|  |  |
| --- | --- |
| Student Name: | Adwaiy Singh |
| Reg No | 220968424 |
| Assignment No. | FISAC |
| Subject Code | DSE 2123 |
| Subject | OOP with Java |
| Marks | 10M |

Admission Entrance Test Application

**Overview**:

The Admission Entrance Test Application is a Java-based GUI application designed for conducting admission entrance tests with multiple-choice questions (MCQs). The application includes a login window for user authentication, an MCQ page with 5 questions, and a result display page.

**Flow Diagram:**

Login Window

MCQ Page

Result Display

**Components Used**:

Login Window:

* JFrame (pageFrame): Main frame for the application.
* JLabel (loginTitle): Displayed "Login" as the title.
* JPanel (loginBody, loginBodyShadow): Panels used for organizing components on the login window.
* JLayeredPane (loginLayer): Layered pane for managing the overlapping of panels.
* JTextField (loginUserName, loginUserPass): Text fields for entering username and password.
* JButton (userLoginButton, userForgotPass): Buttons for login and forgot password.

MCQ Page:

* JFrame (mcqWindow): Main frame for the MCQ page.
* JPanel (mcqTitlePanel, mcqBody): Panels for organizing components on the MCQ page.
* JLabel (mcqTitle, mcqQuesNum, mcqQuestion): Labels for the title, question number, and question text.
* ButtonGroup (mcqOptionsGrp): Group for radio buttons to ensure only one option is selected.
* JRadioButton (mcqOption1, mcqOption2, mcqOption3, mcqOption4): Radio buttons for selecting answer options.
* JButton (mcqPrevButton, mcqNextButton): Buttons for navigating between questions.

Result Display:

* JFrame (resultWindow): Main frame for the result display.
* JPanel (resultBody, ansPanel, ansIconPanel): Panels for organizing components on the result display.
* JLabel (resultTitle, scoreLabel, userAnsLabel, correctAnsLabel): Labels for the title, score, and user/correct answers.

**Events and Actions:**

Login Window:

* ActionListener (loginWindowClass implements ActionListener):

1. Handles actions for login and forgot password buttons.
2. Verifies user credentials.
3. Navigates to the MCQ page upon successful login.

MCQ Page:

* ActionListener (mcqWindowClass implements ActionListener):

1. Handles actions for next and previous buttons.
2. Updates and stores user answers.
3. Submits the quiz on the last question.

Result Display:

* ImageIcon (createImagePanel):

1. Loads and resizes images for correct/incorrect icons.
2. calculateScore (resultWindowClass):
3. Calculates the user's total score based on the correctness of answers.

* addAnswersToPanel (resultWindowClass):

1. Dynamically adds user's answers, correct answers, and icons to panels.
2. createResultLabel and createAnsLabel (resultWindowClass):
3. Creates labels for score and answer display.

Additional:

* FocusListener (loginPgTextField): Provides functionality for placeholder text in the login text fields.
* ImageIO (createImagePanel): Loads and resizes images for icons.
* clearSelection() (mcqOptionsGrp): Clears radio button selection to handle no option selected state.
* JOptionPane (loginWindowClass, mcqWindowClass, resultWindowClass): Displays dialog boxes for error messages, information, and confirmation.
* Arrays (mcqUserAnswers, mcqQuestionArray, mcqOptionsArrray, mcqAnswerKey): Stores user answers, questions, options, and correct answers.

**Program code:**

Main.java

/\*

 \* OOP [DSE 2123] FISAC : Mini Project

 \* Name - Adwaiy Singh

 \* Reg. No. - 220968424

 \* Sec. - DSE A

 \*

 \* Run Main to start program

 \*/

public class Main {

    public static *void* main(String[] *args*) {

        new loginWindowClass();

    }

}

loginWindowClass.java

import java.awt.Color;

import java.awt.Font;

import java.awt.Image;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.FocusEvent;

import java.awt.event.FocusListener;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import javax.imageio.ImageIO;

import javax.swing.BorderFactory;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JLayeredPane;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JTextField;

// Main frame used for creating window

class pageFrame extends JFrame {

    pageFrame() {

        ImageIcon logo = new ImageIcon("Manipal-Uni-Logo.jpg");

        this.setTitle("Admission Entrance Test");

