OBJECT ORIENTED PROGRAMMING (JAVA)

23CSE111

ASSIGNMENT@1

Problem Statement:

Develop an interactive quiz game application using Java Swing, allowing users to select a domain (e.g., Science, General Knowledge, Computer Science) and answer a series of randomized questions. The application should load questions from JSON files corresponding to the selected domain, display them one by one along with multiple-choice options, and calculate the user's score upon quiz completion. The system should provide a user-friendly interface with features like navigation buttons, scoring, and real-time feedback. Additionally, it should handle exceptions gracefully to ensure smooth operation.

Algorithm:

Step -0: Set up the user interface for the quiz game window.
Step-1: Define file paths for different quiz domains (like Science, General Knowledge, Computer Science).
Step-2: Create UI components such as labels, buttons, and dropdown menus.
Step-3: Initialize variables to store questions, options, correct answers, and score.
Step-4: When the user clicks "Start," select a domain from the dropdown menu.
Step-5: Load questions from the corresponding JSON file based on the selected domain.
Step-6: Display the first question along with its options.
Step-7: Enable the "Next" button for the user to proceed to the next question.
Step-8: When the user selects an option, enable the "Next" button.
Step-9: When the user clicks "Next," display the next question.
Step-10: If all questions are answered, disable the "Next" button and enable the "Submit" button.
Step-11: When the user clicks "Submit," calculate the score based on

selected options and correct answers.
Step-12: Display the final score and the current date and time.
Step-13: Identify the option selected by the user for each question.
Step-14: Execute the main method to start the quiz application.
Step-15: Launch the application window and make it visible to the user.
Step-16: Allow users to select a domain, answer questions, and navigate through the quiz.
Step-17: Keep track of the current question index and the user's score.
Step-18: Display the final score and completion message when the quiz is over.
Step-19: Handle exceptions gracefully, such as file not found or JSON parsing errors.
Step-20: Comment the code for better understanding and maintenance.
Step-21: Test the application thoroughly under various scenarios to ensure functionality.
Step-22: End the program.

Program:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Collections;

import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import org.json.simple.parser.ParseException;

public class QuizPresentation2 extends JFrame {
    private JLabel questionLabel;
    private List<JRadioButton> optionButtons;
    private ButtonGroup optionButtonGroup;
    private JButton startButton;
    private JButton nextButton;
    private JButton exitButton;
    private JButton submitButton;

    private Map<String, String> domainFilePaths;

