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 23 de marzo del 2022

Principal

Librerías

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.BorderFactory;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.category.DefaultCategoryDataset;
```

Graficar

El botón examinar guarda la ruta en el textbox

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    JFileChooser archivo = new JFileChooser();
    int valor = archivo.showOpenDialog(this);
    if (valor == JFileChooser.APPROVE_OPTION) {
        txtruta.setText(archivo.getSelectedFile().getAbsolutePath());
    }
}
```

El botón generar gráfica, obtiene la ruta y el título (si no tiene título se utiliza un predeterminado).

```
// TODO add your handling code here:
String titulo = null;
if (!txtruta.getText().equalsIgnoreCase("")) {
    if (txtnombre.getText().equalsIgnoreCase("")) {
        titulo = "Grafica de ordenamiento";
        txtnombre.setText(titulo);
    } else {
        titulo = txtnombre.getText();
    }
    File datos = new File(txtruta.getText());
    BufferedReader br = null;
```

Con al archivo determino el número de líneas y creo los vectores con ese tamaño, luego lleno los datos en las vectores.

```
//tamaño del vector
Integer tamano = 0;
Scanner entrada = new Scanner(datos);
while (entrada.hasNext()) {
    entrada.nextLine();
    tamano++;
}
//llenar los datos en vectores separados
br = new BufferedReader(new FileReader(datos));
String[][] temporal = new String[tamano][2];
valoresy = new int[tamano - 1];
valoresx = new String[tamano - 1];
String line = br.readLine();
int i = 0;
String[] encabezado1 = line.split(",");
line = br.readLine();
while (null != line) {
    temporal[i] = line.split(",");
    line = br.readLine();
    valoresy[i] = Integer.parseInt(temporal[i][1]);
    valoresx[i] = temporal[i][0];
    i++;
}
```

Invoco el método “mostrar” hace la gráfica y lo muestra en el Jpanel

```
DefaultCategoryDataset dataset = new DefaultCategoryDataset();
//añadir los valores a la grafica
for (int j = 0; j < tempox.length; j++) {
    dataset.setValue(tempoy[j], "", tempox[j]);
}
//graficar
JFreeChart barChart = ChartFactory.createBarChart(
        titulo,
        encabezado1[0],
        encabezado1[1],
        dataset,
        PlotOrientation.VERTICAL,
        false, true, false);
ChartPanel panel = new ChartPanel(barChart);
panel.setBorder(BorderFactory.createEmptyBorder(15, 15, 15, 15));
panel.setPreferredSize(new Dimension(400, 400));
panel.setBackground(Color.white);
jPanel1.removeAll();
jPanel1.setLayout(new BorderLayout());
jPanel1.add(panel, BorderLayout.NORTH);
pack();
repaint();
revalidate();
```

Ordenar

Verifico si existen datos y se determina cual radiobutton esta seleccionado para llamar a la clase correspondiente para ordenar y enviar los datos utilizando hilos.

```
if (valoresy != null) {
    ordenadoy = new int[valoresx.length];
    ordenadox = new String[valoresx.length];
    for (int j = 0; j < valoresx.length; j++) {
        ordenadoy[j] = valoresy[j];
        ordenadox[j] = valoresx[j];
    }
    if (insertion.isSelected()) {
        if (ascend.isSelected() == true) {
            Insercion insercion = new Insercion(labelpasos, ordenadox, ordenadoy, jPanel1, encabezadol, titulo,
            insercion.start();

        } else if (descend.isSelected() == true) {
            Insercion insercion = new Insercion(labelpasos, ordenadox, ordenadoy, jPanel1, encabezadol, titulo,
            insercion.start();
        }
        jButton4.setEnabled(true);
        algoritmo = "Ordenamiento por insercion";
    } else if (merge.isSelected() == true) {
        if (ascend.isSelected()) {
            Burbuja burbuja = new Burbuja(labelpasos, ordenadox, ordenadoy, jPanel1, encabezadol, titulo, jLabe
            burbuja.start();
        } else if (descend.isSelected() == true) {
            Burbuja burbuja = new Burbuja(labelpasos, ordenadox, ordenadoy, jPanel1, encabezadol, titulo, jLabe
            burbuja.start();
        }
        jButton4.setEnabled(true);
        algoritmo = "Ordenamiento por burbuja";
    }
} else {
    JOptionPane.showMessageDialog(this, "No hay un archivo cargado", "Error", JOptionPane.ERROR_MESSAGE);
```