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

        this.setSize(800, 600);

        this.setResizable(false);

        this.setIconImage(logo.getImage());

        this.getContentPane().setBackground(new Color(0x37A7DB));

        this.setLayout(null);

    }

}

// TextFields used for input of username and password for login page

class loginPgTextField extends JTextField {

    loginPgTextField (String *text*, *int* *y\_pos*) {

        this.setText(*text*);

        this.setFont(new Font("Fira Sans", Font.PLAIN, 20));

        this.setForeground(new Color(0x292A2F));

        this.setBounds(110, *y\_pos*, 300, 40);

        this.addFocusListener(new FocusListener() {

            public *void* focusGained(FocusEvent *e*) {

                setText("");

            }

            public *void* focusLost(FocusEvent *e*) {

                if (getText().equals("")) {

                    setText(*text*);

                }

            }

        });

    }

}

// Used for creating buttons on login window

class loginPgButton extends JButton {

    loginPgButton(String *type*) {

        if (*type*.equals("login")) {

            setText("Log In");

            setForeground(new Color(0xFFFFFF));

            setBackground(new Color(0x1d28c9));

            setFont(new Font("Sofia Sans", Font.BOLD, 25));

            setBounds(50, 240, 400, 50);

            setFocusPainted(false);

            setBorder(BorderFactory.createBevelBorder(0));

        } else if (*type*.equals("forgotPass")) {

            setText("Forgot Password?");

            setFocusPainted(false);

            setFont(new Font("Sofia Sans", Font.ITALIC, 15));

            setBounds(165, 300, 175, 30);

            setContentAreaFilled(false);

            setBorderPainted(false);

            setOpaque(false);

        } else {

            System.out.println("Invalid input");

        }

    }

}

// Main login window class - start of program

class loginWindowClass implements ActionListener {

    // declaring and initialising all the variables used for login page

    pageFrame loginWindow = new pageFrame();

    JLabel loginTitle = createTitleLabel();

    JPanel loginBody = createBodyPanel(new Color(0xFFFFFF), 150, 100);

    JPanel loginBodyShadow = createBodyPanel(new Color(0xDEE5E8), 160, 110);

    JLayeredPane loginLayer = createLoginLayer();

    loginPgTextField loginUserName = new loginPgTextField("Username", 110);

    JLabel userNameIconLabel = createImagePanel("Username-icon.jpg", 107);

    loginPgTextField loginUserPass = new loginPgTextField("Password", 175);

    JLabel userPassIconLabel = createImagePanel("Password-icon.jpg", 172);

    loginPgButton userLoginButton = new loginPgButton("login");

    loginPgButton userForgotPass = new loginPgButton("forgotPass");

    loginWindowClass() {

        userLoginButton.addActionListener(this);

        userForgotPass.addActionListener(this);

        loginBody.add(loginTitle);

        loginBody.add(userNameIconLabel);

        loginBody.add(loginUserName);

        loginBody.add(userPassIconLabel);

        loginBody.add(loginUserPass);

        loginBody.add(userLoginButton);

        loginBody.add(userForgotPass);

        loginWindow.add(loginLayer);

        loginWindow.setVisible(true);

    }

    // functions to create the login page components

    private JLabel createTitleLabel() {

        JLabel label = new JLabel();

        label.setText("Login");

        label.setForeground(new Color(0x000000));

        label.setFont(new Font("Open Sans", Font.BOLD, 50));

        label.setBounds(35, 10, 250, 75);

        return label;

    }

    private JPanel createBodyPanel(Color *color*, *int* *x\_pos*, *int* *y\_pos*) {

        JPanel panel = new JPanel();

        panel.setBackground(*color*);

        panel.setBounds(*x\_pos*, *y\_pos*, 500, 350);

        panel.setLayout(null);

        panel.setOpaque(true);

        return panel;

    }

    private JLayeredPane createLoginLayer() {

        JLayeredPane layeredPane = new JLayeredPane();

        layeredPane.setBounds(0, 0, 800, 600);

        layeredPane.add(loginBodyShadow, JLayeredPane.DEFAULT\_LAYER);

        layeredPane.add(loginBody, JLayeredPane.DRAG\_LAYER);

        return layeredPane;

