Southern Luzon State University



College Of Engineering

Computer Engineering Department

**Activity 17: Java Event and Action Listener**

**CpE05 -**  **Object Oriented Programming**

Name: Dave Jhared G. Paduada Date: May 19, 2024

Section/Schedule: IF T & TH 10:30 - 13:30 Score:\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**SOURCE CODE:**

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.io.File;

import java.io.FileWriter;

import java.io.PrintWriter;

import java.util.Scanner;

public class GUIACtivity\_17\_Java\_Event\_and\_Action\_Listener {

private static JTextField angle\_A\_JTextField;

private static JTextField angle\_B\_JTextField;

private static JTextField side\_a\_JTextField;

private static JTextField side\_b\_JTextField;

private static JTextField side\_c\_JTextField;

private static JTextField[] textFields;

private static int currentIndex;

public static void main(String[] args) {

JFrame frame = new JFrame("Main Solver+");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(500, 700);

frame.setLocationRelativeTo(null);

ImageIcon icon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/March 7th\_6.png");

frame.setIconImage(icon.getImage());

// Menu Bar

JMenuBar menuBar = new JMenuBar();

// Create the File Menu

JMenu fileMenu = new JMenu("File");

fileMenu.setMnemonic(KeyEvent.VK\_F);

ImageIcon newIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/NewWindow.png");

Image image = newIcon.getImage();

Image resizedImage = image.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

newIcon = new ImageIcon(resizedImage);

ImageIcon exitIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/close.png");

Image exitImage = exitIcon.getImage();

Image resizedExitImage = exitImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

exitIcon = new ImageIcon(resizedExitImage);

JMenuItem newWindowItem = new JMenuItem(" New Window ", newIcon);

JMenuItem exItem = new JMenuItem(" Exit ", exitIcon);

newWindowItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_N, ActionEvent.CTRL\_MASK));

exItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F4, KeyEvent.ALT\_DOWN\_MASK));

fileMenu.add(newWindowItem);

fileMenu.add(exItem);

JMenu historyMenu = new JMenu("History");

historyMenu.setMnemonic(KeyEvent.VK\_I);

ImageIcon historyIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/history.png");

Image historyImage = historyIcon.getImage();

Image resizedHistoryImage = historyImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

historyIcon = new ImageIcon(resizedHistoryImage);

ImageIcon clearHistoryIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/broom.png");

Image clearHistoryImage = clearHistoryIcon.getImage();

Image resizedClearHistoryImage = clearHistoryImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

clearHistoryIcon = new ImageIcon(resizedClearHistoryImage);

JMenuItem viewHistoryItem = new JMenuItem(" View History ", historyIcon);

JMenuItem clearHistoryItem = new JMenuItem(" Clear History ", clearHistoryIcon);

viewHistoryItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_H, ActionEvent.CTRL\_MASK));

clearHistoryItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_C, ActionEvent.CTRL\_MASK));

viewHistoryItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

File file = new File("Lesson\_8/GUIActivity\_17\_Calculation\_History.txt");

Scanner scanner = new Scanner(file);

StringBuilder history = new StringBuilder();

while (scanner.hasNextLine()) {

history.append(scanner.nextLine() + "\n");

}

JTextArea textArea = new JTextArea(history.toString());

textArea.setEditable(false);

textArea.setFont(new Font("Arial", Font.PLAIN, 17)); // Set font size to 17

JScrollPane scrollPane = new JScrollPane(textArea);

scrollPane.setPreferredSize(new Dimension(420, 500));

JOptionPane.showMessageDialog(null, scrollPane, "Calculation History", JOptionPane.INFORMATION\_MESSAGE);

scanner.close();

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);

}

}

});

clearHistoryItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int confirmation = JOptionPane.showConfirmDialog(null, "Are you sure you want to clear the history?", "Confirmation", JOptionPane.YES\_NO\_OPTION);

if (confirmation == JOptionPane.YES\_OPTION) {

try {

File file = new File("Lesson\_8/GUIActivity\_17\_Calculation\_History.txt");

PrintWriter writer = new PrintWriter(file);

writer.print("");

writer.close();

JOptionPane.showMessageDialog(null, "History cleared successfully!", "Cleared", JOptionPane.INFORMATION\_MESSAGE);

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

});

historyMenu.add(viewHistoryItem);

historyMenu.add(clearHistoryItem);

// Create the Help menu

JMenu helpMenu = new JMenu("Help");

helpMenu.setMnemonic(KeyEvent.VK\_H);

ImageIcon aboutIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/information.png");

Image aboutImage = aboutIcon.getImage();