    private List<Map<String, Object>> questions;
```

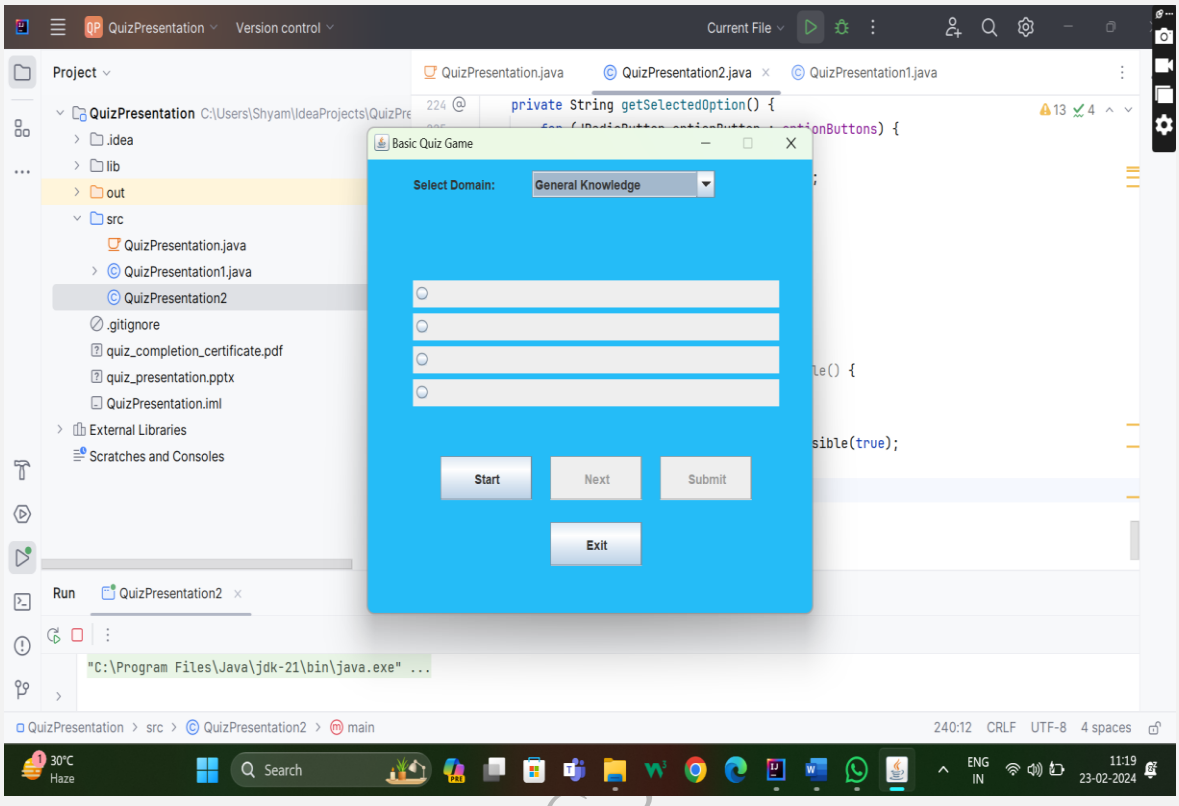
```
    private int questionIndex;
    private int score;

    public QuizPresentation2() {
        setTitle("Basic Quiz Game");
        setSize(500, 450);
        setLocationRelativeTo(null);
        setResizable(false);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        domainFilePaths = new HashMap<>();
        domainFilePaths.put("Science",
"C:\\Users\\Shyam\\Desktop\\newpython\\GUI python\\Science.json");
        domainFilePaths.put("General Knowledge",
"C:\\Users\\Shyam\\Desktop\\newpython\\GUI python\\General
Knowledge.json");
        domainFilePaths.put("Computer Science",
"C:\\Users\\Shyam\\Desktop\\newpython\\GUI
python\\Computerscience.json");

        initComponents();
    }

    private void initComponents() {
        JPanel mainPanel = new JPanel();
        mainPanel.setLayout(null);
        mainPanel.setBackground(new Color(37, 188, 247));
        add(mainPanel);

        JLabel domainLabel = new JLabel("Select Domain:");
        domainLabel.setBounds(50, 10, 150, 25);
        mainPanel.add(domainLabel);

        String[] domainOptions = {"General Knowledge", "Science",
"Computer Science"};
        JComboBox<String> domainComboBox = new
JComboBox<>(domainOptions);
        domainComboBox.setBounds(180, 10, 200, 25);
        mainPanel.add(domainComboBox);

        questionLabel = new JLabel();
        questionLabel.setBounds(50, 50, 400, 50);
        questionLabel.setHorizontalAlignment(SwingConstants.CENTER);
        mainPanel.add(questionLabel);

        optionButtons = new ArrayList<>();
        optionButtonGroup = new ButtonGroup();
        for (int i = 0; i < 4; i++) {
            JRadioButton optionButton = new JRadioButton();
            optionButton.setBounds(50, 110 + i * 30, 400, 25);
            optionButtonGroup.add(optionButton);
            optionButtons.add(optionButton);
            mainPanel.add(optionButton);