    }

    // shrink the size of image to 45x45 pixels

    private JLabel createImagePanel(String *imageName*, *int* *y\_pos*) {

        JLabel label = new JLabel();

        try {

            BufferedImage originalImage = ImageIO.read(new File(*imageName*));

            Image resizedImage = originalImage.getScaledInstance(45, 45, Image.SCALE\_SMOOTH);

            ImageIcon icon = new ImageIcon(resizedImage);

            label.setIcon(icon);

        } catch (IOException e) {

            e.printStackTrace();

        }

        label.setBounds(55, *y\_pos*, 45, 45);

        return label;

    }

    @*Override*

    public *void* actionPerformed(ActionEvent *e*) {

        if (*e*.getSource() == userLoginButton) {

            if (loginUserName.getText().equals("220968424") &&

                    loginUserPass.getText().equals("220968424")) {

                // removes the login window and runs the MCQ page

                loginWindow.dispose();

                new mcqWindowClass();

            } else {

                JOptionPane.showMessageDialog(loginWindow, "Invalid Username or Password",

                null, JOptionPane.ERROR\_MESSAGE);

            }

        }

        if (*e*.getSource() == userForgotPass) {

            JOptionPane.showMessageDialog(loginWindow, "Contact Website Admin",

            null, JOptionPane.INFORMATION\_MESSAGE);

        }

    }

}

mcqWindowClass.java

import java.awt.Color;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

import java.util.Arrays;

import javax.swing.BorderFactory;

import javax.swing.ButtonGroup;

import javax.swing.JButton;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JRadioButton;

// Main MCQ window class - continuation of program

class mcqWindowClass implements ActionListener {

    // declaring MCQ page variables

    pageFrame mcqWindow = new pageFrame();

    JPanel mcqTitlePanel = createMcqPanel(50, 69);

    JPanel mcqBody = createMcqPanel(120, 400);

    JLabel mcqTitle = createMCQLabel("Title");

    JLabel mcqQuesNum = createMCQLabel("QuesNum");

    private String[] mcqQuestionArray = {"<html>Q. What is the primary function of mitochondria in a cell?</html>",

                        "<html>Q. Which gas is responsible for the greenhouse effect on Earth?</html>",

                        "<html>Q. What is the chemical symbol for gold?</html>",

                        "<html>Q. In the context of genetics, what does DNA stand for?</html>",

                        "<html>Q. Which planet is known as the Red Planet in our solar system?</html>"};

    private String[][] mcqOptionsArrray = {{"Protein synthesis", "Energy production", "Waste elimination", "Cell division"},

                        {"Oxygen", "Nitrogen", "Carbon dioxide", "Hydrogen"},

                        {"Go", "Au", "Ag", "Ge"},

                        {"Deoxyribonucleic acid", "Ribonucleic acid", "Deoxyribose nucleotide assembly", "RNA polymerase"},

                        {"Venus", "Saturn", "Jupiter", "Mars"}};

    private *int*[] mcqUserAnswers = new *int*[5];

    private String[] mcqAnswerKey = {"Energy production", "Carbon dioxide", "Au", "Deoxyribonucleic acid", "Mars"};

    JLabel mcqQuestion = createMCQLabel("Question");

    ButtonGroup mcqOptionsGrp = new ButtonGroup();

    JRadioButton mcqOption1 = createMcqOption(110);

    JRadioButton mcqOption2 = createMcqOption(150);

    JRadioButton mcqOption3 = createMcqOption(190);

    JRadioButton mcqOption4 = createMcqOption(230);

    JButton mcqPrevButton = createMcqButton(" < Previous ", Color.red, 40);

    JButton mcqNextButton = createMcqButton(" Next > ", Color.green, 400);

*int* quesNum = 0;

    ArrayList<JRadioButton> mcqOptionsButtonList = new ArrayList<>();

    mcqWindowClass() {

        setMcqQuestion();

        addOptionButtons();

        mcqTitlePanel.add(mcqTitle);

        mcqTitlePanel.add(mcqQuesNum);

        mcqBody.add(mcqQuestion);

        mcqBody.add(mcqOption1);

        mcqBody.add(mcqOption2);

        mcqBody.add(mcqOption3);

        mcqBody.add(mcqOption4);

        mcqBody.add(mcqPrevButton);

        mcqBody.add(mcqNextButton);

        mcqPrevButton.addActionListener(this);

        mcqNextButton.addActionListener(this);

        mcqWindow.add(mcqTitlePanel);

        mcqWindow.add(mcqBody);

        mcqWindow.setVisible(true);

