```
package cpu;

import Info.About;
import Info.Instruction;
import Items.Cell;
import Items.Job;
import Items.MyTable;
import Items.Queue;

import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;



import java.awt.Color;

import javax.swing.JLabel;

import java.awt.Font;



import java.awt.SystemColor;
import java.awt.Component;









import javax.swing.ScrollPaneConstants;
import javax.swing.border.LineBorder;
import javax.swing.JFrame;
import javax.swing.JButton;

import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
```

```
import javax.swing.JSlider;

import java.awt.Cursor;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.FocusAdapter;
import java.awt.event.FocusEvent;

import com.jgoodies.forms.factories.DefaultComponentFactory;

import javax.swing.SwingConstants;
import javax.swing.border.BevelBorder;
import javax.swing.ButtonGroup;
import javax.swing.JOptionPane;
import javax.swing.JTextArea;
import javax.swing.JScrollPane;
import javax.swing.JRadioButton;


/**
 * Main page of the programs
 */
@SuppressWarnings("serial")
public class Face extends javax.swing.JFrame implements ItemListener {

   /**
    * Creates new frame and start the thread
    */
   public Face() {
         getContentPane().setCursor(Cursor.getPredefinedCursor(Cursor.DEFAULT_CURSOR));
         getContentPane().setBackground(SystemColor.inactiveCaptionText);
      initComponents();
      stopBttn.setEnabled(false);
      restartBttn.setEnabled(false);
      getContentPane().setLayout(null);
      getContentPane().add(jLabel7);
      getContentPane().add(jLabel1);
      getContentPane().add(numOfJobs);
      getContentPane().add(simulateBttn);
      getContentPane().add(anotherSimBttn);
      getContentPane().add(jLabel2);
```

```
        /*getContentPane().add(AlgorithmsMenu);*/
        getContentPane().add(stopBttn);
        getContentPane().add(jLabel3);
        getContentPane().add(simSpeed);
        getContentPane().add(jLabel4);
        getContentPane().add(quantum);
        getContentPane().add(nextStepBttn);
        getContentPane().add(restartBttn);
        getContentPane().add(finishBttn);
        jLabel10 = new javax.swing.JLabel();
        jLabel10.setBounds(20, 135, 104, 27);
        getContentPane().add(jLabel10);

            jLabel10.setFont(new Font("Cambria", Font.BOLD, 24)); // NOI18N
            jLabel10.setForeground(Color.YELLOW);
            jLabel10.setText("RESULTS");
            jScrollPane1 = new javax.swing.JScrollPane();

jScrollPane1.setHorizontalScrollBarPolicy(ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS);
            jScrollPane1.setBounds(20, 164, 751, 192);
            getContentPane().add(jScrollPane1);
            table = new javax.swing.JTable();
            table.setForeground(Color.DARK_GRAY);
            table.setBorder(new LineBorder(new Color(255, 192, 203), 1, true));
            table.setBackground(Color.LIGHT_GRAY);
            table.setFont(new Font("Book Antiqua", Font.PLAIN, 13));


                table.setModel(new javax.swing.table.DefaultTableModel
                (
                   new Object [][]
                   {
                      { new Integer(1), null, null, null, null, null, null, null, null, null, null},
                      { new Integer(2), null, null, null, null, null, null, null, null, null, null},
                      { new Integer(3), null, null, null, null, null, null, null, null, null, null},
                      { new Integer(4), null, null, null, null, null, null, null, null, null, null},
                      { new Integer(5), null, null, null, null, null, null, null, null, null, null},
                      { new Integer(6), null, null, null, null, null, null, null, null, null, null},
                      { new Integer(7), null, null, null, null, null, null, null, null, null, null},
                      {null, null, null, null, null, null, null, null, null, null, null}
                   },
                   new String []
                   {
                      "#", "Arrive", "Burst", "Priority", "Start", "Response", "Wait", "Remain", "Finish",
"Turnaround", "%"
                   }

                ) {
```

```
                @SuppressWarnings("rawtypes")
                                                    Class[] types = new Class [] {
                  java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class,
java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class,
java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class, java.lang.Integer.class
                };
                boolean[] canEdit = new boolean [] {
                  false, false, false, false, false, false, false, false, false, false, false
                };


                                                    public Class getColumnClass(int columnIndex) {
                  return types [columnIndex];
                }

                public boolean isCellEditable(int rowIndex, int columnIndex) {
                  return canEdit [columnIndex];
                }
              });
              jScrollPane1.setViewportView(table);
                                          jLabel11 = new javax.swing.JLabel();
                                          jLabel11.setForeground(Color.YELLOW);
                                          jLabel11.setBounds(45, 599, 70, 27);
                                          getContentPane().add(jLabel11);

                                              jLabel11.setFont(new Font("Dialog", Font.PLAIN,
18)); // NOI18N

jLabel11.setHorizontalAlignment(SwingConstants.LEFT);
                                                jLabel11.setText("Waiting");
                                                AverWait = new javax.swing.JLabel();
                                                AverWait.setForeground(Color.YELLOW);
                                                AverWait.setBounds(45, 637, 65, 27);
                                                getContentPane().add(AverWait);

                                                    AverWait.setBackground(new
java.awt.Color(255, 255, 255));
                                                    AverWait.setFont(new Font("Dialog",
Font.PLAIN, 18)); // NOI18N

AverWait.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
                                                AverWait.setText("0");
                                                jLabel12 = new javax.swing.JLabel();
                                                jLabel12.setForeground(Color.YELLOW);
                                                jLabel12.setBounds(183, 599, 104, 27);
                                                getContentPane().add(jLabel12);
```

```
                                                    jLabel12.setFont(new Font("Dialog",
Font.PLAIN, 18)); // NOI18N

jLabel12.setHorizontalAlignment(SwingConstants.LEFT);
