```
/*****************************************************************
 * Program Name: PoolVolumeCalculator.java
 * Name: Barbara Williams
 * Program Description: This program will display the current
 *                      date, allow the user to change program title
 *                      pane, add and delete customers and contractors,
 *                      calculate pool volume as well as volume of
 *                      round and oval hot tubs, perform temperature
 *                      and length calculations. Uses/calls on Customer,
 *                      Contractor, and ExitButton classes.
 *****************************************************************/

package Project;
/*
 * File: TestExitButton.java There are two classes defined in this file. The
 * name of this file matches the name of the class that contains the main
method
 * and this class must be public. All other classes do not need the to be
public
 */

import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.swing.JButton;
import javax.swing.ButtonGroup;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JTabbedPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;

public class PoolVolumeCalculator extends JFrame
{
      public PoolVolumeCalculator()
      {
            // Create the tabbed page and add it to the content pane
            JTabbedPane jtp = new JTabbedPane();
            getContentPane().add(jtp);

            // create the General tab and add it to the tabbed pane
            JPanel jp1 = new JPanel();
            jp1 = getGeneralPanel();
            jtp.addTab("General", jp1);

            // create the Options tab and add it to the tabbed pane
            JPanel jp2 = new JPanel();
```

```
        jp2 = getOptionsPanel();
              setTitle("Enter a company name in the Options tab");
        jtp.addTab("Options", jp2);

        // create the Customers tab and add it to the tabbed pane
        JPanel jp3 = new JPanel();
        jp3 = getCustomerPanel();
        jtp.addTab("Customers", jp3);

        //    create the Contractors tab and add it to the tabbed pane
        JPanel jp4 = new JPanel();
        jp4 = getContractorPanel();
        jtp.addTab("Contractors",jp4);

        //    create the Pools tab and add it to the tabbed pane
        JPanel jp5 = new JPanel();
        jp5 = getPoolsPanel();
        jtp.addTab("Pools",jp5);

        //    create the Hot Tubs tab and add it to the tabbed pane
        JPanel jp6 = new JPanel();
        jp6 = getHotTubsPanel();
        jtp.addTab("Hot Tubs",jp6);

        //    create the Temp Calc tab and add it to the tabbed pane
        JPanel jp7 = new JPanel();
        jp7 = getTempCalcPanel();
        jtp.addTab("Temp Calc",jp7);

        //    create the Length Calc tab and add it to the tabbed pane
        JPanel jp8 = new JPanel();
        jp8 = getLengthCalcPanel();
        jtp.addTab("Length Calc",jp8);

        setSize(325, 300);
        setVisible(true);
    }

    /**
     * Method that creates and return the general panel
     *
     * @return a JPanel object
     */
    public JPanel getGeneralPanel()
    {
        final JLabel date = new JLabel();
        final JLabel dateLabel = new JLabel("Today's Date: ");
        Date currentDate = new Date();
        ExitButton exitButton = new ExitButton();

        // General Panel
        JPanel general = new JPanel();

        //    add features to general panel
        general.add(dateLabel);
        general.add(date);
        general.add(exitButton.getExitButton());
```

```
            //    format date
            SimpleDateFormat dateformatMMDDYYYY = new
SimpleDateFormat("MM/dd/yyyy");
      new StringBuilder( dateformatMMDDYYYY.format(currentDate) );
      date.setText(dateformatMMDDYYYY.format(currentDate));

      return general;

     }            // end getGeneralPanel

      /**
       * Method that creates and return the options panel
       *
       * @return a JPanel object
       */
      public JPanel getOptionsPanel()
      {
          final JTextField newTitle = new JTextField(25);
          final JLabel labelOptions = new JLabel("Change Company Name:");
          JButton setNameBtn = new JButton("Set New Name");
          ExitButton exitButton = new ExitButton();

          //      Options Panel
          JPanel options = new JPanel();

            //    add features to the options panel
            options.add(labelOptions);
            options.add(newTitle);
            options.add(setNameBtn);
            options.add(exitButton.getExitButton());

            //    set Mnemonic for button
            setNameBtn.setMnemonic('S');

            //    ActionListener for setNameButton
            setNameBtn.addActionListener(new ActionListener()
            {
                  //    actionPerformed method for determining which button
selected
                  public void actionPerformed(ActionEvent event)
                  {
                        String titleString;
                        titleString = newTitle.getText();

                        //    check titleString length - change title
                        if(titleString.length() > 0)
                  {
                        setTitle(titleString);
                  }           //    end if statement
                  else  // set title back to original
                  {
                        setTitle("Enter a company name in the Options tab" );