            optionButton.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    nextButton.setEnabled(true);
                }
            });
        }
```

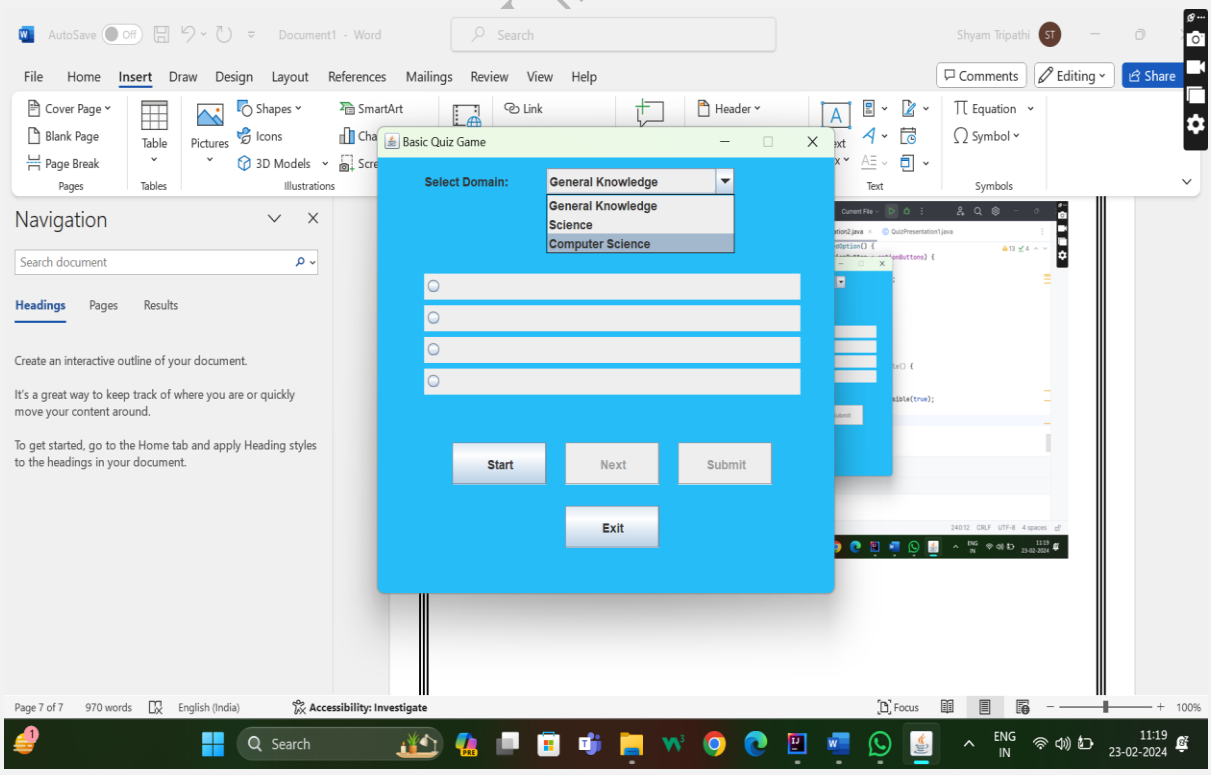
```
        startButton = new JButton("Start");
        startButton.setBounds(80, 270, 100, 40);
        mainPanel.add(startButton);

        nextButton = new JButton("Next");
        nextButton.setBounds(200, 270, 100, 40);
        nextButton.setEnabled(false);
        mainPanel.add(nextButton);

        submitButton = new JButton("Submit");
        submitButton.setBounds(320, 270, 100, 40);
        submitButton.setEnabled(false);
        mainPanel.add(submitButton);

        exitButton = new JButton("Exit");
        exitButton.setBounds(200, 330, 100, 40);
        mainPanel.add(exitButton);

        startButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String selectedDomain = (String)
domainComboBox.getSelectedItem();
                startQuiz(selectedDomain);
            }
        });

        nextButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                nextQuestion();
            }
        });

        submitButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                submitQuiz();
            }
        });

        exitButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                dispose();
            }
        });
    }