    }

    // functions to create the MCQ page components

    private JLabel createMCQLabel(String *type*) {

        JLabel label = new JLabel();

        if (*type*.equals("Title")) {

            label.setText("MCQ Quiz");

            label.setForeground(new Color(0x000000));

            label.setFont(new Font("Open Sans", Font.BOLD, 35));

            label.setBounds(30, -2, 200, 75);

        } else if (*type*.equals("QuesNum")) {

            label.setForeground(new Color(0x9295A0));

            label.setFont(new Font("Open Sans", Font.PLAIN, 15));

            label.setBounds(540, 25, 50, 20);

        } else if (*type*.equals("Question")) {

            label.setForeground(new Color(0x27282B));

            label.setFont(new Font("DM Sans", Font.PLAIN, 25));

            label.setBounds(25, 25, 550, 60);

        } else {

        }

        return label;

    }

    private *void* addOptionButtons() {

        mcqOptionsGrp.add(mcqOption1);

        mcqOptionsGrp.add(mcqOption2);

        mcqOptionsGrp.add(mcqOption3);

        mcqOptionsGrp.add(mcqOption4);

        mcqOptionsButtonList.add(mcqOption1);

        mcqOptionsButtonList.add(mcqOption2);

        mcqOptionsButtonList.add(mcqOption3);

        mcqOptionsButtonList.add(mcqOption4);

        Arrays.fill(mcqUserAnswers, -1);

    }

    private JPanel createMcqPanel(*int* *y\_pos*, *int* *height*) {

        JPanel panel = new JPanel();

        panel.setBackground(new Color(0xFFFFFF));

        panel.setBounds(100, *y\_pos*, 600, *height*);

        panel.setLayout(null);

        panel.setOpaque(true);

        return panel;

    }

    private JRadioButton createMcqOption(*int* *y\_pos*) {

        JRadioButton radioButton = new JRadioButton();

        radioButton.setFont(new Font("Alegreya Sans", Font.PLAIN, 17));

        radioButton.setFocusable(false);

        radioButton.setBounds(60, *y\_pos*, 300, 25);

        radioButton.setContentAreaFilled(false);

        radioButton.setBorderPainted(false);

        radioButton.setOpaque(false);

        return radioButton;

    }

    private JButton createMcqButton (String *text*, Color *color*, *int* *x\_pos*) {

        JButton button = new JButton();

        button.setText(*text*);

        button.setForeground(new Color(0xFFFFFF));

        button.setBackground(*color*);

        button.setFont(new Font("Sofia Sans", Font.BOLD, 20));

        button.setBounds(*x\_pos*, 330, 150, 40);

        button.setFocusPainted(false);

        button.setBorder(BorderFactory.createBevelBorder(0));

        return button;

    }

    @*Override*

    public *void* actionPerformed(ActionEvent *e*) {

        updateUserAnswers();

        if (*e*.getSource() == mcqNextButton) {

            if (quesNum != 4) {

                quesNum++;

                setMcqQuestion();

            } else {

*int* choice = JOptionPane.showConfirmDialog(mcqWindow, "Submit Quiz?",

                null, JOptionPane.OK\_CANCEL\_OPTION);

                if (choice == 0) {

                    String[] userAnswers = new String[5];

                    for (*int* i = 0; i < 5; i++) {

                        if (mcqUserAnswers[i] == -1) {

                            userAnswers[i] = "";

                        } else {

                            userAnswers[i] = mcqOptionsArrray[i][mcqUserAnswers[i]];

                        }

                    }

                    // removes MCQ window and runs the result page

                    mcqWindow.dispose();

                    new resultWindowClass(userAnswers , mcqAnswerKey);

                }

            }

        }

        if (*e*.getSource() == mcqPrevButton && quesNum != 0) {

            quesNum--;

            setMcqQuestion();

        }

        if (mcqUserAnswers[quesNum] == -1) {

            mcqOptionsGrp.clearSelection();

        } else {

            mcqOptionsButtonList.get(mcqUserAnswers[quesNum]).setSelected(true);