                                                    jLabel12.setText("Turnaround");
                                                    AverTurn = new javax.swing.JLabel();
                                                    AverTurn.setForeground(Color.YELLOW);
                                                    AverTurn.setBounds(183, 625, 92, 50);
                                                    getContentPane().add(AverTurn);

                                                        AverTurn.setFont(new Font("Dialog",
Font.PLAIN, 18)); // NOI18N

AverTurn.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
                                                        AverTurn.setText("0");

                                                        JLabel lblReadyQueue = new JLabel();
                                                        lblReadyQueue.setBounds(532, 552,
239, 40);
                                                        lblReadyQueue.setText("READY
QUEUE");

lblReadyQueue.setForeground(Color.YELLOW);
                                                        lblReadyQueue.setFont(new
Font("Cambria", Font.BOLD, 24));

getContentPane().add(lblReadyQueue);

                                                        JLabel lblGanttChart = new JLabel();
                                                        lblGanttChart.setBounds(30, 367,
239, 50);
                                                        lblGanttChart.setText("GANTT
CHART");

lblGanttChart.setForeground(Color.YELLOW);
                                                        lblGanttChart.setFont(new
Font("Cambria", Font.BOLD, 24));
                                                        getContentPane().add(lblGanttChart);

                                                        btnAddData = new JButton("Add
Data");

btnAddData.setForeground(Color.GREEN);
                                                        btnAddData.setFont(new
Font("Cambria", Font.BOLD, 18));
                                                        btnAddData.setBackground(new
Color(105, 105, 105));
```

```
btnAddData.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
                                                        btnAddData.addActionListener(new
ActionListener() {
                                                            public void
actionPerformed(ActionEvent evt) {

        jMenuItem1ActionPerformed(evt);
                                                            }
                                                        });
                                                        btnAddData.setBounds(802, 164,
499, 33);
                                                        getContentPane().add(btnAddData);



lblInstructions.addMouseListener(new MouseAdapter() {
                                                            @Override
                                                            public void
mouseClicked(MouseEvent evt) {

        jMenuItem4ActionPerformed(evt);
                                                            }
                                                        });

lblInstructions.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));

lblInstructions.setForeground(Color.ORANGE);
                                                        lblInstructions.setFont(new
Font("Cambria", Font.BOLD, 17));
                                                        lblInstructions.setBounds(1052, 85,
149, 27);

getContentPane().add(lblInstructions);

                                                        lblInstructions.addFocusListener(new
FocusAdapter() {
                                                            @Override
                                                            public void
focusGained(FocusEvent evt) {
                                                                //Frame lblInstruction =
null;

                                                lblInstructions.setForeground(Color.YELLOW);
                                                            }
                                                        });
```

```
                                                        lblExit = new JLabel("EXIT");
                                                        lblExit.addMouseListener(new
MouseAdapter() {
                                                            @Override
                                                            public void
mouseClicked(MouseEvent evt) {

        jMenuItem3ActionPerformed(evt);
                                                            }
                                                        });

lblExit.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));

lblExit.setForeground(Color.ORANGE);
                                                        lblExit.setFont(new Font("Cambria",
Font.BOLD, 17));
                                                        lblExit.setBounds(1249, 88, 45, 20);
                                                        getContentPane().add(lblExit);

                                                        lblCpuSchedulingSimulator =
DefaultComponentFactory.getInstance().createTitle("CPU Scheduling Simulator");

lblCpuSchedulingSimulator.setForeground(new Color(100, 149, 237));

lblCpuSchedulingSimulator.setFont(new Font("PhrasticMedium", Font.PLAIN, 47));

lblCpuSchedulingSimulator.setHorizontalAlignment(SwingConstants.CENTER);

lblCpuSchedulingSimulator.setBounds(468, 11, 524, 50);

getContentPane().add(lblCpuSchedulingSimulator);

                                                        lblNewLabel = new JLabel("");
                                                        lblNewLabel.setOpaque(true);

lblNewLabel.setBackground(SystemColor.activeCaptionBorder);
                                                        lblNewLabel.setBounds(10, 74, 1334,
6);
                                                        getContentPane().add(lblNewLabel);

                                                        lblUtilization = new JLabel();
                                                        lblUtilization.setText("UTILIZATION");

lblUtilization.setForeground(Color.YELLOW);
                                                        lblUtilization.setFont(new
Font("Cambria", Font.BOLD, 24));
```

```
                                                        lblUtilization.setBounds(330, 552,
157, 40);
                                                        getContentPane().add(lblUtilization);

                                                        cpuUtilize = new JLabel();
                                                        cpuUtilize.setText("0%");

cpuUtilize.setHorizontalAlignment(SwingConstants.CENTER);

cpuUtilize.setForeground(Color.YELLOW);
                                                        cpuUtilize.setFont(new Font("Dialog",
Font.PLAIN, 18));
                                                        cpuUtilize.setBounds(355, 625, 92,