    private void startQuiz(String selectedDomain) {
        String filePath = domainFilePaths.get(selectedDomain);
        if (filePath == null) {
            JOptionPane.showMessageDialog(this, "No file path found
for " + selectedDomain, "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        try {
            JSONParser parser = new JSONParser();
```

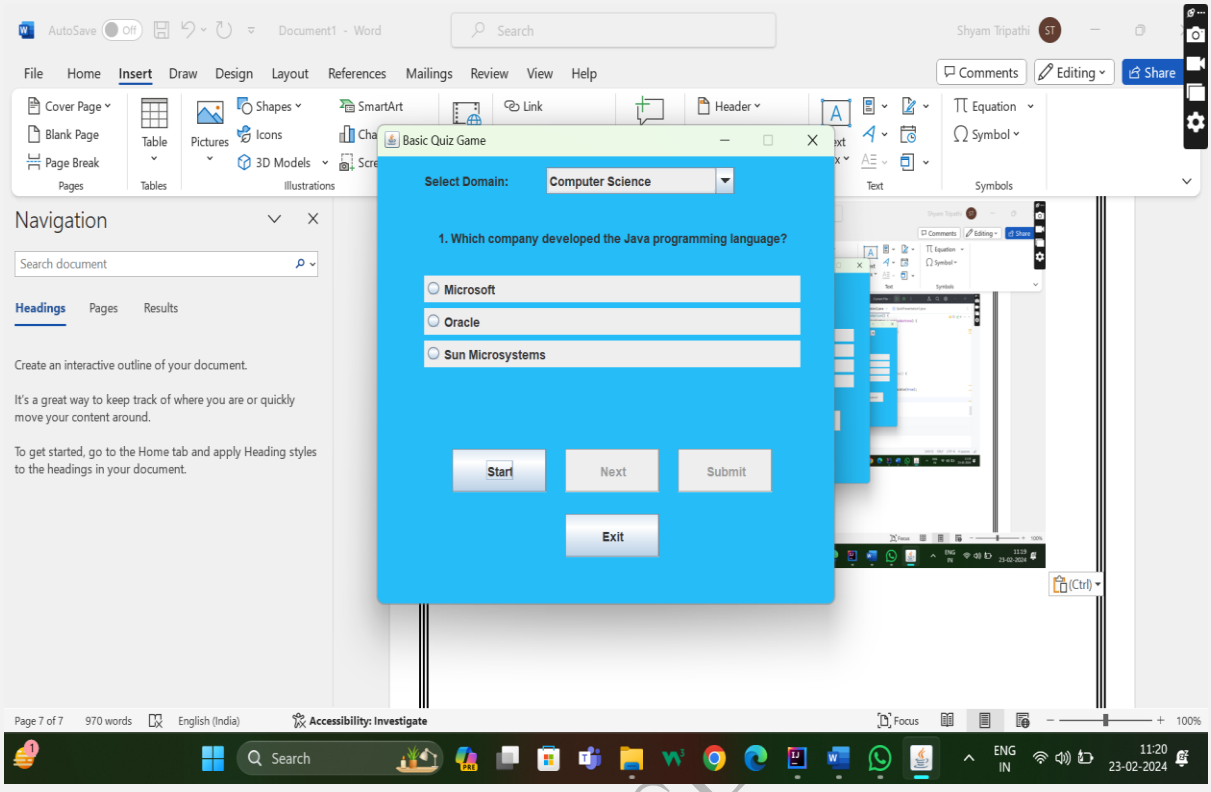
```
            JSONObject jsonData = (JSONObject) parser.parse(new
FileReader(filePath));

            JSONArray jsonArray = (JSONArray)
jsonData.get("questions");
            questions = new ArrayList<>();
            for (Object obj : jsonArray) {
                JSONObject jsonObj = (JSONObject) obj;
                Map<String, Object> question = new HashMap<>();
                question.put("question", jsonObj.get("question"));
                question.put("options", jsonObj.get("options"));
                question.put("correctAnswer",
jsonObj.get("correctAnswer"));
                questions.add(question);
            }

            Collections.shuffle(questions);
            questions = questions.subList(0, Math.min(15,
questions.size()));

            questionIndex = 0;
            score = 0;
            nextQuestion();
        } catch (IOException | ParseException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(this, "Error reading JSON
file", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }


    private void nextQuestion() {
        if (questionIndex < questions.size()) {
            Map<String, Object> question =
questions.get(questionIndex);
            questionIndex++;
            questionLabel.setText(questionIndex + ". " +
question.get("question"));