        }

    }

    // changes the question options based on the user input

    private *void* setMcqQuestion() {

        mcqQuesNum.setText("(" + (quesNum+1) + " of 5)");

        mcqQuestion.setText(mcqQuestionArray[quesNum]);

        mcqOption1.setText(mcqOptionsArrray[quesNum][0]);

        mcqOption2.setText(mcqOptionsArrray[quesNum][1]);

        mcqOption3.setText(mcqOptionsArrray[quesNum][2]);

        mcqOption4.setText(mcqOptionsArrray[quesNum][3]);

    }

    // stores the user answers

    private *void* updateUserAnswers() {

        if (mcqOption1.isSelected()) {

            mcqUserAnswers[quesNum] = 0;

        } else if (mcqOption2.isSelected()) {

            mcqUserAnswers[quesNum] = 1;

        } else if (mcqOption3.isSelected()) {

            mcqUserAnswers[quesNum] = 2;

        } else if (mcqOption4.isSelected()) {

            mcqUserAnswers[quesNum] = 3;

        }

    }

}

resultWindowClass.java

import java.awt.Color;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.Image;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import javax.imageio.ImageIO;

import javax.swing.ImageIcon;

import javax.swing.JLabel;

import javax.swing.JPanel;

// Main result window class - continuation of program

class resultWindowClass {

    // declaring variables

    private String[] userAnswers;

    private String[] answerKey;

    pageFrame resultWindow = new pageFrame();

    JPanel resultbody = createResultPanel();

    JLabel resultTitle = createResultLabel("scoreTitle");

    JLabel scoreLabel = createResultLabel("score");

    private *int* score = 0;

    private *boolean* checkAnswers[] = new *boolean*[5];

    JPanel ansPanel = createAnsPanel(125, 425, 2);

    JPanel ansIconPanel = createAnsPanel(50, 75, 1);

    JLabel userAnsLabel = createAnsLabel("Your answers");

    JLabel correctAnsLabel = createAnsLabel("Correct answers");

    resultWindowClass(String[] *userAnswers*, String[] *answerKey*) {

        this.userAnswers = *userAnswers*;

        this.answerKey = *answerKey*;

        score = calculateScore();

        scoreLabel.setText(score +  "/" + *answerKey*.length);

        resultbody.add(resultTitle);

        resultbody.add(scoreLabel);

        resultbody.add(ansPanel);

        resultbody.add(ansIconPanel);

        ansPanel.add(userAnsLabel);

        ansPanel.add(correctAnsLabel);

        ansIconPanel.add(new JLabel());

        addAnswersToPanel();

        resultWindow.add(resultbody);

        resultWindow.setVisible(true);

    }

    // functions to create swing components

    private JPanel createResultPanel() {

        JPanel panel = new JPanel();

        panel.setBackground(new Color(0xFFFFFF));

        panel.setBounds(90, 75, 600, 400);

        panel.setLayout(null);

        panel.setOpaque(true);

        return panel;

    }

    private JLabel createResultLabel(String *text*) {

        JLabel label = new JLabel();

        if (*text*.equals("scoreTitle")) {

            label.setText("Score:");

            label.setForeground(new Color(0x000000));

            label.setFont(new Font("Open Sans", Font.PLAIN, 50));

            label.setBounds(175, 30, 200, 50);

        } else if (*text*.equals("score")) {

            label.setForeground(new Color(0x000000));

            label.setFont(new Font("Open Sans", Font.BOLD, 50));

            label.setBounds(330, 30, 100, 50);

        } else {

        }

        return label;

    }

    private JPanel createAnsPanel(*int* *x\_pos*, *int* *width*, *int* *col*) {

        JPanel panel = new JPanel();

        panel.setBounds(*x\_pos*, 90, *width*, 275);

        panel.setLayout(new GridLayout(6, *col*, 5, 5));

        panel.setOpaque(false);

        return panel;

    }

    private *int* calculateScore() {

*int* x = 0;

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

            if (userAnswers[i].equals(answerKey[i])) {

                checkAnswers[i] = true;

                x++;

            }

        }

        return x;

    }

    private JLabel createAnsLabel(String *text*) {

        JLabel label = new JLabel(*text*);

        label.setForeground(new Color(0x000000));

        label.setFont(new Font("DM Sans", Font.BOLD, 22));

        label.setHorizontalAlignment(JLabel.CENTER);

        label.setVerticalAlignment(JLabel.CENTER);

        return label;