50);
                                                        getContentPane().add(cpuUtilize);

                                                        JTextArea txtrHaroldTAsuncion = new
JTextArea();

txtrHaroldTAsuncion.setOpaque(false);
                                                        txtrHaroldTAsuncion.setBorder(null);
                                                        txtrHaroldTAsuncion.setFont(new
Font("Dotum", Font.BOLD, 14));

txtrHaroldTAsuncion.setForeground(Color.PINK);

txtrHaroldTAsuncion.setBackground(SystemColor.inactiveCaptionText);

txtrHaroldTAsuncion.setEditable(false);
                                                        txtrHaroldTAsuncion.setText("Harold
T. Asuncion\r\nMichael Joshua G. Eresuela\r\nGladys T. Obmerga\r\nPinky S. Pal-lingayan\r\n\r\nBSCpE
4-3");
                                                        txtrHaroldTAsuncion.setBounds(833,
567, 229, 109);

getContentPane().add(txtrHaroldTAsuncion);

                                                        lblSubmittedBy = new JLabel();
                                                        lblSubmittedBy.setText("SUBMITTED
BY:");

lblSubmittedBy.setForeground(Color.YELLOW);
                                                        lblSubmittedBy.setFont(new
Font("Cambria", Font.BOLD, 20));
                                                        lblSubmittedBy.setBounds(823, 536,
176, 24);

getContentPane().add(lblSubmittedBy);
```

```
                                                    lblSubmittedTo = new JLabel();
                                                    lblSubmittedTo.setText("SUBMITTED
TO:");
lblSubmittedTo.setForeground(Color.YELLOW);
                                                    lblSubmittedTo.setFont(new
Font("Cambria", Font.BOLD, 20));
                                                    lblSubmittedTo.setBounds(1106, 535,
155, 27);
getContentPane().add(lblSubmittedTo);

                                                    txtrEngrJuliusCansino = new
JTextArea();
                                                    txtrEngrJuliusCansino.setBorder(null);
                                                    txtrEngrJuliusCansino.setText("Engr.
Julius Cansino");
txtrEngrJuliusCansino.setForeground(Color.PINK);
                                                    txtrEngrJuliusCansino.setFont(new
Font("Dotum", Font.BOLD, 15));
txtrEngrJuliusCansino.setEditable(false);
txtrEngrJuliusCansino.setBackground(SystemColor.inactiveCaptionText);
txtrEngrJuliusCansino.setBounds(1138, 595, 160, 35);
getContentPane().add(txtrEngrJuliusCansino);

                                                    lblNewLabel_1 = new JLabel("");
                                                    lblNewLabel_1.setBorder(new
LineBorder(SystemColor.activeCaptionBorder, 3, true));
                                                    lblNewLabel_1.setBounds(793, 529,
551, 147);
getContentPane().add(lblNewLabel_1);
                                                    FCFS.setSelected(false);


                                                    FCFS.setForeground(Color.YELLOW);
FCFS.setBackground(SystemColor.inactiveCaptionText);
                                                    FCFS.setBounds(808, 234, 208, 23);
                                                    getContentPane().add(FCFS);
```

```
                                                    SJF.setForeground(Color.YELLOW);
SJF.setBackground(SystemColor.inactiveCaptionText);
                                                    SJF.setBounds(808, 260, 208, 23);
                                                    getContentPane().add(SJF);

                                                    STRF.setForeground(Color.YELLOW);
STRF.setBackground(SystemColor.inactiveCaptionText);
                                                    STRF.setBounds(808, 286, 208, 23);
                                                    getContentPane().add(STRF);

                                                    RR.setForeground(Color.YELLOW);
RR.setBackground(SystemColor.inactiveCaptionText);
                                                    RR.setBounds(808, 312, 208, 23);
                                                    getContentPane().add(RR);


Priority1.setForeground(Color.YELLOW);
Priority1.setBackground(SystemColor.inactiveCaptionText);
                                                    Priority1.setBounds(808, 336, 208,
23);
                                                    getContentPane().add(Priority1);


Priority2.setForeground(Color.YELLOW);
Priority2.setBackground(SystemColor.inactiveCaptionText);
                                                    Priority2.setBounds(808, 358, 214,
23);
                                                    getContentPane().add(Priority2);

                                                    group.add(FCFS);
                                                        group.add(SJF);
                                                        group.add(STRF);
                                                        group.add(RR);
                                                        group.add(Priority1);
                                                        group.add(Priority2);

                                                        desctext = new JLabel();
        desctext.setVerticalAlignment(SwingConstants.TOP);
```

```
        desctext.setForeground(Color.YELLOW);
                                                                desctext.setFont(new
Font("Dialog", Font.PLAIN, 12));
                                                                desctext.setAlignmentX(0.5f);
                                                                desctext.setBounds(1028, 234,
273, 147);
        getContentPane().add(desctext);


                                                                lblDescription = new JLabel();

        lblDescription.setText("Description");
        lblDescription.setForeground(Color.YELLOW);
                                                                lblDescription.setFont(new
Font("Dialog", Font.PLAIN, 18));
                                                                lblDescription.setBounds(1010,
207, 151, 20);
        getContentPane().add(lblDescription);


                                                                label = new JLabel("");
                                                                label.setBorder(new
LineBorder(SystemColor.activeCaptionBorder, 3, true));
                                                                label.setBounds(793, 153, 551,
365);
                                                                getContentPane().add(label);

                                                                lblAverage = new JLabel();
                                                                lblAverage.setText("AVERAGE");