            List<String> options = (List<String>)
question.get("options");
            for (int i = 0; i < optionButtons.size(); i++) {
                JRadioButton optionButton = optionButtons.get(i);
                if (i < options.size()) {
                    optionButton.setText(options.get(i));
                    optionButton.setVisible(true);
                } else {
                    optionButton.setVisible(false);
                }
            }

            if (questionIndex == 15) {
                nextButton.setEnabled(false);
                submitButton.setEnabled(true);
            } else {
                nextButton.setEnabled(false);
            }
        } else {
//            JOptionPane.showMessageDialog(  "Quiz Over",
JOptionPane.INFORMATION_MESSAGE);
            nextButton.setEnabled(false);
```

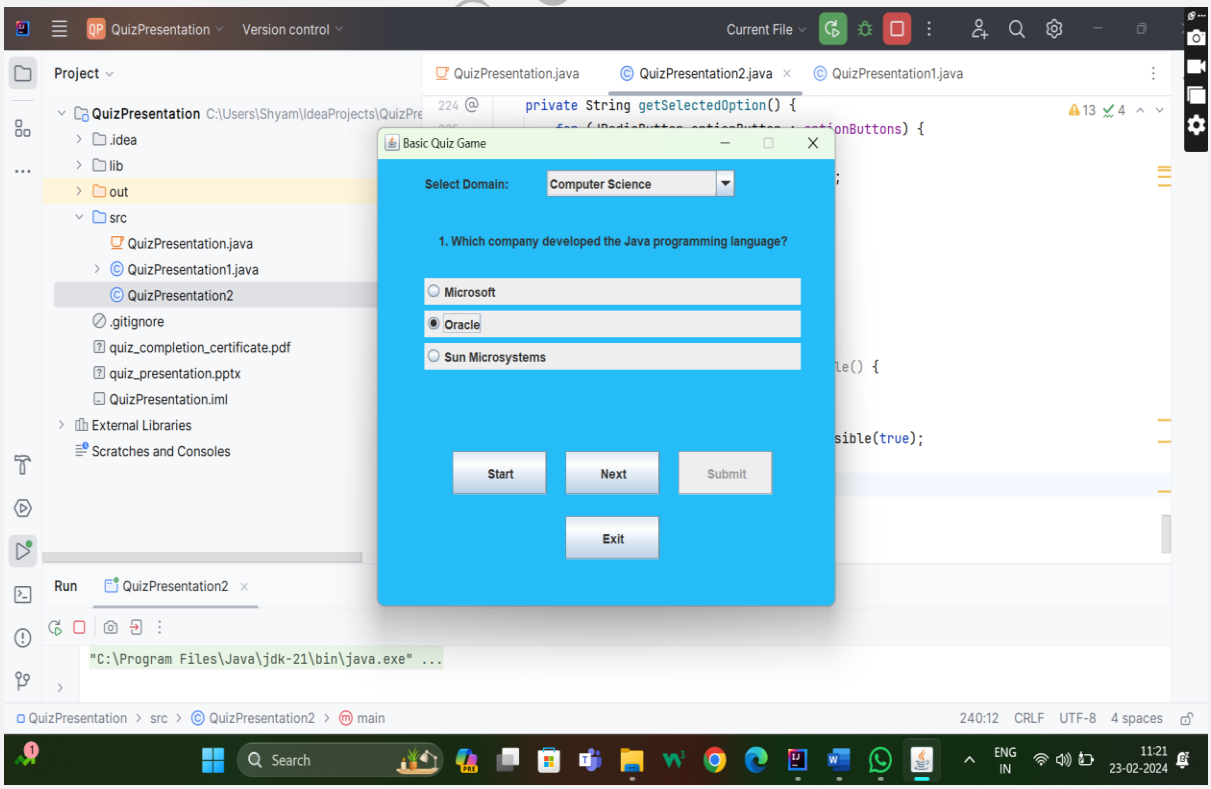
```
                submitButton.setEnabled(true);
            }
        }


        private void submitQuiz() {
            int totalQuestions = questions.size();
            int correctAnswers = 0;

            for (int i = 0; i < totalQuestions; i++) {
                Map<String, Object> question = questions.get(i);
                String selectedOption = getSelectedOption();
                String correctOption = ((String)
question.get("correctAnswer")).toLowerCase();
                if (selectedOption != null &&
selectedOption.toLowerCase().equals(correctOption)) {
                    correctAnswers++;
                }
            }

            score = (int) Math.round((double) correctAnswers /
totalQuestions * 100);