Image resizedAboutImage = aboutImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

aboutIcon = new ImageIcon(resizedAboutImage);

ImageIcon gCashIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/gcash.png");

Image gCashImage = gCashIcon.getImage();

Image resizedGCashImage = gCashImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

gCashIcon = new ImageIcon(resizedGCashImage);

ImageIcon paypalIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/paypal.png");

Image paypalImage = paypalIcon.getImage();

Image resizedPaypalImage = paypalImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

paypalIcon = new ImageIcon(resizedPaypalImage);

ImageIcon landbankIcon = new ImageIcon("/home/deyb/Documents/Java OOP/Pictures/landbank.png");

Image landbankImage = landbankIcon.getImage();

Image resizedLandbankImage = landbankImage.getScaledInstance(27, 27, Image.SCALE\_SMOOTH);

landbankIcon = new ImageIcon(resizedLandbankImage);

JMenuItem aboutItem = new JMenuItem(" About ", aboutIcon);

JMenu subDonateMenu = new JMenu(" Donate Via... ");

JMenuItem donateGcash = new JMenuItem(" GCash ", gCashIcon);

JMenuItem donatePaypal = new JMenuItem(" Paypal ", paypalIcon);

JMenuItem donateLandbank = new JMenuItem(" Landbank ", landbankIcon);

subDonateMenu.add(donateGcash);

subDonateMenu.add(donatePaypal);

subDonateMenu.add(donateLandbank);

aboutItem.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F1, 0));

subDonateMenu.setMnemonic(KeyEvent.VK\_D);

donateGcash.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_G, ActionEvent.CTRL\_MASK));

donatePaypal.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_P, ActionEvent.CTRL\_MASK));

donateLandbank.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_L, ActionEvent.CTRL\_MASK));

helpMenu.add(aboutItem);

helpMenu.add(subDonateMenu);

fileMenu.setFont(new Font("Arial", Font.BOLD, 17));

newWindowItem.setFont(new Font("Arial", Font.BOLD, 17));

exItem.setFont(new Font("Arial", Font.BOLD, 17));

historyMenu.setFont(new Font("Arial", Font.BOLD, 17));

viewHistoryItem.setFont(new Font("Arial", Font.BOLD, 17));

clearHistoryItem.setFont(new Font("Arial", Font.BOLD, 17));

helpMenu.setFont(new Font("Arial", Font.BOLD, 17));

aboutItem.setFont(new Font("Arial", Font.BOLD, 17));

subDonateMenu.setFont(new Font("Arial", Font.BOLD, 17));

donateGcash.setFont(new Font("Arial", Font.BOLD, 17));

donatePaypal.setFont(new Font("Arial", Font.BOLD, 17));

donateLandbank.setFont(new Font("Arial", Font.BOLD, 17));

// Add menus to the menu bar

menuBar.add(fileMenu);

menuBar.add(historyMenu);

menuBar.add(helpMenu);

JPanel northDisplayPanel = new JPanel(new GridLayout(1, 2, 20, 0));

northDisplayPanel.setBackground(Color.BLACK);

JPanel leftPanel = new JPanel(new GridLayout(1, 1));

leftPanel.setBackground(Color.BLACK);

JLabel type\_of\_givenJLabel = new JLabel("Type of Given: ");

type\_of\_givenJLabel.setForeground(Color.WHITE);

leftPanel.add(type\_of\_givenJLabel);

JPanel rightPanel = new JPanel(new GridLayout(1, 1));

JComboBox<String> type\_of\_givenJComboBox = new JComboBox<>();

type\_of\_givenJComboBox.addItem("-- Please Select --");

type\_of\_givenJComboBox.addItem("1 side and 1 Angle");

type\_of\_givenJComboBox.addItem("2 sides w/ or w/out Angle");

type\_of\_givenJComboBox.setBackground(Color.LIGHT\_GRAY);

rightPanel.add(type\_of\_givenJComboBox);

type\_of\_givenJLabel.setFont(new Font("Arial", Font.PLAIN, 20));

type\_of\_givenJComboBox.setFont(new Font("Arial", Font.PLAIN, 17));

northDisplayPanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

northDisplayPanel.add(leftPanel);

northDisplayPanel.add(rightPanel);

JPanel centerPanel = new JPanel(new GridLayout(10, 1));

centerPanel.setBackground(Color.BLACK);

JLabel angle\_A = new JLabel("Angle A: ");

angle\_A\_JTextField = new JTextField();

JLabel angle\_B = new JLabel("Angle B: ");

angle\_B\_JTextField = new JTextField();

JLabel side\_a = new JLabel("Side a: ");

side\_a\_JTextField = new JTextField();

