        if (existe==false) {
            JOptionPane.showMessageDialog(null,"El usuario no existe, "
```

Para crear un usuario

```
if (txt_contra.getText().equals(txt_confirmar.getText())) {
    try{
        int j=0;
        boolean existe=false;
        while(guardar[j]!=null && existe==false ){
            if (guardar[j].getDpi()==Integer.parseInt(txt_id.getText())) {
                JOptionPane.showMessageDialog(null,"El ID ya existe"
                    + ", utilice otro",
                    "Error",JOptionPane.ERROR_MESSAGE);
                existe=true;
            }
            j++;
            if (j>49) {
                break;
            }
        }
        if (!existe) {
            Libros[] prestamos= new Libros[50];
            guardar[j]= new Usuarios(Integer.parseInt(txt_id.getText()),txt_nombre.getText(),txt_ap
                txt_user.getText(),txt_rol.getText(),txt_contra.getText(),prestamos);
            txt_id.setText("");
            txt_nombre.setText("");
            txt_apellido.setText("");
            txt_user.setText("");
            txt_rol.setText("");
            txt_contra.setText("");
            txt_confirmar.setText("");
            JOptionPane.showMessageDialog(null,"El usuario se creó con exito","Mensaje",JOptionPane
        }

    }catch(Exception e){
```

Para eliminar un usuario:

```
if (j>1) {
    int i=j-1;
    for (; i < eliminar.length-1; i++) {
        eliminar[i]=eliminar[i+1];
    }
    eliminar[i]=null;
    j=0;
    JOptionPane.showMessageDialog(null,"El usuario se eliminó correctamente","Mensaje",JOptionPane.I
    txt_id.setText("");
    txt_nombre.setText("");
    txt_apellido.setText("");
    txt_user.setText("");
    txt_rol.setText("");
    txt_contra.setText("");
}
```

Para modificar un usuario:

```
if (j>1) {
    try {
        int i=0;
        boolean existe=false;
        while(modificar[i]!=null && existe==false){
            if (modificar[i].getDpi()==Integer.parseInt(txt_id.getText())) {
                if (j-1!=i) {
                    existe=true;
                }
            }
            i++;
            if (i>49) {
                break;
            }
        }
        if (!existe) {
            modificar[j-1].setDpi(Integer.parseInt(txt_id.getText()));
            modificar[j-1].setNombre(txt_nombre.getText());
            modificar[j-1].setApellido(txt_apellido.getText());
            modificar[j-1].setUser(txt_user.getText());
            modificar[j-1].setRol(txt_rol.getText());
            modificar[j-1].setPass(txt_contra.getText());
```

Para mostrar un usuario:

```
public void mostrar(){
    String matriz[][]= new String[mostrar.length][7];
    for (int i = 0; i < mostrar.length; i++) {
        if (mostrar[i]!=null) {
            matriz[i][0]=String.valueOf(i);
            matriz[i][1]=String.valueOf(mostrar[i].getDpi());
            matriz[i][2]=mostrar[i].getNombre();
            matriz[i][3]=mostrar[i].getApellido();
            matriz[i][4]=mostrar[i].getUser();
            matriz[i][5]=mostrar[i].getRol();
            matriz[i][6]=mostrar[i].getPass();
        }
    }
     Table_mostrar.setModel(new javax.swing.table.DefaultTableModel(
        matriz,
        new String [] {
            "No.", "DPI","Nombre", "Apellido","User","Rol","Contraseña"
        }
    ));
}
```

Crear bibliografía

Para la carga individual