Reportes

Luego de ordenar los datos se habilitará el botón para el reporte, creará un archivo y tendrá el nombre del título de la gráfica.

```
String cwd = System.getProperty("user.dir");
File dir = new File(cwd + "\\Reportes");
FileWriter escribir;
PrintWriter nuevaLinea;
if (!dir.exists() && !dir.isDirectory()) {
    try {
        dir.mkdir();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e);
    }
}
File archivo = new File(cwd + "\\Reportes\\" + txtnombre.getText() + ".html");
archivo.delete();
try {
    archivo.createNewFile();
} catch (IOException ex) {
    Logger.getLogger(Inicio.class.getName()).log(Level.SEVERE, null, ex);
}

try {
    escribir = new FileWriter(archivo, true);
    nuevaLinea = new PrintWriter(escribir);
    nuevaLinea.println("<!DOCTYPE html>\n"
            + "<html lang=\"en\">\n"
            + "<head>\n"
            + "    <meta charset=\"UTF-8\">\n"
            + "    <title>Reporte</title>\n"
            + "</head>\n"
            + "<body>");
```

Luego con un ciclo para irá creando filas con la información de los datos ordenados.

```
    nuevaLinea.println("<h1>Reporte </h1>");
    nuevaLinea.println("<p><b>Nombre: </b>Rubén Ralda</p>");
    nuevaLinea.println("<p><b>Carné: </b>202111835</p>");
    nuevaLinea.println("<p><b>Algoritmo: </b>" + algoritmo + "</p>");
    nuevaLinea.println("<p><b>Tiempo: </b>" + pasos + "</p>");
    nuevaLinea.println("<p><b>Cantidad de pasos: </b>" + pasos + "</p>");
    nuevaLinea.println("<table border=\"1\">");
    for (int i = 0; i < ordenadox.length; i++) {
        if (ordenadox[i] != null) {
            nuevaLinea.println("<tr>");
            nuevaLinea.print("<td>");
            nuevaLinea.print("<b>" + ordenadox[i] + "</b>");
            nuevaLinea.print("</td>");
            nuevaLinea.print("<td>");
            nuevaLinea.print(ordenadoy[i]);
            nuevaLinea.print("</td>");
            nuevaLinea.println("</tr>");
        }
    }
    nuevaLinea.println("</table>");
```

Y lo mismo para los datos no ordenados.

```
nuevaLinea.println("<p><b>Datos no ordenados</b></p>");
nuevaLinea.println("<table border=\"1\">");
for (int i = 0; i < valoresx.length; i++) {
    if (valoresx[i] != null) {
        nuevaLinea.println("<tr>");
        nuevaLinea.print("<td>");
        nuevaLinea.print("<b>" + valoresx[i] + "</b>");
        nuevaLinea.print("</td>");
        nuevaLinea.print("<td>");
        nuevaLinea.print(valoresy[i]);
        nuevaLinea.print("</td>");
        nuevaLinea.println("</tr>");
    }
}
nuevaLinea.println("</table>");
nuevaLinea.println(" ");
nuevaLinea.println("</body>\n"
        + "</html>");
// me cierra mi archivo
escribir.close();
JOptionPane.showMessageDialog(this, "El reporte se ha creado con exito");
```

Cronometro

En la clase cronometro heredo Thread y en su constructor recibe un JLabel para mostrar cada segundo y un atributo tipo boolean para terminar el ciclo de conteo.

```
Cronometro(JLabel inicio) {
    this.inicio = inicio;
    terminar=true;
}

public void setTerminar(boolean terminar) {
    this.terminar = terminar;
}
```

Sobrescribo el método run y mientras “terminar” sea verdadero irá modificando el label y con un sleep de un segundo para hacer el cronometro.

```java
public void reloj() {
    while (terminar==true) {
        if (segundos == 30) {
            segundos = 0;
            minutos++;
        }
        inicio.setText(minutos + ":" + segundos);
        segundos++;
        try {
            TimeUnit.SECONDS.sleep(1);
        } catch (Exception e) {
            System.out.println("Error: " + e);
        }
    }
}

// creacion del hilo
@Override
public void run() {
    reloj();
}
```

Ordenamiento por inserción

Este es el constructor de inserción que recibe como parámetros los datos a ordenar, los label para modificar y el panel para mostrar paso a paso.

```java
public Insercion(JLabel labelpasos, String[] ordenadox, int[] ordenadoy, JPanel jpanel, String[] encabezado,
    this.labelpasos = labelpasos;
    this.ordenarx = ordenadox;
    this.ordenary = ordenadoy;
    this.jpanel = jpanel;
    this.encabezado = encabezado;
    this.titulo = titulo;
    this.crono = crono;
    this.tipo = tipo;
}
```