JLabel side\_b = new JLabel("Side b: ");

side\_b\_JTextField = new JTextField();

JLabel side\_c = new JLabel("Side c: ");

side\_c\_JTextField = new JTextField();

textFields = new JTextField[]{side\_a\_JTextField, side\_b\_JTextField, side\_c\_JTextField, angle\_A\_JTextField, angle\_B\_JTextField};

currentIndex = 0;

centerPanel.add(side\_a);

centerPanel.add(side\_a\_JTextField);

centerPanel.add(side\_b);

centerPanel.add(side\_b\_JTextField);

centerPanel.add(side\_c);

centerPanel.add(side\_c\_JTextField);

centerPanel.add(angle\_A);

centerPanel.add(angle\_A\_JTextField);

centerPanel.add(angle\_B);

centerPanel.add(angle\_B\_JTextField);

angle\_A.setFont(new Font("Arial", Font.PLAIN, 20));

angle\_A.setForeground(Color.WHITE);

angle\_A\_JTextField.setFont(new Font("Arial", Font.BOLD, 17));

angle\_A\_JTextField.setBackground(Color.LIGHT\_GRAY);

angle\_A\_JTextField.setForeground(Color.BLACK);

angle\_B.setFont(new Font("Arial", Font.PLAIN, 20));

angle\_B.setForeground(Color.WHITE);

angle\_B\_JTextField.setFont(new Font("Arial", Font.BOLD, 17));

angle\_B\_JTextField.setBackground(Color.LIGHT\_GRAY);

angle\_B\_JTextField.setForeground(Color.BLACK);

side\_a.setFont(new Font("Arial", Font.PLAIN, 20));

side\_a.setForeground(Color.WHITE);

side\_a\_JTextField.setFont(new Font("Arial", Font.BOLD, 17));

side\_a\_JTextField.setBackground(Color.LIGHT\_GRAY);

side\_a\_JTextField.setForeground(Color.BLACK);

side\_b.setFont(new Font("Arial", Font.PLAIN, 20));

side\_b.setForeground(Color.WHITE);

side\_b\_JTextField.setFont(new Font("Arial", Font.BOLD, 17));

side\_b\_JTextField.setBackground(Color.LIGHT\_GRAY);

side\_b\_JTextField.setForeground(Color.BLACK);

side\_c.setFont(new Font("Arial", Font.PLAIN, 20));

side\_c.setForeground(Color.WHITE);

side\_c\_JTextField.setFont(new Font("Arial", Font.BOLD, 17));

side\_c\_JTextField.setBackground(Color.LIGHT\_GRAY);

side\_c\_JTextField.setForeground(Color.BLACK);

for (JTextField textField : textFields) {

textField.setEnabled(false);

textField.setDisabledTextColor(Color.DARK\_GRAY);

}

for (JTextField textField : textFields) {

textField.addFocusListener(new FocusAdapter() {

@Override

public void focusGained(FocusEvent e) {

textField.setBorder(BorderFactory.createLineBorder(Color.BLUE, 3));

}

@Override

public void focusLost(FocusEvent e) {

textField.setBorder(BorderFactory.createLineBorder(Color.BLACK));

}

});

}

centerPanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

JPanel southPanel = new JPanel(new GridLayout(1, 2, 10, 0));

southPanel.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 20));

southPanel.setBackground(Color.BLACK);

JPanel leftSouthPanel = new JPanel(new GridLayout(4, 3, 3, 3));

String[] buttonLabels = {

"7", "8", "9",

"4", "5", "6",

"1", "2", "3",

"00", "0", "."

};

JButton[] numberButtons = new JButton[buttonLabels.length];

for (int i = 0; i < buttonLabels.length; i++) {

numberButtons[i] = new JButton(buttonLabels[i]);

numberButtons[i].setFont(new Font("Arial", Font.BOLD, 20)); // Set font to Arial BOLD with size 20

numberButtons[i].setForeground(Color.BLACK);

numberButtons[i].setBackground(Color.LIGHT\_GRAY);

leftSouthPanel.add(numberButtons[i]);

// Add ActionListener to each button

numberButtons[i].addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

JButton source = (JButton) e.getSource();

textFields[currentIndex].setText(textFields[currentIndex].getText() + source.getText());

}

});

}

leftSouthPanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

leftSouthPanel.setBackground(Color.DARK\_GRAY);

JPanel rightSouthPanel = new JPanel(new GridLayout(5, 1, 0, 10));

JButton clearJButton = new JButton("C");

JButton clearAllButton = new JButton("AC");

JButton previousButton = new JButton("Previous");

JButton nextButton = new JButton("Next");