```
for (; libros[i] != null; i++) {
}
String[] claves = txt_clave.getText().split(",");
String[] temas = txt_temas.getText().split(",");
switch (combo_tipo.getSelectedIndex()) {
    case 0:
        //libros
        libros[i] = new Libros(combo_tipo.getSelectedIndex(), txt_autor.getText(), Integer.parseInt(txt_an
                Integer.parseInt(txt_isbn.getText()), txt_titulo.getText(), Integer.parseInt(txt_edicion.
               Integer.parseInt(txt_copias.getText()),
                Integer.parseInt(txt_disponible.getText()));
        JOptionPane.showMessageDialog(null, "El libro se creó con exito", "Mensaje", JOptionPane.INFORMATI
        break;
    case 1:
        //revista
        libros[i] = new Libros(combo_tipo.getSelectedIndex(), txt_autor.getText(), Integer.parseInt(txt_an
                txt_titulo.getText(), Integer.parseInt(txt_edicion.getText()), claves, txt_descripcion.ge
               Integer.parseInt(txt_copias.getText()), txt_categoria.getText(), Integer.parseInt(txt_ejem
                Integer.parseInt(txt_disponible.getText()));
        JOptionPane.showMessageDialog(null, "La revista se creó con exito", "Mensaje", JOptionPane.INFORMA
        break;
    case 2:
        //tesis
        libros[i] = new Libros(combo_tipo.getSelectedIndex(), txt_autor.getText(), Integer.parseInt(txt_an
                txt_titulo.getText(), Integer.parseInt(txt_edicion.getText()), claves, txt_descripcion.ge
               Integer.parseInt(txt_copias.getText()), txt_area.getText(),
                Integer.parseInt(txt_disponible.getText()));
        JOptionPane.showMessageDialog(null, "La tesis se creó con exito", "Mensaje", JOptionPane.INFORMATI
        break;
    case 3:
        //digital
```

Para la carga masiva:

```
String[] tipos = txt_area.getText().split("\n");
String[] libro;

for (int i = 0; i < tipos.length; i++) {
    if (tipos[i] != null) {
        libro = tipos[i].split(";");
        try {
            int k=0;
            for (; libros[k] != null; k++) {
            }
            String[] claves;
            String[] temas;
            switch (Integer.parseInt(libro[0])) {
                case 0:
                    //libros
                    claves=libro[6].split(",");
                    temas=libro[8].split(",");
                    libros[k] = new Libros(Integer.parseInt(libro[0]), libro[1], Integer.parseInt(libro[2]),
                            Integer.parseInt(libro[3]), libro[4], Integer.parseInt(libro[5]), claves, libro[
                           Integer.parseInt(libro[9]),
                            Integer.parseInt(libro[13]));
                    JOptionPane.showMessageDialog(null, "El libro se creó con exito", "Mensaje", JOptionPane.
                    break;
                case 1:
                    //revista
                    claves=libro[6].split(",");
                    temas=libro[8].split(",");
                    libros[k] = new Libros(Integer.parseInt(libro[0]), libro[1], Integer.parseInt(libro[2]),
                            libro[4], Integer.parseInt(libro[5]), claves, libro[7], temas,
                           Integer.parseInt(libro[9]), libro[10], Integer.parseInt(libro[11]),
                            Integer.parseInt(libro[13]));
```

Para la creación del reporte de libros:

```
nuevaLinea.println("<h1>Reporte libros</h1>");
for (int i = 0; i < libros.length; i++) {
    if (libros[i] != null) {
        //muestro el libro
        nuevaLinea.println("<p><b>Titulo:</b> " + libros[i].getTitulo() + "</p>");
        nuevaLinea.println("<p><b>Autor:</b> " + libros[i].getAutor() + "</p>");
        nuevaLinea.println("<p><b>Descripcion:</b></p><p>" + libros[i].getDescripcion() + "</p>");
        nuevaLinea.println("<table border=\"1\">");
        nuevaLinea.println("<tr>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Nombre</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Apellido</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Rol</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.println("</tr>");
        for (int j = 0; j < usuario.length; j++) {
            if (usuario[j] != null) {
                for (int k = 0; k < usuario[j].getLibros().length; k++) {
                    if (usuario[j].getLibros()[k] != null) {
                        if (usuario[j].getLibros()[k].equals(libros[i])) {
                            //si el libro coincide con el libro prestado por cada usuario
                            //fila de datos del usuario
                            nuevaLinea.println("<tr>");
                            nuevaLinea.print("<td>");
                            nuevaLinea.print(usuario[j].getNombre());
                            nuevaLinea.print("</td>");
                            nuevaLinea.print("<td>");
```

Para el reporte de usuarios:

```
nuevaLinea.println("<h1>Reporte usuarios</h1>");
for (int i = 1; i < usuario.length; i++) {
    if (usuario[i] != null) {
        //muestro el libro
        nuevaLinea.println("<p><b>Nombres:</b> " + usuario[i].getNombre() + "</p>");
        nuevaLinea.println("<p><b>Apellidos:</b> " + usuario[i].getApellido() + "</p>");
        nuevaLinea.println("<p><b>Rol:</b></p><p>" + usuario[i].getRol() + "</p>");
        nuevaLinea.println("<table border=\"1\">");
        nuevaLinea.println("<tr>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Titulo</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Autor</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Año de publicación</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>Tipo</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.println("</tr>");
        for (int j = 0; j < usuario[i].getLibros().length; j++) {
            if (usuario[i].getLibros()[j] != null) {
                //si el libro coincide con el libro prestado por cada usuario
                //fila de datos del usuario
                nuevaLinea.println("<tr>");
                nuevaLinea.print("<td>");
                nuevaLinea.print(usuario[i].getLibros()[j].getTitulo());
                nuevaLinea.print("</td>");
                nuevaLinea.print("<td>");
```

Biblioteca virtual

Para agregar un libro a la biblioteca virtual:

```
Libros[] digital = usuarios[posi].getLibros();
int ultimo = 0;
boolean existe = false;

if (tabla_libros.getSelectedRow() != -1) {
    while (digital[ultimo] != null && existe == false) {
        if (digital[ultimo].equals(libros[disponibles[tabla_libros.getSelectedRow()]])) {
            existe = true;
            JOptionPane.showMessageDialog(null, "El libro ya existe en su "
                    + "Biblioteca virtual",
                    "Error", JOptionPane.ERROR_MESSAGE);
        }
        ultimo++;
    }
    if (!existe) {
        int i = disponibles[tabla_libros.getSelectedRow()];
        digital[ultimo] = libros[i];
        usuarios[posi].setLibros(digital);
        JOptionPane.showMessageDialog(null, "Se ha agregado el libro a su biblioteca virtual");
    }

} else {
    JOptionPane.showMessageDialog(null, "Seleccione una fila de la tabla");
}
```

Para mostrar los libros virtuales en la tabla:

```
public void mostrar(int i, int j) {
    Object matriz[][] = new Object[libros.length][14];
    if (libros[j].getTipo() == 3) {
        matriz[i][0] = i;
        disponibles[i] = j;
        matriz[i][1] = libros[j].getAutor();
        matriz[i][2] = String.valueOf(libros[j].getAnio_publi());
        matriz[i][3] = libros[j].getTitulo();
        matriz[i][4] = String.valueOf(libros[j].getEdicion());
        matriz[i][5] = "";
        for (int k = 0; k < libros[j].getClaves().length; k++) {
            matriz[i][5] += libros[j].getClaves()[k] + ",";
        }
        matriz[i][6] = libros[j].getDescripcion();
        matriz[i][7] = "";
        for (int k = 0; k < libros[j].getTemas().length; k++) {
            matriz[i][7] += libros[j].getTemas()[k] + ",";
        }
        matriz[i][8] = String.valueOf(libros[j].getTamano());
        DefaultTableModel model = (DefaultTableModel) tabla_libros.getModel();
        model.addRow(matriz[i]);
    }

}
```

Para buscar por atributos los libros:

```
DefaultTableModel modelo = (DefaultTableModel) tabla_libros.getModel();
int fila = tabla_libros.getRowCount();
for (int i = 0; i < fila; i++) {
    modelo.removeRow(0);
}
int j = 0;
try {
    switch (combo_atributo.getSelectedIndex()) {
        case 0:
            for (int i = 0; i < libros.length; i++) {
                if (libros[i] != null) {
                    if (libros[i].getAutor().equals(txt_buscar.getText())) {

                        mostrar(j, i);
                        j++;
                    }
                }
            }
            break;
        case 1:
            for (int i = 0; i < libros.length; i++) {
                if (libros[i] != null) {
                    if (String.valueOf(libros[i].getAnio_publi()).equals(txt_buscar.getText())) {
                        mostrar(j, i);
                        j++;
                    }
                }
            }
            break;
```

Para eliminar un libro digital del usuario:

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    int i = disponibles[list_libros.getSelectedIndex()];
    for (; i < usuario[posi].getLibros().length - 1; i++) {
        usuario[posi].getLibros()[i] = usuario[posi].getLibros()[i + 1];
    }
    txt_atributos.setText("");
    model.clear();
    mostrar();
    JOptionPane.showMessageDialog(this, "Se ha eliminado el libro de su biblioteca virtual");
    jButton3.setEnabled(false);
}
```