                  }           //    end else statement
            }                 //    end actionPerformed
```

```
        });                         // actionListener

        return options;
}                           //    end getOptions()

        /**
         * Method that creates and return the customer panel
         *
         * @return a JPanel object
         */
        public JPanel getCustomerPanel()
        {
                final JTextArea custArea = new JTextArea(7, 25);
                JButton addCustomer = new JButton("Add Customer");
                JButton custRefButton = new JButton("Refresh");
                ExitButton exitbutton = new ExitButton();
                final JTextArea custMessage = new JTextArea(2,25);

                // Customer Panel
                JPanel custPanel = new JPanel();

                //    add features to custPanel
                custPanel.add(custArea);
                custPanel.add(exitbutton.getExitButton());
                custPanel.add(addCustomer);
                custPanel.add(custRefButton);
                custPanel.add(custMessage);

                //    set custArea features
                custArea.setText("Select Add Customer to add customer. Select
Refresh to refresh this pane.");
                custArea.setForeground(Color.blue);
                custArea.setLineWrap(true);
                custArea.setWrapStyleWord(true);
                custArea.setEditable(false);

                //    set custMessage features
                custMessage.setText("File customer.txt does not exist yet-will be
created when you add contractors!");
                custMessage.setForeground(Color.blue);
                custMessage.setLineWrap(true);
                custMessage.setWrapStyleWord(true);

                //    set Mnemonics for buttons
                custRefButton.setMnemonic('R');
                addCustomer.setMnemonic('A');

                //    ActionListener for addCustomer button
                addCustomer.addActionListener(new ActionListener()
                {
                        public void actionPerformed(ActionEvent e)
                        {
                                // action if button is used
                                new Customer("Customer");
                        }     // end actionPerformed()
                });   // end performed action
```

```
            //    ActionListener for custRefButton
            custRefButton.addActionListener(new ActionListener()
            {
                  public void actionPerformed(ActionEvent e)
                  {
                        custMessage.setText("");

                        try
                        {
                              File custOpen = new File("customer.txt");
                              FileReader custAreaIn = new
FileReader(custOpen);
                              BufferedReader buffIn = new
BufferedReader(custAreaIn);
                              String textData = "";
                              StringBuffer sb = new StringBuffer();

                              while ((textData = buffIn.readLine()) != null)
                              {
                                    sb.append(textData+"\n");
                              }                    //    end while statement

                              custArea.setText(sb.toString());
                              buffIn.close();

                              custMessage.setText("The file exists and can be
read from!");
                        }                          //    end try function

                        catch (IOException e3)
                        {
                              custMessage.setText("The file could not be
read. "
                                    + e3.getMessage());
                        }           //    end catch function
                  }                 //    end actionPerformed
            });                     //    end actionListener for
custRefButton

            return custPanel;
      }                             //    end getCustomer()

      /**
       * Method that creates and return the contractor panel
       *
       * @return a JPanel object
       */
      public JPanel getContractorPanel()
      {
            final JTextArea contArea = new JTextArea(7,25);
            JButton addContractor = new JButton("Add Contractor");
            JButton contRefButton = new JButton("Refresh");
            ExitButton exitbutton = new ExitButton();
            final JTextArea contMessage = new JTextArea(2,25);

            // Contractor Panel
            JPanel contPanel = new JPanel();
```

```
            //    add contPanel features
            contPanel.add(contArea);
            contPanel.add(exitbutton.getExitButton());
            contPanel.add(addContractor);
            contPanel.add(contRefButton);
            contPanel.add(contMessage);

            // set contArea features
            contArea.setText("Select Add Contractor to add contractor. Select
Refresh to refresh this pane.");
            contArea.setForeground(Color.blue);
            contArea.setLineWrap(true);
            contArea.setWrapStyleWord(true);
            contArea.setEditable(false);

            //    set contMessage features
            contMessage.setText("File contractor.txt does not exist yet-will
be created when you add contractors!");
            contMessage.setForeground(Color.blue);