JButton solveJButton = new JButton("Solve");

clearJButton.setFont(new Font("Arial", Font.BOLD, 17));

clearAllButton.setFont(new Font("Arial", Font.BOLD, 17));

previousButton.setFont(new Font("Arial", Font.BOLD, 17));

nextButton.setFont(new Font("Arial", Font.BOLD, 17));

solveJButton.setFont(new Font("Arial", Font.BOLD, 17));

clearJButton.setForeground(Color.BLACK);

clearAllButton.setForeground(Color.BLACK);

clearJButton.setBackground(new Color(255, 140, 0));

clearAllButton.setBackground(new Color(255, 140, 0));

previousButton.setForeground(Color.WHITE);

previousButton.setBackground(Color.DARK\_GRAY);

nextButton.setForeground(Color.WHITE);

nextButton.setBackground(Color.DARK\_GRAY);

solveJButton.setForeground(Color.BLACK);

solveJButton.setBackground(Color.GREEN);

rightSouthPanel.add(clearJButton);

rightSouthPanel.add(clearAllButton);

rightSouthPanel.add(previousButton);

rightSouthPanel.add(nextButton);

rightSouthPanel.add(solveJButton);

rightSouthPanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

rightSouthPanel.setBackground(Color.DARK\_GRAY);

southPanel.add(leftSouthPanel);

southPanel.add(rightSouthPanel);

frame.setJMenuBar(menuBar);

frame.add(northDisplayPanel, BorderLayout.NORTH);

frame.add(centerPanel, BorderLayout.CENTER);

frame.add(southPanel, BorderLayout.SOUTH);

type\_of\_givenJComboBox.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

boolean enable = !type\_of\_givenJComboBox.getSelectedItem().equals("-- Please Select --");

clearAllTextFields();

currentIndex = 0;

updateTextFieldState(enable);

clearJButton.setEnabled(enable);

clearAllButton.setEnabled(enable);

nextButton.setEnabled(enable);

previousButton.setEnabled(enable);

solveJButton.setEnabled(enable);

for (JButton button : numberButtons) {

button.setEnabled(enable);

}

if (type\_of\_givenJComboBox.getSelectedIndex() == 1 || type\_of\_givenJComboBox.getSelectedIndex() == 2) {

side\_a\_JTextField.setBorder(BorderFactory.createLineBorder(Color.BLUE, 3));

} else {

side\_a\_JTextField.setBorder(BorderFactory.createEmptyBorder());

}

}

});

// When the user clicks the next button, it move to the next text field

// and disable the previous text field

nextButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

currentIndex++;

if (currentIndex >= textFields.length) {

currentIndex = 0;

}

textFields[currentIndex].requestFocus();

side\_a\_JTextField.setBorder(BorderFactory.createEmptyBorder());

}

});

previousButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if (currentIndex > 0) {

currentIndex--;

updateTextFieldState(true);

JTextField nextTextField = textFields[currentIndex];

nextTextField.requestFocus();

nextTextField.setBorder(BorderFactory.createLineBorder(Color.BLUE, 3));

}

}

});

angle\_A\_JTextField.addFocusListener(new FocusAdapter() {

@Override

public void focusGained(FocusEvent e) {

angle\_A\_JTextField.setBorder(BorderFactory.createLineBorder(Color.GREEN, 3));

}

@Override

public void focusLost(FocusEvent e) {

angle\_A\_JTextField.setBorder(BorderFactory.createEmptyBorder());

}

});

angle\_B\_JTextField.addFocusListener(new FocusAdapter() {

@Override

public void focusGained(FocusEvent e) {

angle\_B\_JTextField.setBorder(BorderFactory.createLineBorder(Color.GREEN, 3));

}

@Override

public void focusLost(FocusEvent e) {

angle\_B\_JTextField.setBorder(BorderFactory.createEmptyBorder());

}

});

clearJButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

textFields[currentIndex].setText("");

}

});

clearAllButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

clearAllTextFields();

}

});

solveJButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

if (type\_of\_givenJComboBox.getSelectedIndex() == 2) {

twoSidesWithOrWithoutAngle();

} else {

oneSideWithAngle();

}

displayResults();

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);

}

}

});

// Initially select the first item in JComboBox

type\_of\_givenJComboBox.setSelectedIndex(0);

frame.setVisible(true);

}

private static void updateTextFieldState(boolean enable) {

for (int i = 0; i < textFields.length; i++) {

textFields[i].setEnabled(i == currentIndex && enable);

}

}

private static void clearAllTextFields() {

for (JTextField textField : textFields) {

textField.setText("");

textField.setEnabled(false);