    }

    // function to show user's answers and the correct answers

    private *void* addAnswersToPanel() {

        ImageIcon tickIcon = createImagePanel("Green-tick-mark-icon.jpg");

        ImageIcon crossIcon = createImagePanel("Red-cross-mark-icon.jpg");

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

            JLabel quesIcon = createAnsLabel("Q."+(i+1));

            quesIcon.setFont(new Font("DM Sans", Font.PLAIN, 15));

            if (userAnswers[i].equals(answerKey[i])) {

                quesIcon.setIcon(tickIcon);

            } else {

                quesIcon.setIcon(crossIcon);

            }

            quesIcon.setIconTextGap(20);

            ansIconPanel.add(quesIcon);

            JLabel userAns = createAnsLabel(userAnswers[i]);

            userAns.setFont(new Font("DM Sans", Font.PLAIN, 15));

            ansPanel.add(userAns);

            JLabel answer = createAnsLabel(answerKey[i]);

            answer.setFont(new Font("DM Sans", Font.PLAIN, 15));

            ansPanel.add(answer);

        }

    }

    private ImageIcon createImagePanel(String *imageName*) {

        ImageIcon icon = new ImageIcon();

        try {

            BufferedImage originalImage = ImageIO.read(new File(*imageName*));

            Image resizedImage = originalImage.getScaledInstance(25, 25, Image.SCALE\_SMOOTH);

            icon = new ImageIcon(resizedImage);

        } catch (IOException e) {

            e.printStackTrace();

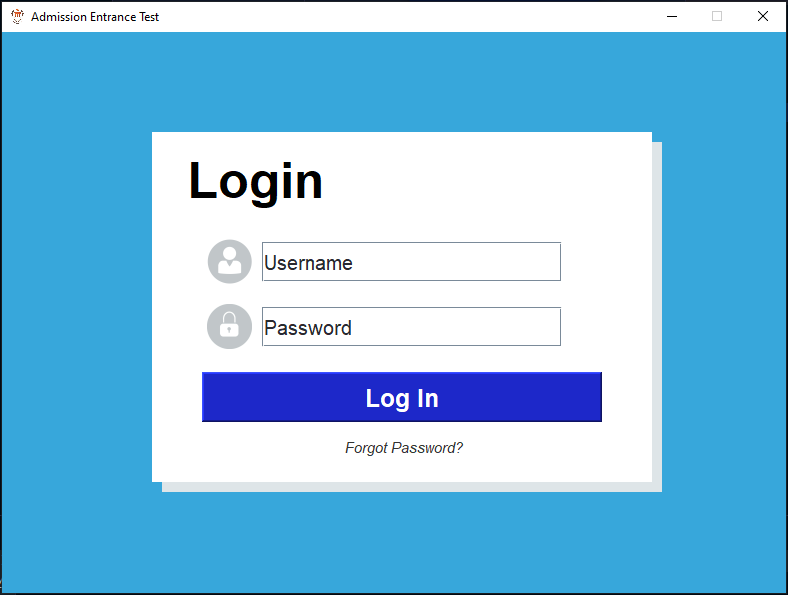
        }

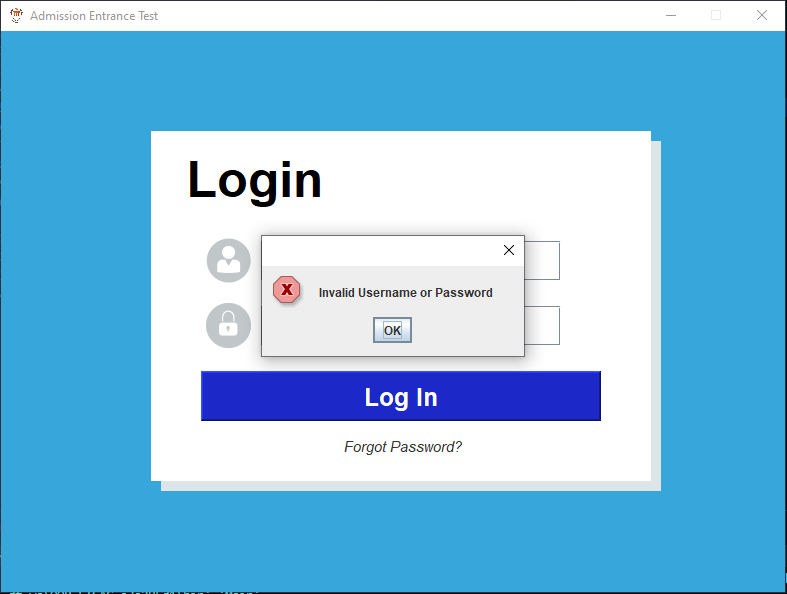
        return icon;