            contMessage.setLineWrap(true);
            contMessage.setWrapStyleWord(true);

            //    add Mnemonics to buttons
            addContractor.setMnemonic('A');
            contRefButton.setMnemonic('R');

            //    ActionListener for addContractor button
            addContractor.addActionListener(new ActionListener()
            {
                  public void actionPerformed(ActionEvent e)
                  {
                        // action if button is used
                        new Contractor("Contractor");
                  }     // end actionPerformed()
            });   // end ActionListener for addContractor button

            //    ActionListener for contRefButton
            contRefButton.addActionListener(new ActionListener()
            {
                  public void actionPerformed(ActionEvent e)
                  {
                        //    action if button is used
                        contMessage.setText("");
                        try
                        {
                              File contOpen = new File("contractor.txt");
                              FileReader contAreaIn = new
FileReader(contOpen);
                              BufferedReader buffIn = new
BufferedReader(contAreaIn);
                              String textData = "";
                              StringBuffer sb = new StringBuffer();
                              while ((textData = buffIn.readLine()) != null)
                              {
                                    sb.append(textData+"\n");
                              }     //    end while statement
```

```
                        contArea.setText(sb.toString());
                        buffIn.close();

                        contMessage.setText("The file exists and can be
read from!");
                    }           //    end try/while statement
                    catch (IOException e3)
                    {
                        contMessage.setText("The file could not be
read. "
                                + e3.getMessage());
                    }           //    end catch function
                }               //    end actionPerformed
            });                 //    end ActionListener for
contRefButton

            return contPanel;
        }                       //    end getContractor()

        /**
         * Method that creates and return the pools panel
         *
         * @return a JPanel object
         */
        public JPanel getPoolsPanel()
        {
            final JLabel lengthLabel = new JLabel( "Enter the pool's
length(ft):" );
            final JLabel widthLabel = new JLabel( "Enter the pool's
width(ft):" );
            final JLabel depthLabel = new JLabel( "Enter the pool's
depth(ft):" );
            final JLabel volumeLabel = new JLabel("The pool volume is(ft
^3):");
          final JTextField volumeText = new JTextField(10);
            final JTextField depthText = new JTextField(10);
            final JTextField lengthText = new JTextField(10);
            final JTextField widthText = new JTextField(10);
            final JTextField Msgs = new JTextField(23);
            JButton calculateButton = new JButton("Calculate Volume");
            ExitButton exitButton = new ExitButton();

            // Pools panel
            JPanel pools = new JPanel();

            //    add features to pools panel
            pools.add(lengthLabel);
            pools.add(lengthText);
            pools.add(widthLabel);
            pools.add(widthText);
            pools.add(depthLabel);
            pools.add(depthText);
            pools.add(calculateButton);
            pools.add(exitButton.getExitButton());
            pools.add(volumeLabel);
            pools.add(volumeText);
            pools.add(Msgs);
```

```
//    set Editable features
volumeText.setEditable(false);
Msgs.setEditable(false);

//     set Mnemonic
calculateButton.setMnemonic('C');

//    ActionListener for calculateButton
calculateButton.addActionListener(new ActionListener()
{
      public void actionPerformed(ActionEvent e)
      {
            String lengthString, widthString, depthString;
            int length=0;
            int width=0;
            int depth=0;
            double vol = 0.0;

            //    create a format variable to format the volume
results
            DecimalFormat df = new DecimalFormat(",###.##");

            lengthString = lengthText.getText();
            widthString = widthText.getText();
            depthString = depthText.getText();

            try         // verify numbers in fields
            {
                  length = Integer.parseInt(lengthString);
                  width = Integer.parseInt(widthString);
                  depth = Integer.parseInt(depthString);

                  if ( lengthString.length() < 1 ||
widthString.length() < 1 || depthString.length() < 1 )
                  {
                        volumeText.setText("Error! Must enter in
all three numbers!!"); return;
                  }           // end if statement

                  else if ( length != 0 || width != 0 || depth !=
0 )

                  {
                        vol = (length * width) * depth;
                  }           //    end else if

                  //    format and display the volume
                  volumeText.setText(df.format(vol));

                  // reset the background color of the fields and
hide the message field