}

}

// Calculations

// One side and one angle

private static void oneSideWithAngle() throws Exception {

double angleA = parseField(angle\_A\_JTextField);

double angleB = parseField(angle\_B\_JTextField);

double sideA = parseField(side\_a\_JTextField);

double sideB = parseField(side\_b\_JTextField);

double sideC = parseField(side\_c\_JTextField);

// Check if the sides form a valid right triangle

if (sideC != -1 && (sideA > sideC || sideB > sideC)) {

throw new Exception("The hypotenuse must be greater than the other sides.");

}

// Now use trigonometry to find missing sides:

if (angleA == -1 && angleB != -1) {

angleA = 90 - angleB;

} else if (angleB == -1 && angleA != -1) {

angleB = 90 - angleA;

}

if (sideC != -1) {

if (sideA == -1 && angleA != -1) {

sideA = Math.sin(Math.toRadians(angleA)) \* sideC;

}

if (sideB == -1 && angleA != -1) {

sideB = Math.cos(Math.toRadians(angleA)) \* sideC;

}

}

if (sideB != -1) {

if (sideC == -1 && angleA != -1) {

sideC = sideB / Math.cos(Math.toRadians(angleA));

}

if (sideA == -1 && angleA != -1) {

sideA = Math.tan(Math.toRadians(angleA)) \* sideB;

}

}

if (sideA != -1) {

if (sideC == -1 && angleA != -1) {

sideC = sideA / Math.sin(Math.toRadians(angleA));

}

if (sideB == -1 && angleA != -1) {

sideB = sideA / Math.tan(Math.toRadians(angleA));

}

}

if (sideB != -1 && angleB != -1) {

if (sideC == -1) {

sideC = sideB / Math.sin(Math.toRadians(angleB));

}

if (sideA == -1) {

sideA = Math.cos(Math.toRadians(angleB)) \* sideB;

}

}

sideA = Math.round(sideA \* 100.0) / 100.0;

sideB = Math.round(sideB \* 100.0) / 100.0;

sideC = Math.round(sideC \* 100.0) / 100.0;

angleA = Math.round(angleA \* 100.0) / 100.0;

angleB = Math.round(angleB \* 100.0) / 100.0;

if (angleA != -1 && angleB == -1) {

angleB = 90 - angleA;

} else if (angleB != -1 && angleA == -1) {

angleA = 90 - angleB;

}

angle\_A\_JTextField.setText(String.valueOf(angleA));

angle\_B\_JTextField.setText(String.valueOf(angleB));

side\_a\_JTextField.setText(String.valueOf(sideA));

side\_b\_JTextField.setText(String.valueOf(sideB));

side\_c\_JTextField.setText(String.valueOf(sideC));

}

// Two sides with or without angles

private static void twoSidesWithOrWithoutAngle() throws Exception{

double angleA = parseField(angle\_A\_JTextField);

double angleB = parseField(angle\_B\_JTextField);

double sideA = parseField(side\_a\_JTextField);

double sideB = parseField(side\_b\_JTextField);

double sideC = parseField(side\_c\_JTextField);

// Check if the sides form a valid right triangle

if (sideC != -1 && (sideA > sideC || sideB > sideC)) {

throw new Exception("The hypotenuse must be greater than the other sides.");

}

// Pythagorean theorem

if (sideA == -1 && sideB != -1 && sideC != -1) {

sideA = Math.round(Math.sqrt(Math.pow(sideC, 2) - Math.pow(sideB, 2)) \* 100) / 100.0; // We use -(negative) here because we transposed the formula

} else if (sideB == -1 && sideA != -1 && sideC != -1) {

sideB = Math.round(Math.sqrt(Math.pow(sideC, 2) - Math.pow(sideA, 2)) \* 100) / 100.0;

} else if (sideC == -1 && sideA != -1 && sideB != -1) {

/\* If side C is missing, Calculate sideC using the Pythagorean theorem

By adding the calculation for side C as shown, the method now covers the case where

side C is missing and ensures that all sides and angles can be determined from just two known sides. \*/

sideC = Math.round(Math.sqrt(Math.pow(sideA, 2) + Math.pow(sideB, 2)) \* 100) / 100.0;// Standard a^2 + b^2 = c^2 then square root to get c

}

if (angleA == -1 && sideA != -1 && sideB != -1) { // If angle A is missing and side A and side B are present, we can calculate angle A

angleA = Math.round(Math.toDegrees(Math.atan(sideA / sideB)) \* 100) / 100.0;

angleB = Math.round((90 - angleA) \* 100) / 100.0;

}

if (angleB == -1 && sideA != -1 && sideB != -1) { // If angle B is missing and side A and side B are present, we can calculate angle B

angleB = Math.round(Math.toDegrees(Math.atan(sideB / sideA)) \* 100) / 100.0;

angleA = Math.round((90 - angleB) \* 100) / 100.0;