        lblAverage.setForeground(Color.YELLOW);
                                                                lblAverage.setFont(new
Font("Cambria", Font.BOLD, 24));
                                                                lblAverage.setBounds(104, 552,
125, 40);
        getContentPane().add(lblAverage);


                                                                lblLegend = new
JLabel("LEGEND:");
                                                                lblLegend.setFont(new
Font("Dialog", Font.PLAIN, 13));
        lblLegend.setForeground(Color.YELLOW);
                                                                lblLegend.setBounds(34, 465,
62, 35);
```

```
        getContentPane().add(lblLegend);

                                                label_3 = new JLabel("1");
                                                label_3.setOpaque(true);
                                                label_3.setBackground(new
Color(0xdb353d));
        label_3.setHorizontalAlignment(SwingConstants.CENTER);
                                                label_3.setBounds(92, 494, 45,
27);
                                                getContentPane().add(label_3);

                                                label_5 = new JLabel("2");

        label_5.setBackground(Color.DARK_GRAY);
                                                label_5.setOpaque(true);

        label_5.setHorizontalAlignment(SwingConstants.CENTER);
                                                label_5.setBounds(147, 494, 45,
27);
                                                getContentPane().add(label_5);

                                                label_6 = new JLabel("3");
                                                label_6.setBackground(new
Color(0xdb35c2));
                                                label_6.setOpaque(true);

        label_6.setHorizontalAlignment(SwingConstants.CENTER);
                                                label_6.setBounds(202, 494, 45,
27);
                                                getContentPane().add(label_6);

                                                label_7 = new JLabel("4");
                                                label_7.setBackground(new
Color(0x3935db));
                                                label_7.setOpaque(true);

        label_7.setHorizontalAlignment(SwingConstants.CENTER);
                                                label_7.setBounds(253, 494, 45,
27);
                                                getContentPane().add(label_7);

                                                label_8 = new JLabel("5");
                                                label_8.setBackground(new
Color(0x35a6db));
                                                label_8.setOpaque(true);
```

```
        label_8.setHorizontalAlignment(SwingConstants.CENTER);
                                                                label_8.setBounds(304, 494, 45,
27);
                                                                getContentPane().add(label_8);

                                                                label_9 = new JLabel("6");
                                                                label_9.setBackground(new
Color(0x35dbbe));
                                                                label_9.setOpaque(true);

        label_9.setHorizontalAlignment(SwingConstants.CENTER);
                                                                label_9.setBounds(355, 494, 45,
27);
                                                                getContentPane().add(label_9);

                                                                label_10 = new JLabel("7");
                                                                label_10.setBackground(new
Color(0x35db39));
                                                                label_10.setOpaque(true);

        label_10.setHorizontalAlignment(SwingConstants.CENTER);
                                                                label_10.setBounds(406, 494,
45, 27);
        getContentPane().add(label_10);

                                                                label_11 = new JLabel("8");
                                                                label_11.setBackground(new
Color(0x8fdb35));
                                                                label_11.setOpaque(true);

        label_11.setHorizontalAlignment(SwingConstants.CENTER);
                                                                label_11.setBounds(457, 494,
45, 27);
        getContentPane().add(label_11);

                                                                label_12 = new JLabel("9");
                                                                label_12.setBackground(new
Color(0xdbd935));
                                                                label_12.setOpaque(true);

        label_12.setHorizontalAlignment(SwingConstants.CENTER);
                                                                label_12.setBounds(508, 494,
45, 27);
        getContentPane().add(label_12);
```

```
                                                label_13 = new JLabel("10");
                                                label_13.setBackground(new
Color(0xdb8735));
                                                label_13.setOpaque(true);
        label_13.setHorizontalAlignment(SwingConstants.CENTER);
                                                label_13.setBounds(559, 494,
45, 27);
        getContentPane().add(label_13);
                                                lblEmptyJob = new
JLabel("Empty Job");
        lblEmptyJob.setBackground(Color.WHITE);
                                                lblEmptyJob.setOpaque(true);
        lblEmptyJob.setHorizontalAlignment(SwingConstants.CENTER);
                                                lblEmptyJob.setBounds(610,
494, 82, 27);
        getContentPane().add(lblEmptyJob);

                                                label_1 = new JLabel("");
                                                label_1.setBorder(new
LineBorder(SystemColor.activeCaptionBorder, 3, true));
                                                label_1.setBounds(10, 153, 773,
523);
                                                getContentPane().add(label_1);

                                                label_2 = new JLabel("");
                                                label_2.setBorder(new
LineBorder(SystemColor.activeCaptionBorder, 3, true));
                                                label_2.setBounds(20, 536, 738,
5);
                                                getContentPane().add(label_2);

                                                label_4 = new JLabel("");
                                                label_4.setBorder(new
LineBorder(SystemColor.activeCaptionBorder, 3, true));