                  Msgs.setVisible(false);
                  lengthText.setBackground(Color.WHITE);
                  widthText.setBackground(Color.WHITE);
                  depthText.setBackground(Color.WHITE);
            }                 // end try function
```

```
                        catch (NumberFormatException ne)
                        {
                              Msgs.setVisible(true);
                              Msgs.setText("Please fill out all fields!");

                              //change the background color of the fields to
alert the user
                              Msgs.setBackground(Color.ORANGE);
                              lengthText.setBackground(Color.ORANGE);
                              widthText.setBackground(Color.ORANGE);
                              depthText.setBackground(Color.ORANGE);

                        }           // end catch statement
                  }                 //    end actionPerformed

            });                     //    end ActionListener
(calculateButton)

            return pools;
      }                             //    end getPoolsPanel()

      /**
       * Method that creates and return the hot tubs panel
       *
       * @return a JPanel object
       */
      public JPanel getHotTubsPanel()
      {
            final JTextField htLength = new JTextField(10);
            final JTextField htWidth = new JTextField(10);
            final JTextField htDepth = new JTextField(10);
            final JTextField htVolume = new JTextField(10);
            final JTextField htMsgs = new JTextField(24);
            final JLabel blank = new JLabel("");
            final JLabel lengthLabel = new JLabel("Enter the tub's
length(ft):");
            final JLabel widthLabel = new JLabel("Enter the tub's width(ft):
");
            final JLabel depthLabel = new JLabel("Enter the tub's depth(ft):
");
            final JLabel volumeLabel = new JLabel("The tub's volume is
(ft^3):");
            JButton calcButton = new JButton("Calculate Volume");
            ExitButton exitButton = new ExitButton();
            final JRadioButton rbRoundTub = new JRadioButton("Round Tub");
            final JRadioButton rbOvalTub = new JRadioButton("Oval Tub");
            final ButtonGroup hotTubsGroup = new ButtonGroup();

            // Hot Tubs panel
            JPanel hotTubs = new JPanel();

            //    add features to the hotTubs panel
            hotTubs.add(blank);
            hotTubs.add(rbRoundTub);
            hotTubs.add(rbOvalTub);
            hotTubs.add(lengthLabel);
            hotTubs.add(htLength);
```

```
            hotTubs.add(widthLabel);
            hotTubs.add(htWidth);
            hotTubs.add(depthLabel);
            hotTubs.add(htDepth);
            hotTubs.add(calcButton);
            hotTubs.add(exitButton.getExitButton());
            hotTubs.add(volumeLabel);
            hotTubs.add(htVolume);
            hotTubs.add(htMsgs);

            //    place radio buttons into Button Group so only one can be
selected at a time
            hotTubsGroup.add(rbRoundTub);
            hotTubsGroup.add(rbOvalTub);

            // select Oval radio button as default
            rbOvalTub.setSelected(true);

            //    set calButton Mnemonic
            calcButton.setMnemonic('C');

            //    setEditables
            htVolume.setEditable(false);
            htMsgs.setEditable(false);


            //    ActionListener for OvalTub button
            rbOvalTub.addActionListener(new ActionListener()
            {
                  public void actionPerformed(ActionEvent ea)
                  {
                        //    opens widthField for use
                        htWidth.setEditable(true);
                  }           //    end actionPerformed
            });               //    end ActionListener for OvalTub

            //    ActionListener for RoundTub button
            rbRoundTub.addActionListener(new ActionListener()
            {
                  public void actionPerformed(ActionEvent ea)
                  {
                        //    greys out widthField since unnecessary
                        htWidth.setEditable(false);
                        htMsgs.setText("Width will be set to the same value
as length");
                  }           //    end actionPerformed
            });               //    end ActionListener for roundTub

            // ActionListener for calcButton
            calcButton.addActionListener(new ActionListener()
            {
                  // Determine which button was Selected
                  public void actionPerformed(ActionEvent e)
                  {
                        String lengthString, widthString, depthString;
                        int length = 0;
```

```
                        int width = 0;
                        int depth = 0;
                        double vol = 0.0;

                        // create a format variable to format the volume
results
                        DecimalFormat df = new DecimalFormat(",###.##");

                        lengthString = htLength.getText();
                        widthString = htWidth.getText();
                        depthString = htDepth.getText();

                        // check for valid input value
                        try
                        {
                              //    isSelected method is used to check which
radio button is selected.