}

// If angle A or B is not present and angle B or A is present, we can calculate angle A or B using the formula: 90 - angle B or A

if (angleA != -1 && angleB == -1) {

angleB = 90 - angleA;

} else if (angleB != -1 && angleA == -1) {

angleA = 90 - angleB;

}

angle\_A\_JTextField.setText(String.valueOf(angleA));

angle\_B\_JTextField.setText(String.valueOf(angleB));

side\_a\_JTextField.setText(String.valueOf(sideA));

side\_b\_JTextField.setText(String.valueOf(sideB));

side\_c\_JTextField.setText(String.valueOf(sideC));

}

// Display the results

private static void displayResults() throws Exception {

double angleA = parseField(angle\_A\_JTextField);

double angleB = parseField(angle\_B\_JTextField);

double sideA = parseField(side\_a\_JTextField);

double sideB = parseField(side\_b\_JTextField);

double sideC = parseField(side\_c\_JTextField);

String results = "Results:\n\n" +

"Side a (opposite): " + sideA + "\n" +

"Side b (adjacent): " + sideB + "\n" +

"Side c (hypotenuse): " + sideC + "\n\n" +

"Angle A: " + angleA + " degrees\n" +

"Angle B: " + angleB + " degrees\n" +

"Angle C: 90.0 degrees\n" +

"Total Angle: " + (angleA + angleB + 90.0) + " degrees\n\n";

Object[] options = { "OK", "SAVE" };

int option = JOptionPane.showOptionDialog(null, results, "Results",

JOptionPane.DEFAULT\_OPTION, JOptionPane.INFORMATION\_MESSAGE, null, options, options[0]);

if (option == 0) {

return;

} else if (option == 1) { // SAVE selected

saveResultsOutside();

}

}

private static double parseField(JTextField textField) {

try {

return Double.parseDouble(textField.getText());

} catch (NumberFormatException e) {

return -1;

}

}

// Save the results to a file

private static void saveResultsOutside() throws Exception {

File file = new File("Lesson\_8/GUIActivity\_17\_Calculation\_History.txt");

PrintWriter writer = new PrintWriter(new FileWriter(file, true));

writer.println(" Hypotenuse: " + side\_c\_JTextField.getText());

writer.println(" Adjacent: " + side\_b\_JTextField.getText());

writer.println(" Opposite: " + side\_a\_JTextField.getText() + "\n");

writer.println(" Angle A: " + angle\_A\_JTextField.getText());

writer.println(" Angle B: " + angle\_B\_JTextField.getText() + "\n");

writer.println(" Total Angle: " + (parseField(angle\_A\_JTextField) + parseField(angle\_B\_JTextField) + 90.0) + " degrees\n");

writer.println(" Last Updated on: " + new java.util.Date());

writer.println();

writer.println("---------------------------------------------------------------");