                                                label_4.setBounds(20, 376, 751,
5);
                                                getContentPane().add(label_4);

                                                FCFS.addItemListener(this);
                                                SJF.addItemListener(this);
                                                STRF.addItemListener(this);
```

```
                                                                RR.addItemListener(this);
                                                                Priority1.addItemListener(this);
                                                                Priority2.addItemListener(this);
            table.getColumnModel().getColumn(0).setResizable(false);
            table.getColumnModel().getColumn(1).setResizable(false);
            table.getColumnModel().getColumn(2).setResizable(false);
            table.getColumnModel().getColumn(3).setResizable(false);
            table.getColumnModel().getColumn(4).setResizable(false);
            table.getColumnModel().getColumn(5).setResizable(false);
            table.getColumnModel().getColumn(6).setResizable(false);
            table.getColumnModel().getColumn(7).setResizable(false);
            table.getColumnModel().getColumn(8).setResizable(false);
            table.getColumnModel().getColumn(9).setResizable(false);
            table.getColumnModel().getColumn(10).setResizable(false);
   setLocationRelativeTo(null);
   MainQueue.createNew(10);  // create random data main queue
   updateTable(MainQueue.get()); // view initial jobs data in the table
   thread.start();
}

/**
 * set assigned jobs by the user to the Main queue and update the table
 * @param queue queue of jobs
 */
public void setAssignedQueue(Queue queue)
{
   MainQueue.add(queue);
   updateTable(MainQueue.get());
}

private MyTable myTable;  // table on the GUI
/**
 * update table data
 * @param queue queue of jobs to be shown on the table
 */
private void updateTable(Queue queue)
{
   myTable = new MyTable(queue);
   table.setModel(myTable);
}

// <editor-fold defaultstate="collapsed" desc="Visuals" >

/**
 * view CPU, Gantt and readyQueue visuals
 * @param job current job processed in the simulation
 * @param readyQueue  current ready queue in the simulation
 */
```

```
private void viewVisuals(Job job , Queue readyQueue)
{
   cpuVisual(job, Simulation.Time);
   showGantt(job, Simulation.Time);
   showReadyQueue(readyQueue);
}

/**
 * clear and reset CPU, Gantt and ReadyQueue visuals
 */
private void clearVisuals()
{
   cpuClear();
   clearGantt();
   clearReadyQueue();
}
// </editor-fold>

// <editor-fold defaultstate="collapsed" desc="cpu visual" >

private static int idleTime =0; // CPU idle time
/**
 * view current data of the CPU during the simulation
 * @param job current job to be processed by the CPU
 * @param time current time of the simulation
 */
private void cpuVisual(Job job , int time){
  if(job == null )
   {
      idleTime++;
   }

   if(time != 0)
   {
      cpuUtilize.setText(((time - idleTime) *100 / time) +"%");
   }
   else { cpuUtilize.setText(100 +"%"); }
}

/**
 * reset CPU visual data
 */
private void cpuClear(){
   cpuUtilize.setText("0%");
   idleTime = 0;
}

/**
```

```
 * update and show Gantt chart
 * @param job job to be added to the chart
 * @param time time of the simulation
 */
private void showGantt(Job job , int time){
  GanttChart.addJob(job, time);
  addToGUI(GanttChart.List);
}

/**
 * remove all the elements from the Gantt chart
 */
private void clearGantt(){
   removeFromGUI(GanttChart.List);
   GanttChart.clear();
}

/**
 * update and show ready queue chart
 * @param list list of jobs to be represented
 */
private void showReadyQueue(Queue list){
   clearReadyQueue();
   ReadyChart.update(list);
   addToGUI(ReadyChart.List);
}

/**
 * remove all the elements form the ready queue chart
 */
private void clearReadyQueue(){
   removeFromGUI(ReadyChart.List);
   ReadyChart.clear();
 }

/**
 * add list of cell elements to the GUI
 * @param list of element from the charts
 */
private void addToGUI (ArrayList<Cell> list)
{
   for(int i =0 ; i< list.size() ; i++)
   {
     getContentPane().add(list.get(i));
   }
   repaint();
}
```

```
/**
 * remove list of cell elements from the GUI
 * @param list list of elements to be removed from the GUI
 */
private void removeFromGUI ( ArrayList<Cell> list)
{
   for(int i =0 ; i< list.size() ; i++)
   {
     remove(list.get(i));
   }
   repaint();
}

Thread thread = new Thread(new Runnable (){
   @Override
   public void run () {
   while(true)
   {
      if(!Simulation.Finished && !Simulation.Stoped) // stops the simulation
      {
       nextStepBttnActionPerformed(null);  // press next step button
       delay();  // delay time after every step
      }
   }
}});

/**
 * responsible for the delay time between every step in the
 * simulation.
 */
public void delay ()
{
   int num = Integer.parseInt(simSpeed.getSelectedItem()+"");  // get the delay factor from GUI
   try {
      Thread.sleep(150 * num); // 150 is here by try and error
   }
   catch (InterruptedException ex) {
      Logger.getLogger(Face.class.getName()).log(Level.SEVERE, null, ex);
   }
}
// </editor-fold>

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
```

```
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {


        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("CPU Scheduling");
        setMinimumSize(new java.awt.Dimension(1370, 725));

        jLabel7 = new javax.swing.JLabel();
        jLabel7.setHorizontalAlignment(SwingConstants.CENTER);
        jLabel7.setAlignmentX(Component.CENTER_ALIGNMENT);
        jLabel7.setBounds(778, 123, 254, 50);

            jLabel7.setFont(new Font("Cambria", Font.BOLD, 24)); // NOI18N
            jLabel7.setForeground(Color.YELLOW);
            jLabel7.setText("SCHEDULING DATA");
        jLabel1 = new javax.swing.JLabel();
        jLabel1.setForeground(Color.YELLOW);
        jLabel1.setAlignmentX(Component.CENTER_ALIGNMENT);
        jLabel1.setBounds(813, 406, 186, 20);

            jLabel1.setFont(new Font("Dialog", Font.PLAIN, 18)); // NOI18N
            jLabel1.setText("Number of Process");
        numOfJobs = new javax.swing.JComboBox();
        numOfJobs.setBounds(997, 401, 76, 35);

            numOfJobs.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
            numOfJobs.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "1", "2", "3", "4",
"5", "6", "7", "8", "9","10" }));
            numOfJobs.setSelectedIndex(0);
            numOfJobs.setLightWeightPopupEnabled(false);
            numOfJobs.setMaximumSize(new java.awt.Dimension(30, 20));
            numOfJobs.setMinimumSize(new java.awt.Dimension(30, 20));
            numOfJobs.setPreferredSize(new java.awt.Dimension(30, 20));
            numOfJobs.addActionListener(new java.awt.event.ActionListener() {