                              if (rbRoundTub.isSelected())  //    round tub
is selected
                              {
                                    htMsgs.setText("Tub's width set to
length");

                                    length = Integer.parseInt(lengthString);
                                    depth = Integer.parseInt(depthString);
                                    vol = Math.PI * Math.pow(length / 2, 2) *
depth;
                                    htWidth.setText(htLength.getText());

                              }                    //  end if statement

                              else if (rbOvalTub.isSelected())  //    oval
tub is seleced
                              {
                                    length = Integer.parseInt(lengthString);
                                    width = Integer.parseInt(widthString);
                                    depth = Integer.parseInt(depthString);
                                    vol = Math.PI * Math.pow((length *
width),2) * depth;
                              }                    //  end else if statement

                              // format and display the volume
                              htVolume.setText(df.format(vol));

                              //    reset the background color of the fields
and hide the message field
                              htMsgs.setVisible(false);
                              htLength.setBackground(Color.WHITE);
                              htWidth.setBackground(Color.WHITE);
                              htDepth.setBackground(Color.WHITE);
                        }                          //  end try statement

                        catch (NumberFormatException ne)
                        {
                              htMsgs.setVisible(true);
                              htMsgs.setText("Please fill out all fields!");
```

```
                        //change the background color of the fields to
alert the user
                        htMsgs.setBackground(Color.ORANGE);
                        htLength.setBackground(Color.ORANGE);
                        htWidth.setBackground(Color.ORANGE);
                        htDepth.setBackground(Color.ORANGE);

                    }                   // end catch statement
                }                       // end actionPerformed
            });                     // end calcButton actionListener

            return hotTubs;

        }                               // end getHotTubs()

        /**
         * Method that creates and return the temp calc panel
         *
         * @return a JPanel object
         */
        public JPanel getTempCalcPanel()
        {
            String[] items = {"C", "F"};
            final JComboBox comboBox = new JComboBox(items);
            ExitButton exitButton = new ExitButton();
            JButton btnConvert = new JButton("Convert");
            final JTextField tempText = new JTextField(5);
            final JTextField results = new JTextField(20);
            final JTextField sysMsgs = new JTextField(20);
            final JLabel tempCalcLabel = new JLabel("F");
            final JLabel tempLabel = new JLabel("Enter temperature:");
            final JLabel resultLabel = new JLabel("Result:");

            // Temp Calc panel
            JPanel tempCalc = new JPanel();

            //    add features to the tempCalc panel
        tempCalc.add(tempLabel);
        tempCalc.add(tempText);
        tempCalc.add(comboBox);
        tempCalc.add(resultLabel);
        tempCalc.add(results);
        tempCalc.add(tempCalcLabel);
        tempCalc.add(btnConvert);
        tempCalc.add(exitButton.getExitButton());
        tempCalc.add(sysMsgs);


        //    set Editables
          comboBox.setEditable(true);
          results.setEditable(false);
          sysMsgs.setEditable(false);

          //  set button Mnemonic
          btnConvert.setMnemonic('C');

          sysMsgs.setText("System Messages");
```

```
        /*  Add listener when user selects "C" or "F" changes
         *   tempCalLabel to the opposite letter */
        comboBox.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent e)
            {
                  if(comboBox.getSelectedIndex()== 0)
                  {
                        tempCalcLabel.setText("F");
                  }           //    end if statement

                  else
                  {
                        tempCalcLabel.setText("C");
                  }           //    end else statement
            }                 //    end actionPerformed
        });                   //    end comboBox ActionListener

        /* Add listener when the user selects an item in the
         * combobox calculates appropriate calculations */
        btnConvert.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent ae)
            {
                if(comboBox.getSelectedIndex() == 0)
                {
                  double fahrenheit = Integer.parseInt(tempText.getText());
                    double fah = 1.8 * fahrenheit + 32;
                    DecimalFormat formatters = new DecimalFormat(",###.###");
                    results.setText("" + formatters.format(fah));
                    sysMsgs.setText("");
                }           //    end if statement

                else
                {
                  double celsius = Integer.parseInt(tempText.getText());
                  double cel = (celsius -32)/1.8;
                  DecimalFormat formatters = new DecimalFormat(",###.###");
                  results.setText("" + formatters.format(cel));
                  sysMsgs.setText("");
                }           //    end else statement
            }               //    end actionPerformed
        });                 //    end btnConvert ActionListener

        return tempCalc;
    }           //    end getTempCalc()

     /**