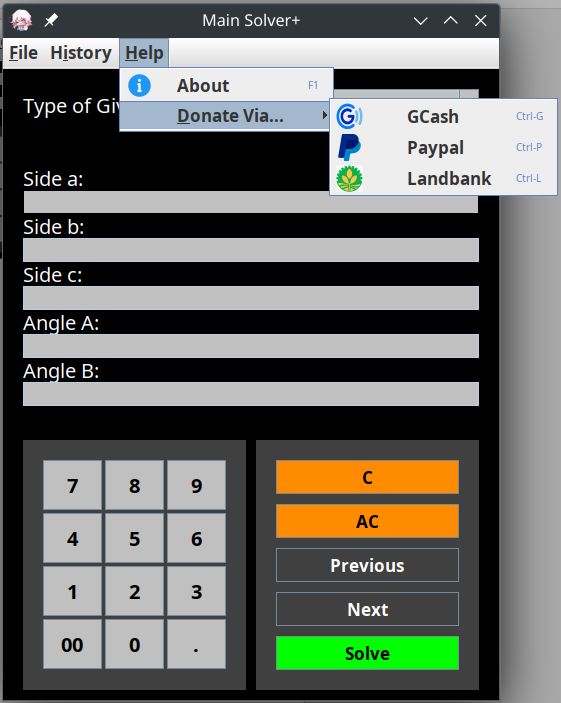
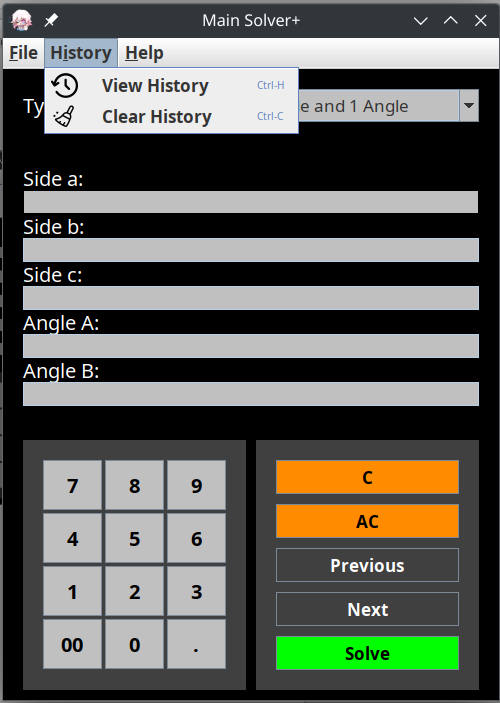
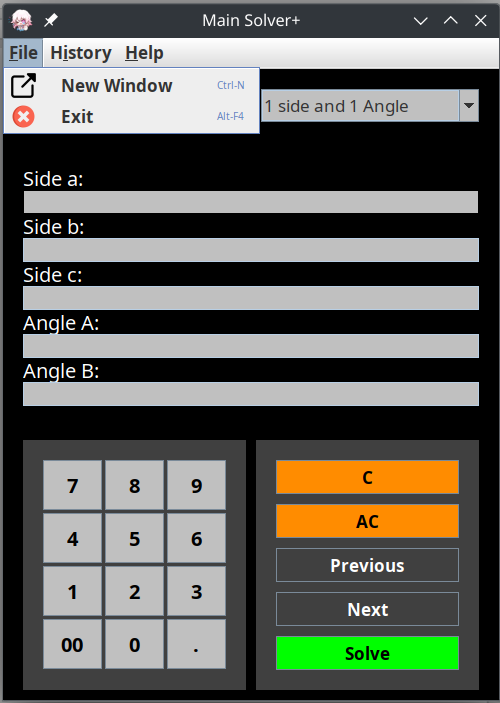
writer.close();

JOptionPane.showMessageDialog(null, "Results saved successfully!", "Saved", JOptionPane.INFORMATION\_MESSAGE);