               public void actionPerformed(java.awt.event.ActionEvent evt) {
                  numOfJobsActionPerformed(evt);
               }
            });
        simulateBttn =new javax.swing.JButton();
        simulateBttn.setForeground(Color.GREEN);
        simulateBttn.setBackground(new Color(105, 105, 105));
        simulateBttn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
        simulateBttn.setBounds(1099, 395, 117, 20);

            simulateBttn.setFont(new Font("Cambria", Font.BOLD, 18)); // NOI18N
            simulateBttn.setText("Simulate");
            simulateBttn.setMaximumSize(new java.awt.Dimension(85, 25));
```

```
        simulateBttn.setMinimumSize(new java.awt.Dimension(85, 25));
        simulateBttn.setPreferredSize(new java.awt.Dimension(85, 25));
        simulateBttn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                simulateBttnActionPerformed(evt);
            }
        });
    anotherSimBttn = new javax.swing.JButton();
    anotherSimBttn.setForeground(Color.GREEN);
    anotherSimBttn.setBackground(new Color(105, 105, 105));
    anotherSimBttn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
    anotherSimBttn.setBounds(1099, 452, 231, 27);

        anotherSimBttn.setFont(new Font("Cambria", Font.BOLD, 17)); // NOI18N
        anotherSimBttn.setText("Start another simulation");
        anotherSimBttn.setMaximumSize(new java.awt.Dimension(185, 25));
        anotherSimBttn.setMinimumSize(new java.awt.Dimension(185, 25));
        anotherSimBttn.setPreferredSize(new java.awt.Dimension(185, 25));
        anotherSimBttn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                anotherSimBttnActionPerformed(evt);
            }
        });
    jLabel2 = new javax.swing.JLabel();
    jLabel2.setForeground(Color.YELLOW);
    jLabel2.setBounds(802, 207, 151, 20);

        jLabel2.setFont(new Font("Dialog", Font.PLAIN, 18));
        jLabel2.setText("Type of Algorithm");

    stopBttn = new javax.swing.JButton();
    stopBttn.setForeground(Color.GREEN);
    stopBttn.setBackground(new Color(105, 105, 105));
    stopBttn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
    stopBttn.setBounds(1226, 395, 104, 20);

        stopBttn.setFont(new Font("Cambria", Font.BOLD, 18)); // NOI18N
        stopBttn.setText("Stop");
        stopBttn.setMaximumSize(new java.awt.Dimension(85, 25));
        stopBttn.setMinimumSize(new java.awt.Dimension(85, 25));
        stopBttn.setPreferredSize(new java.awt.Dimension(85, 25));
        stopBttn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                stopBttnActionPerformed(evt);
            }
        });
    jLabel3 = new javax.swing.JLabel();
    jLabel3.setForeground(Color.YELLOW);
```

```
        jLabel3.setBounds(813, 452, 110, 27);

            jLabel3.setFont(new Font("Dialog", Font.PLAIN, 18)); // NOI18N
            jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
            jLabel3.setText("Speed");
        simSpeed = new javax.swing.JComboBox();
        simSpeed.setBounds(823, 483, 105, 20);

            simSpeed.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
            simSpeed.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "1", "2", "3", "4",
"5", "6", "7" }));
            simSpeed.setSelectedIndex(0);
            simSpeed.setMaximumSize(new java.awt.Dimension(65, 20));
            simSpeed.setMinimumSize(new java.awt.Dimension(65, 20));
            simSpeed.setPreferredSize(new java.awt.Dimension(65, 20));
        jLabel4 = new javax.swing.JLabel();
        jLabel4.setForeground(Color.YELLOW);
        jLabel4.setBounds(964, 447, 125, 27);

            jLabel4.setFont(new Font("Dialog", Font.PLAIN, 18)); // NOI18N
            jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
            jLabel4.setText("Time Slice");
        quantum = new javax.swing.JComboBox();
        quantum.setBounds(977, 483, 96, 20);

            quantum.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
            quantum.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "2", "3", "4", "5",
"6", "7", "8" }));
        nextStepBttn = new javax.swing.JButton();
        nextStepBttn.setForeground(Color.GREEN);
        nextStepBttn.setBackground(new Color(105, 105, 105));
        nextStepBttn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
        nextStepBttn.setBounds(1099, 426, 117, 21);

            nextStepBttn.setFont(new Font("Cambria", Font.BOLD, 18)); // NOI18N
            nextStepBttn.setText("Next step");
            nextStepBttn.setMaximumSize(new java.awt.Dimension(95, 25));
            nextStepBttn.setMinimumSize(new java.awt.Dimension(95, 25));
            nextStepBttn.setPreferredSize(new java.awt.Dimension(95, 25));
            nextStepBttn.addActionListener(new java.awt.event.ActionListener() {
               public void actionPerformed(java.awt.event.ActionEvent evt) {
                  nextStepBttnActionPerformed(evt);
               }
            });
        restartBttn = new javax.swing.JButton();
        restartBttn.setForeground(Color.GREEN);
        restartBttn.setBackground(new Color(105, 105, 105));
        restartBttn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
```

```
    restartBttn.setBounds(1226, 426, 104, 21);

        restartBttn.setFont(new Font("Cambria", Font.BOLD, 18)); // NOI18N
        restartBttn.setText("Restart");
        restartBttn.setMaximumSize(new java.awt.Dimension(80, 25));
        restartBttn.setMinimumSize(new java.awt.Dimension(80, 25));
        restartBttn.setPreferredSize(new java.awt.Dimension(80, 25));
        restartBttn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                restartBttnActionPerformed(evt);
            }
        });
    finishBttn = new javax.swing.JButton();
    finishBttn.setForeground(Color.GREEN);
    finishBttn.setBackground(new Color(105, 105, 105));
    finishBttn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
    finishBttn.setBounds(1099, 483, 231, 27);

        finishBttn.setFont(new Font("Cambria", Font.BOLD, 18)); // NOI18N
        finishBttn.setText("Finish");
        finishBttn.setMaximumSize(new java.awt.Dimension(80, 25));
        finishBttn.setMinimumSize(new java.awt.Dimension(80, 25));
        finishBttn.setPreferredSize(new java.awt.Dimension(80, 25));
        finishBttn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                finishBttnActionPerformed(evt);
            }
        });

    pack();
}

/**
 * Event triggered when number of jobs menu is changed.
 * it updates the MainQueue and the table data
 * @param evt
 */
private void numOfJobsActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int num = Integer.parseInt(numOfJobs.getSelectedItem()+"");
    MainQueue.createNew(num);
    updateTable(MainQueue.get());
    //setMainQ();
}

/**
 * performs a step in the simulation
 * @param evt
```