      * Method that creates and return the length calc panel
      *
      * @return a JPanel object
      */
     public JPanel getLengthCalcPanel()
     {
           final JTextField mmText = new JTextField(6);
           final JTextField mText = new JTextField(4);
```

```
        final JTextField ydsText = new JTextField(5);
        final JTextField ftText = new JTextField(4);
        final JTextField inText = new JTextField(5);
        ExitButton exitButton = new ExitButton();
        JButton clearBtn = new JButton("Clear");
        JButton convertBtn = new JButton("Convert");
      final JLabel mmLabel = new JLabel("Millimeters    ");
      final JLabel mLabel = new JLabel("Meters        ");
      final JLabel ydsLabel = new JLabel("Yards        ");
      final JLabel ftLabel = new JLabel("Feet          ");
      final JLabel inLabel = new JLabel("Inches       ");

        // Temp Calc panel
        JPanel lengthCalc = new JPanel();

        //    add features to the lengthCalc panel
        lengthCalc.add(mmLabel);
        lengthCalc.add(mLabel);
        lengthCalc.add(ydsLabel);
        lengthCalc.add(ftLabel);
        lengthCalc.add(inLabel);
        lengthCalc.add(mmText);
        lengthCalc.add(mText);
        lengthCalc.add(ydsText);
        lengthCalc.add(ftText);
        lengthCalc.add(inText);
lengthCalc.add(convertBtn);
lengthCalc.add(clearBtn);
lengthCalc.add(exitButton.getExitButton());

//    set Mnemonic
convertBtn.setMnemonic('C');
  convertBtn.setMnemonic('C');

  // Add listener to clear textfields
  clearBtn.addActionListener(new ActionListener()
  {
      public void actionPerformed(ActionEvent evt)
      {
          mmText.setText("");
          mText.setText("");
          ydsText.setText("");
          ftText.setText("");
          inText.setText("");
          mmText.requestFocus();
  }
  });

  /* Add listener when the user enters a number in
   * one of the textfields then makes the appropriate
   *  calculations */
  convertBtn.addActionListener(new ActionListener()
  {
      public void actionPerformed(ActionEvent evt)
            {
            double mm = 0.0;
                  double m = 0.0;
```

```
                double yds = 0.0;
                double ft = 0.0;
                double in = 0.0;

                //    create a format variable to format the volume
results
                DecimalFormat num = new DecimalFormat(",###.##");

                //    if the millimeters input field is not empty
                if(!mmText.getText().isEmpty())
                {
                      //    Then convert the number to the other
measurements and populate the other textboxes
                      mm = Double.parseDouble(mmText.getText());
                      mText.setText(num.format(mm/1000));
                      ydsText.setText(num.format(mm * 0.001093));
                      ftText.setText(num.format(mm * 0.0032808399));
                      inText.setText(num.format(mm * 0.0393700787));
                }           //    end if statement

                //    else if the meter field is not empty
                else if(!mText.getText().isEmpty())
                {
                      //    Then convert the number to the other
measurements and populate the other textboxes
                      m = Double.parseDouble(mText.getText());
                      mmText.setText(num.format(m*1000));
                      ydsText.setText(num.format(m * 1.0936133));
                      ftText.setText(num.format(m * 3.2808399));
                      inText.setText(num.format(m * 39.3700787));
                }                 //    end else if statment

                //    else if the yards field is not empty
                else if(!ydsText.getText().isEmpty())
                {
                      //    Then convert the number to the other
measurements and populate the other textboxes
                      yds = Double.parseDouble(ydsText.getText());
                      ftText.setText(num.format(yds * 3));
                      inText.setText(num.format(yds * 36));
                      mText.setText(num.format(yds * 0.91));
                      mmText.setText(num.format(yds * 914.4));
                }                 //    end else if statement

                //    else if the feet field is not empty
                else if(!ftText.getText().isEmpty())
                {
                      //    Then convert the
                      ft = Double.parseDouble(ftText.getText());
                      inText.setText(num.format(ft * 12));
                      ydsText.setText(num.format(ft / 3));
                      mText.setText(num.format(ft * 0.3048));
                      mmText.setText(num.format(ft * 304.8));

                }                 //    end else if statement

                //    else if the inches field is not empty
```

```
                        else if(!inText.getText().isEmpty())
                        {
                              in = Double.parseDouble(inText.getText());
                              ftText.setText(num.format(in / 12));
                              ydsText.setText(num.format(in / 36));
                              mText.setText(num.format(in / 0.0254));
                              mmText.setText(num.format(in / 25.4));

                        }           //    end else if statement
                  }                 //    end actionPerformed
      });                           //    end ActionListener (convert button)

      return lengthCalc;

}           //    end getLengthCalc()

      /**
       * Main method
       */
      public static void main(String[] args)
      {
            PoolVolumeCalculator teb = new PoolVolumeCalculator();
            teb.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      }                 //    end main method
}                       //    end PoolVolumeCalculator - Final project
```