            LocalDateTime now = LocalDateTime.now();
            DateTimeFormatter formatter =
DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
            String dateTime = now.format(formatter);
            JOptionPane.showMessageDialog(this, "Your final score is: " +
score  + "\nDate and Time: " + dateTime, "Quiz Over",
JOptionPane.INFORMATION_MESSAGE);
        }

        private String getSelectedOption() {
            for (JRadioButton optionButton : optionButtons) {
                if (optionButton.isSelected()) {
                    return optionButton.getText();
                }
            }
            return null;
        }


        public static void main(String[] args) {
            SwingUtilities.invokeLater(new Runnable() {
                @Override
                public void run() {
                    new QuizPresentation2().setVisible(true);
                }
            });
        }
}
```

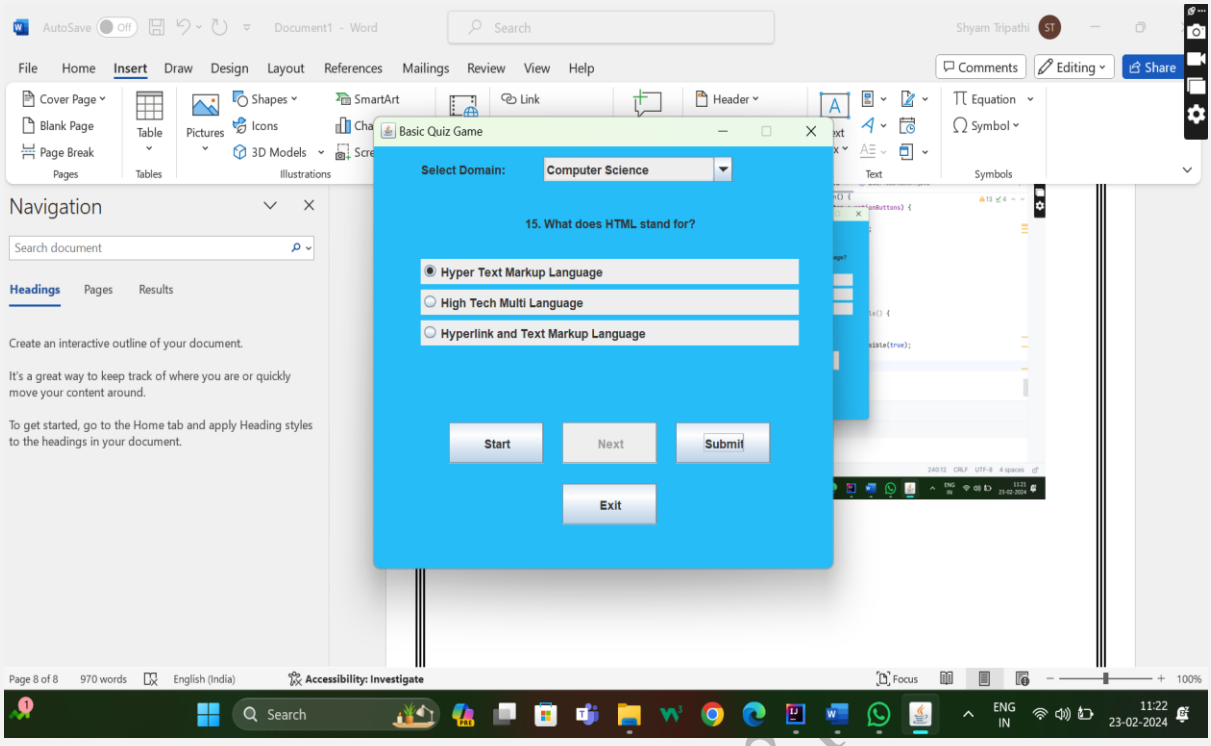
Output Result:

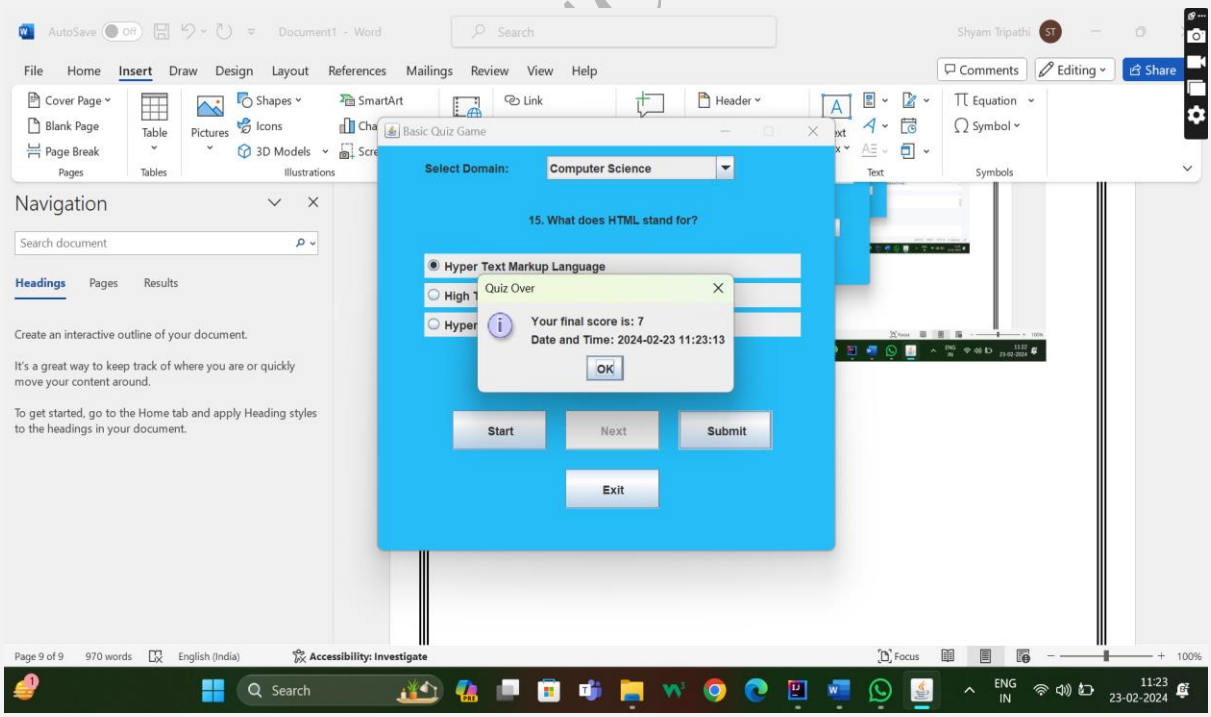





Basic Quiz Game
Select Domain:
Computer Science
1. Which company developed the Java programming language?
Microsoft
Oracle
Sun Microsystems
Start
Next
Submit
Exit


Basic Quiz Game
Select Domain:
Computer Science
1. Which company developed the Java programming language?
Microsoft
Oracle
Sun Microsystems
Start
Next
Submit
Exit


Basic Quiz Game
Select Domain:
Computer Science
15. What does HTML stand for?
Hyper Text Markup Language
High Tech Multi Language
Hyperlink and Text Markup Language
Start
Next
Submit
Exit


Basic Quiz Game
Select Domain:
Computer Science
15. What does HTML stand for?
Hyper Text Markup Language
Quiz Over
Your final score is: 7
Date and Time: 2024-02-23 11:23:13
OK
Start
Next
Submit
Exit