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

        if(group.isSelected(null))
                {
                        JOptionPane.showMessageDialog(null, "Choose an algorithm");
                }
        else{
        if(!Simulation.Finished)
   {
      numOfJobs.setEnabled(false);
      quantum.setEnabled(false);
      restartBttn.setEnabled(true);
      FCFS.setEnabled(false);
      STRF.setEnabled(false);
      SJF.setEnabled(false);
      RR.setEnabled(false);
      Priority1.setEnabled(false);
      Priority2.setEnabled(false);
      Job job = Simulation.workStep();
      viewVisuals(job, Simulation.getReadyQueue());
   }
   if(Simulation.Finished){finishBttnActionPerformed(null);}
   String t1 = myTable.getAverageWaiting() + "";
   String t2 = myTable.getAverageTurn() + "";
   if(t1.length() > 5) { t1 = t1.substring(0, 5);} // set max length to 5
   if(t2.length() > 5) { t2 = t2.substring(0, 5);}
   AverWait.setText(t1);
   AverTurn.setText(t2);
   Simulation.Time++;}
}

/**
 * stops the simulation.
 * also disables stop button and enables next step
 * button and simulate button.
 * @param evt
 */
private void stopBttnActionPerformed(java.awt.event.ActionEvent evt) {
   // TODO add your handling code here:
   Simulation.Stoped = true; // stop the simulation
   stopBttn.setEnabled(false);
   nextStepBttn.setEnabled(true);
   simulateBttn.setEnabled(true);

}
```

```
/**
 * restart simulation with same data
 * @param evt
 */
private void restartBttnActionPerformed(java.awt.event.ActionEvent evt) {
   // TODO add your handling code here:
   stopBttnActionPerformed(null);
   restartBttn.setEnabled(false);
   nextStepBttn.setEnabled(true);
   numOfJobs.setEnabled(true);
   quantum.setEnabled(true);
   AlgorithmsMenu.setEnabled(true);
   simulateBttn.setEnabled(true);
   finishBttn.setEnabled(true);
   MainQueue.reset();
   updateTable(MainQueue.get());
   Simulation.reset();
   AverWait.setText("0");
   AverTurn.setText("0");
   clearVisuals();
}

/**
 * start another simulation with another random data
 * @param evt
 */
private void anotherSimBttnActionPerformed(java.awt.event.ActionEvent evt) {
   // TODO add your handling code here:
   // adjust buttons
   stopBttnActionPerformed(null);
   restartBttn.setEnabled(false);
   numOfJobs.setEnabled(true);
   AlgorithmsMenu.setEnabled(true);
   quantum.setEnabled(true);
   nextStepBttn.setEnabled(true);
   simulateBttn.setEnabled(true);
   finishBttn.setEnabled(true);
   // reset average wait and turnaround
   AverWait.setText("0");
   AverTurn.setText("0");

   numOfJobsActionPerformed(null);
   Simulation.reset();  // reset simulation
   clearVisuals(); // reset CPU, Gantt and readyQueue view
}

/**
 * starts the simulation.
```

```
 * @param evt
 */
private void simulateBttnActionPerformed(java.awt.event.ActionEvent evt) {
   // TODO add your handling code here:
        if(group.isSelected(null))
                {
                        JOptionPane.showMessageDialog(null, "Choose an algorithm");
                }
        else{
   Simulation.Stoped = false;  // start the simulation
   simulateBttn.setEnabled(false);
   stopBttn.setEnabled(true);
   nextStepBttn.setEnabled(false);}
}

/**
 * completes the simulation to the end immediately
 * @param evt
 */
private void finishBttnActionPerformed(java.awt.event.ActionEvent evt) {
   // TODO add your handling code here:
   while(!Simulation.Finished)
   {
      nextStepBttnActionPerformed(null);  // press next step button till the simluation is finished
   }
   // modify the buttons view
   stopBttnActionPerformed(null);
   stopBttn.setEnabled(false);
   finishBttn.setEnabled(false);
   nextStepBttn.setEnabled(false);
   simulateBttn.setEnabled(false);
}

/**
 * view instructions frame
 * @param evt
 */
private void jMenuItem4ActionPerformed(MouseEvent evt) {
   // TODO add your handling code here:
   new Instruction().show();
}

/**
 * exit the program
 * @param evt
 */
private void jMenuItem3ActionPerformed(MouseEvent evt) {
   // TODO add your handling code here:.
```