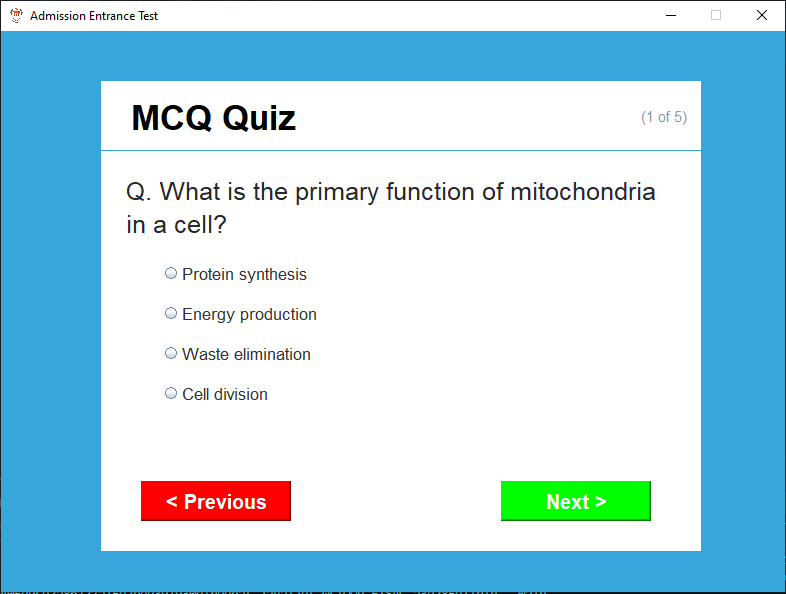
    }

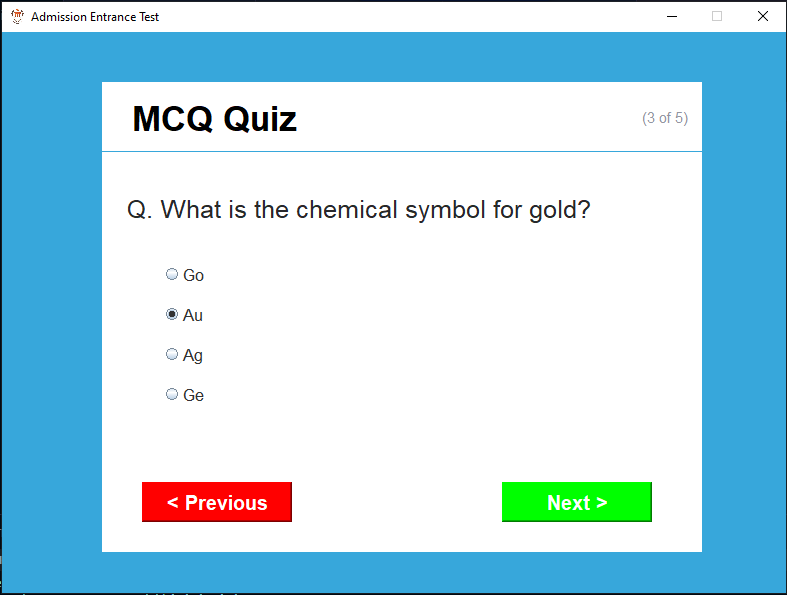
}

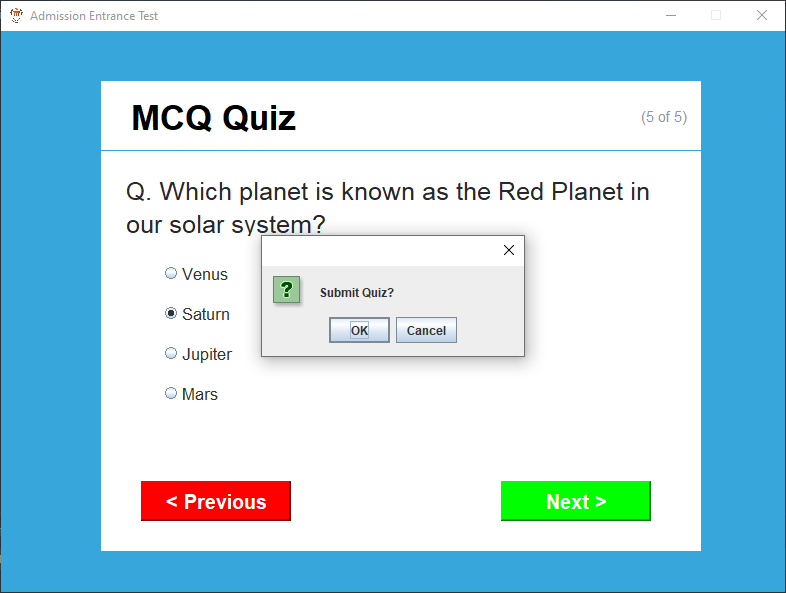
**Screenshot of Output:**

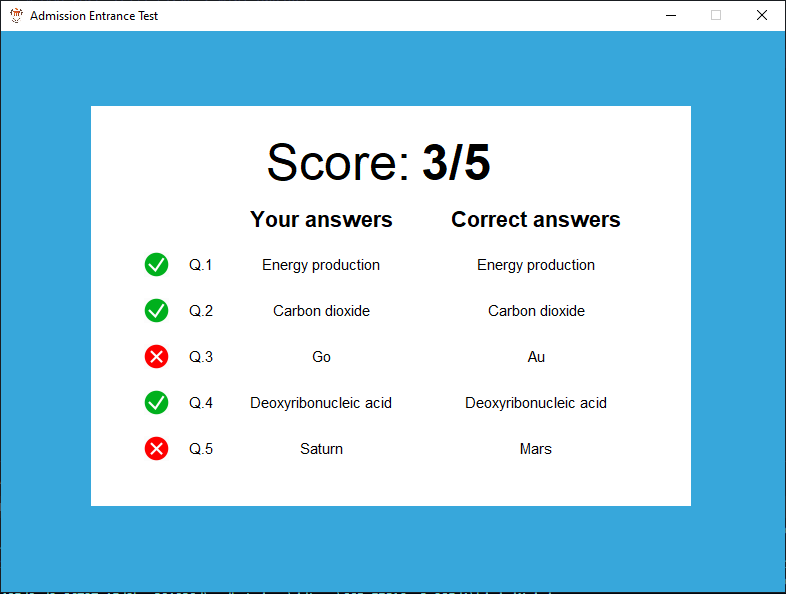
****

****

****

****

****

****

**References:**

* <https://www.youtube.com/watch?v=Kmgo00avvEw>
* <https://google.github.io/styleguide/javaguide.html>
* <https://stackoverflow.com/questions/14916255/how-to-display-a-default-message-in-jtextfield-java>
* <https://stackoverflow.com/questions/9213670/deselect-default-selection-on-jtextfield>
* <https://stackoverflow.com/questions/3025320/draw-a-jbutton-to-look-like-a-jlabel-or-at-least-without-the-button-edge>
* <https://stackoverflow.com/questions/44020638/adding-imageicon-to-jpanel>
* <https://stackoverflow.com/questions/23945251/how-do-i-import-images-using-imageicon>
* <https://stackoverflow.com/questions/41477832/how-many-jlabels-fit-in-my-jpanel>
* <https://stackoverflow.com/questions/26697266/how-to-find-that-item-is-selected-in-a-button-group>
* <https://stackoverflow.com/questions/56328804/joptionpane-showmessagedialog-un-focus-ok-button>
* <https://stackoverflow.com/questions/46553650/jlabel-not-appearing-until-frame-is-moved-out-of-the-screen-and-back-again>
* <https://stackoverflow.com/questions/8875674/setselected-a-specific-jradiobutton-in-a-buttongroup-based-on-action-command>
* <https://stackoverflow.com/questions/12604593/java-swing-how-can-i-change-the-size-of-one-row-in-a-grid-layout-and-allow-it-t>