Prestamos de libros

Para prestar un libro dependiendo del tipo y restando su cantidad:

```
Libros[] digital = usuarios[posi].getLibros();
int ultimo = 0;
boolean existe = false;
if (tabla_libros.getSelectedRow() != -1) {
    while (digital[ultimo] != null && existe == false) {
        if (digital[ultimo].equals(libros[disponibles[tabla_libros.getSelectedRow()]])) {
            existe = true;
            JOptionPane.showMessageDialog(null, "El libro ya ha sido "
                    + "prestado",
                    "Error", JOptionPane.ERROR_MESSAGE);
        }
        ultimo++;
    }
    if (!existe) {
        int i = disponibles[tabla_libros.getSelectedRow()];
        if (libros[i].getDisponible() > 0) {
            digital[ultimo] = libros[i];
            libros[i].setDisponible(libros[i].getDisponible() - 1);
            usuarios[posi].setLibros(digital);
            JOptionPane.showMessageDialog(null, "Se ha prestado el libro a su biblioteca");
        } else {
            JOptionPane.showMessageDialog(null, "No se pudó realizar el préstamo porque "
                    + "actualmente no hay disponibles",
                    "Error", JOptionPane.ERROR_MESSAGE);
        }
    }

} else {
    JOptionPane.showMessageDialog(null, "Seleccione una fila de la tabla");
}
```

Para la creación del reporte de existencias:

```
nuevaLinea.print("<b>Año de publicación</b>");
nuevaLinea.print("</td>");
nuevaLinea.print("<td>");
nuevaLinea.print("<b>Edición</b>");
nuevaLinea.print("</td>");
nuevaLinea.print("<td>");
nuevaLinea.print("<b>Copias</b>");
nuevaLinea.print("</td>");
nuevaLinea.print("<td>");
nuevaLinea.print("<b>Disponible</b>");
nuevaLinea.print("</td>");
nuevaLinea.print("<td>");
nuevaLinea.print("<b>Tipo</b>");
nuevaLinea.print("</td>");
nuevaLinea.println("</tr>");
for (int i = 0; i < libros.length; i++) {
    if (libros[i] != null) {
        //muestro el libro
        if (libros[i].getTipo() != 3) {
            nuevaLinea.println("<tr>");
            nuevaLinea.print("<td>");
            nuevaLinea.print(libros[i].getTitulo());
            nuevaLinea.print("</td>");
            nuevaLinea.print("<td>");
            nuevaLinea.print(libros[i].getAutor());
            nuevaLinea.print("</td>");
            nuevaLinea.print("<td>");
            nuevaLinea.print(libros[i].getAnio_publi());
            nuevaLinea.print("</td>");
            nuevaLinea.print("<td>");
            nuevaLinea.print(libros[i].getEdicion());
```

Para mostrar lo libros prestados en una tabla:

```
private void mostrar() {
    int j = 0;
    Object matriz[][] = new Object[usuario[posi].getLibros().length][5];
    for (int i = 0; i < usuario[posi].getLibros().length; i++) {
        if (usuario[posi].getLibros()[i] != null) {
            if (usuario[posi].getLibros()[i].getTipo() != 3) {
                disponibles[j] = i;
                matriz[j][0] = j;
                matriz[j][1] = usuario[posi].getLibros()[i].getTitulo();
                switch (usuario[posi].getLibros()[i].getTipo()) {
                    case 0:
                        matriz[j][2] = "Libro";
                        break;
                    case 1:
                        matriz[j][2] = "Revista";
                        break;
                    case 2:
                        matriz[j][2] = "Tesis";
                        break;
                    default:
                        throw new AssertionError();
                }
                DefaultTableModel model = (DefaultTableModel) tabla_pres.getModel();
                model.addRow(matriz[j]);
                j++;
            }
        }
    }
}
```

Para devolver un libro prestado y sumar la cantidad de disponibles:

```
if (tabla_pres.getSelectedRow() != -1) {
    int i = disponibles[tabla_pres.getSelectedRow()];

    for (int j = 0; j < libro.length; j++) {
        if (usuario[posi].getLibros()[i].equals(libro[j])) {
            libro[j].setDisponible(libro[j].getDisponible() + 1);
        }
    }
    for (; i < usuario[posi].getLibros().length - 1; i++) {
        usuario[posi].getLibros()[i] = usuario[posi].getLibros()[i + 1];
    }
    DefaultTableModel model = (DefaultTableModel) tabla_pres.getModel();
    int fila = tabla_pres.getRowCount();
    for (int k = 0; k < fila; k++) {
        model.removeRow(0);
    }
    JOptionPane.showMessageDialog(this, "Se ha devuelto el libro a la biblioteca");
    mostrar();
}else{
     JOptionPane.showMessageDialog(null, "Seleccione una fila", "Error", JOptionPane.ERROR_MESSAGE);
}
```