```
    System.exit(0);
  }

  /**
   * view the frame to allow the user to add jobs data manually
   * @param evt
   */
  @SuppressWarnings("deprecation")
        private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    AddData ad = new AddData();
    ad.show();
    this.hide();
  }

  /**
   * Event triggered when Algorithms menu is changed.
   * it changes the algorithm type in the Simulation class
   * @param evt
   */
  @Override
  public void itemStateChanged(ItemEvent e) {
                // TODO Auto-generated method stub

                 Simulation.AlgorithmType = (JRadioButton)e.getSource();

                JRadioButton src = (JRadioButton)e.getSource();

                if(src == FCFS )
                {
                        desctext.setText("<html>By far the simplest CPU-scheduling algorithm is "
                                        + "the first-come, first-served(FCFS) scheduling algorithm. "
                                        +" the process that requests the "
                                        + "CPU first is allocated the CPU first. The implementation is "
                                        + "easily managed with a FIFO queue. </html>" );
                }
                else if (src == SJF)
                {
                        desctext.setText("<html>This algorithm associates with each process the length
of the "
                                        + "process's next CPU burst. When the CPU is available, it is
assigned to the process that has the smallest next CPU burst. If the next CPU bursts of two "
                                        + "processes are the same, FCFS scheduling is used to break the
tie. </html>");
                }
                else if (src == STRF)
                {
```

```
                desctext.setText("<html>Shortest remaining time first (SRTF), is a scheduling
method that is a preemptive version of shortest job next scheduling. "
                        + "In this scheduling algorithm, the process with the smallest
amount of time remaining until completion is selected to execute. "
                        + "Since the currently executing process is the one with the
shortest amount of time remaining by definition, and "
                        + "since that time should only reduce as execution progresses,
processes will always run until they complete or a new process"
                        + " is added that requires a smaller amount of time.</html>");
            }
            else if (src == RR)
            {
                desctext.setText("<html>Round-robin (RR) is one of the algorithms employed by
process and network schedulers in computing."
                        + " As the term is generally used, time slices are assigned to
each process in equal portions and in circular order, "
                        + "handling all processes without priority (also known as cyclic
executive). Round-robin scheduling is simple, "
                        + "easy to implement, and starvation-free. Round-robin
scheduling can also be applied to other scheduling problems, "
                        + "such as data packet scheduling in computer networks. It is an
Operating System concept.</html>");
            }
            else if (src == Priority1)
            {
                desctext.setText("<html>The operating system assigns a fixed priority rank to
every process, "
                        + "and the scheduler arranges the processes in the ready queue
in order of their priority. "
                        + "Lower-priority processes get interrupted by incoming higher-
priority processes..</html>");
            }
            else if (src == Priority2)
            {
                desctext.setText("<html>Every process has its fixed priority rank and the
scheduler arranges the process in order of "
                        + "their priority.But under non-preemptive scheduling, each
running"
                        + "process keeps the CPU until it completes or it"
                        + "switches to the waiting (blocked) state. Process does not get
interrupted by incoming higher-priority process.</html>");
            }



        }
    public static void main(String args[]) {
```

```
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(Face.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(Face.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(Face.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Face.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        }


        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            @Override
            public void run() {
                new Face().setVisible(true);

            }
        });
    }

    JLabel lblInstructions = new JLabel("USER MANUAL");
    private javax.swing.JComboBox AlgorithmsMenu;
    public static javax.swing.JLabel AverTurn;
    public static javax.swing.JLabel AverWait;
    private javax.swing.JButton anotherSimBttn;
    private javax.swing.JButton finishBttn;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel7;
```

```
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JButton nextStepBttn;
private javax.swing.JComboBox numOfJobs;
private javax.swing.JComboBox quantum;
private javax.swing.JButton restartBttn;
private javax.swing.JComboBox simSpeed;
private javax.swing.JButton simulateBttn;
private javax.swing.JButton stopBttn;
public static javax.swing.JTable table;
private JButton btnAddData;
private JLabel lblExit;
private JLabel lblCpuSchedulingSimulator;
private JLabel lblNewLabel;
private JLabel lblUtilization;
private JLabel cpuUtilize;
private JLabel lblSubmittedBy;
private JLabel lblSubmittedTo;
private JTextArea txtrEngrJuliusCansino;
private JLabel lblNewLabel_1;
	public JLabel getJLabel7() {
		return jLabel7;
	}

	public static JRadioButton FCFS = new JRadioButton("First Come First Serve");
	public static JRadioButton SJF = new JRadioButton("Shortest Job First");
	public static JRadioButton STRF = new JRadioButton("Shortest Remaining Time First");
	public static JRadioButton RR = new JRadioButton("Round Robin");
	public static JRadioButton Priority1 = new JRadioButton("Priority (Preemptive)");
	public static JRadioButton Priority2 = new JRadioButton("Priority(Non-Preemptive)");

	ButtonGroup group = new ButtonGroup();
	protected static JLabel desctext;
	private JLabel lblDescription;
	private JLabel label;
	private JLabel lblAverage;
	private JLabel lblLegend;
	private JLabel label_3;
	private JLabel label_5;
	private JLabel label_6;
	private JLabel label_7;
	private JLabel label_8;
	private JLabel label_9;
	private JLabel label_10;
	private JLabel label_11;
	private JLabel label_12;
	private JLabel label_13;
	private JLabel lblEmptyJob;
	private JLabel label_1;
```

```
    private JLabel label_2;
    private JLabel label_4;


}
```