Sobrescribo el métod run de la clase Thread y dentro creo una instancia de la clase cronometro para el tiempo y en un switch determino si es ascendente o descendente. Se suma cada vez pasos y cambio el label del JFrame, invoco el método mostrar para visualizar los cambios en tiempo real. Al final con le cambio el valor a false para que termine el cronometro.

```
@Override
public void run() {
    int insercion, pasos = 0;
    String insercionx;
    Cronometro reloj = new Cronometro(crono);
    reloj.start();
    switch (tipo) {
        case 0: //ascendente
            for (int siguiente = 1; siguiente < ordenary.length; siguiente++) {
                insercion = ordenary[siguiente];
                insercionx = ordenarx[siguiente];
                int moverElemento = siguiente;
                while (moverElemento > 0 && ordenary[moverElemento - 1] > insercion) {
                    ordenary[moverElemento] = ordenary[moverElemento - 1];
                    ordenarx[moverElemento] = ordenarx[moverElemento - 1];
                    moverElemento--;
                    pasos++;
                    labelpasos.setText(String.valueOf(pasos));
                    try {
                        sleep(100);
                    } catch (Exception e) {
                        System.out.println("Error: " + e);
                    }
                    mostrar();
                }
                ordenary[moverElemento] = insercion;
                ordenarx[moverElemento] = insercionx;
                pasos++;
                labelpasos.setText(String.valueOf(pasos));

                labelpasos.setText(String.valueOf(pasos));
                try {
                    sleep(100);
                } catch (Exception e) {
                    System.out.println("Error: " + e);
                }
                mostrar();
            }
            mostrar();
            break;
        case 1: //descendente
            for (int siguiente = 1; siguiente < ordenary.length; siguiente++) {
                insercion = ordenary[siguiente];
                insercionx = ordenarx[siguiente];
                int moverElemento = siguiente;
                while (moverElemento > 0 && ordenary[moverElemento - 1] < insercion) {
                    ordenary[moverElemento] = ordenary[moverElemento - 1];
                    ordenarx[moverElemento] = ordenarx[moverElemento - 1];
                    moverElemento--;
                    pasos++;
                    labelpasos.setText(String.valueOf(pasos));
                    try {
                        sleep(100);
                    } catch (Exception e) {
                        System.out.println("Error: " + e);
                    }
                    mostrar();
                }
                ordenary[moverElemento] = insercion;
                ordenarx[moverElemento] = insercionx;
                pasos++;
```

Ordenamiento de burbuja

Para el algoritmo de burbuja se hace lo mismo, pero con su respectiva forma de ordenar los datos.

```
public Burbuja(JLabel labelpasos, String[] ordenadox, int[] ordenadoy, JPanel jpanel, String[] encabezado, St
    this.labelpasos = labelpasos;
    this.ordenarx = ordenadox;
    this.ordenary = ordenadoy;
    this.jpanel = jpanel;
    this.encabezado = encabezado;
    this.titulo = titulo;
    this.crono = crono;
    this.tipo = tipo;
}
```

```
@Override
public void run() {
    int n, i, l = ordenary.length, temp, pasos = 0;
    String temp2;
    Cronometro reloj = new Cronometro(crono);
    reloj.start();
    switch (tipo) {
        case 0://ascendente
            do {
                n = 0;
                for (i = 1; i < l; i++) {
                    if (ordenary[i - 1] > ordenary[i]) {
                        temp = ordenary[i - 1];
                        ordenary[i - 1] = ordenary[i];
                        ordenary[i] = temp;
                        temp2 = ordenarx[i - 1];
                        ordenarx[i - 1] = ordenarx[i];
                        ordenarx[i] = temp2;
                        n = i;
                        pasos++;
                        labelpasos.setText(String.valueOf(pasos));
                        try {
                            sleep(100);
                        } catch (Exception e) {
                            System.out.println("Error: " + e);
                        }
                        mostrar();
                    }
                }
                l = n;
                pasos++;
```

```
                pasos++;
                labelpasos.setText(String.valueOf(pasos));
                try {
                    sleep(100);
                } catch (Exception e) {
                    System.out.println("Error: " + e);
                }
                mostrar();
            } while (n != 0);
            break;
        case 1://descendente
            do {
                n = 0;
                for (i = 1; i < l; i++) {
                    if (ordenary[i - 1] < ordenary[i]) {
                        temp = ordenary[i - 1];
                        ordenary[i - 1] = ordenary[i];
                        ordenary[i] = temp;
                        temp2 = ordenarx[i - 1];
                        ordenarx[i - 1] = ordenarx[i];
                        ordenarx[i] = temp2;
                        n = i;
                        pasos++;
                        labelpasos.setText(String.valueOf(pasos));
                        try {
                            sleep(100);
                        } catch (Exception e) {
                            System.out.println("Error: " + e);
                        }
                        mostrar();
                    }
                }
                l = n;
                pasos++;
                labelpasos.setText(String.valueOf(pasos));
                try {
                    sleep(100);
                } catch (Exception e) {
                    System.out.println("Error: " + e);
                }
                mostrar();
            } while (n != 0);
            break;
        default:
            throw new AssertionError();
    }
    reloj.setTerminar(false);
```