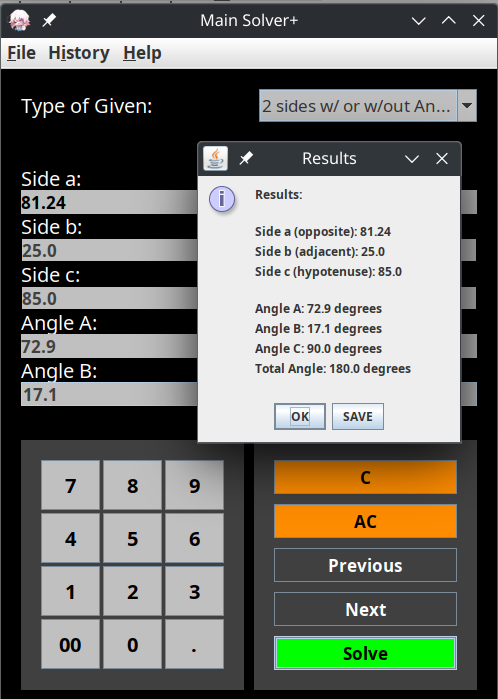
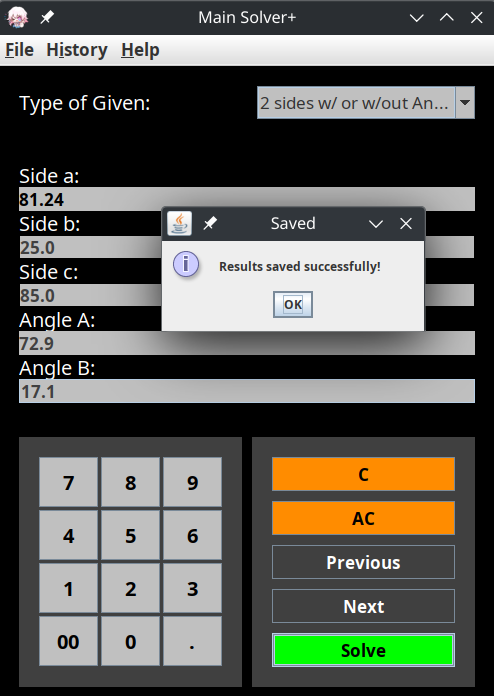
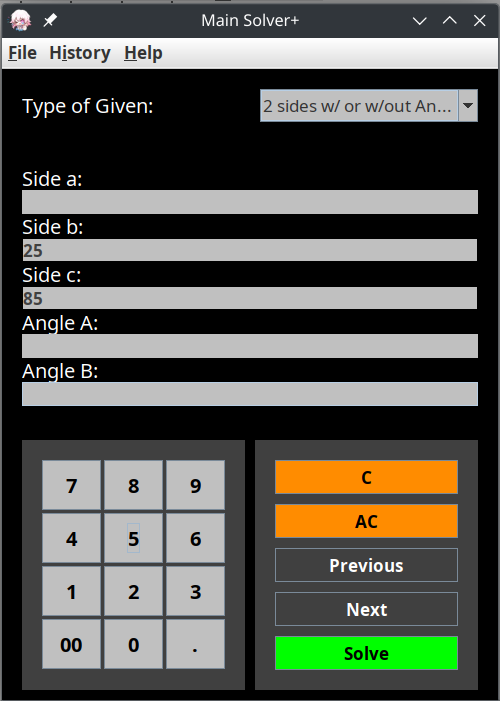
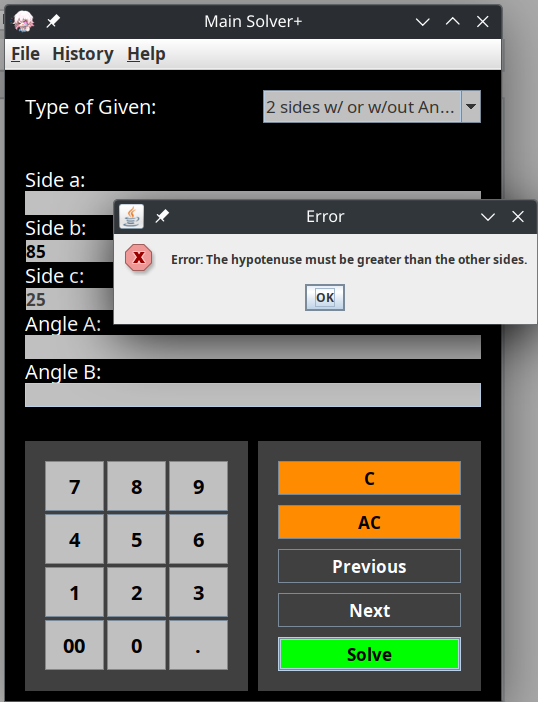
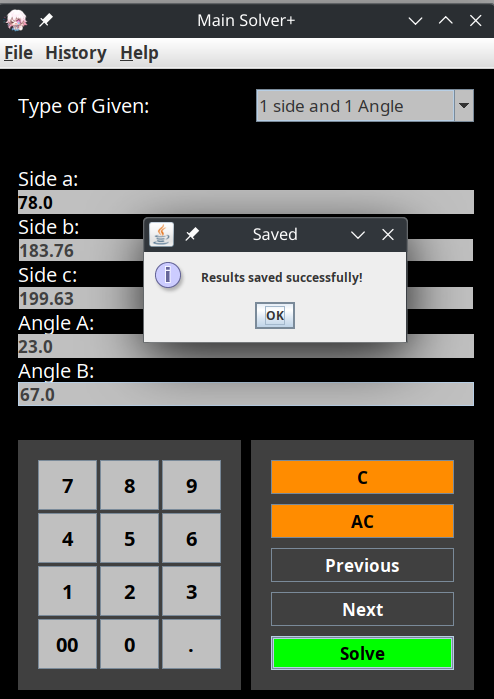
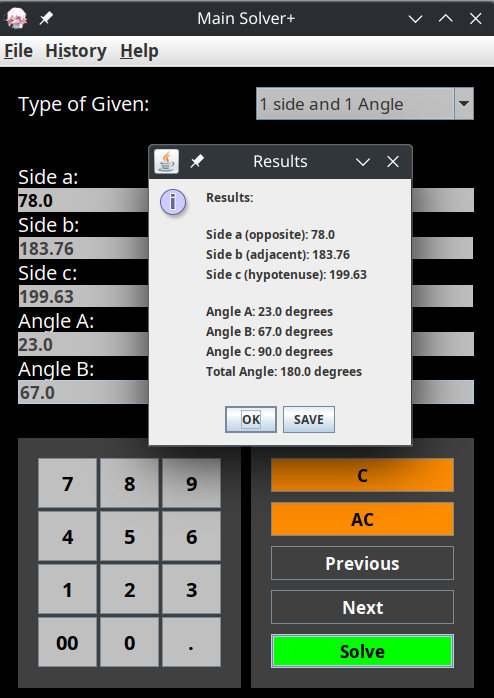
}

}

**SAMPLE OUTPUT:**



**CALCULATION:**



**HISTORY:**

