JAWAHARLAL NEHRU ENGINEERING COLLEGE

(Mahatma Gandhi Mission University, Chatrapati Sambhaji Nagar)

JAVA PROJECT REPORT

For Second year

On

"QUIZ APPLICATION"

Submitted by:

Disha Yogesh Supekar (202101103154)

Vedshree Rajesh Kulkarni (202101103149)

Subject In charge:

Prof. Sushma Ghode (Assistant professor CSE, JNEC)

Department of Computer Science and Engineering

Academic year 2022-2023

TABLE OF CONTENTS

1. **INTRODUCTION:**

This is a Java program that implements a simple multiple choice quiz game using Swing for GUI components. The program displays 10 questions, each with four answer choices. The user selects an answer by clicking one of the four buttons labelled "A", "B", "C", or "D". The program keeps track of the number of correct answers and displays the result at the end of the quiz.

The program uses an array of strings to store the questions, an array of string arrays to store the answer choices for each question, and an array of characters to store the correct answers. The ActionListener interface is implemented to handle button clicks and check the user's answer against the correct answer. A Timer is used to limit the time for each question to 10 seconds.

The program creates a JFrame and adds several components to it using absolute positioning. The components include a JTextField to display the question, a JTextArea to display the answer choices, four JButtons labeled "A", "B", "C", and "D" for the user to select an answer, and four JLabels to display the answer choices for each button. The program also displays a JLabel to show the time left for each question, and a JTextField to show the number of correct answers and the percentage of correct answers.

2. **OBJECTIVE:**

The objective is to implement a simple quiz application in Java using GUI components such as JFrame, JTextField, JTextArea, JButton, JLabel, and Timer. The quiz consists of ten multiple-choice questions with four options each, and the user must select one of the options by clicking on the corresponding button. The application also includes a timer that limits the time available to answer each question and displays the final score of the user at the end of the quiz. The code implements the ActionListener interface to handle the button click events and update the UI components accordingly.

3. **BACKGROUND:**

The history of quiz apps can be traced back to the invention of personal computers and the development of computer-based training programs in the 1960s and 1970s. These early programs were often used in educational and corporate settings to teach employees or students new skills or knowledge.

With the rise of the internet in the 1990s and the widespread adoption of smartphones in the 2000s, quiz apps became increasingly popular as a form of entertainment and education. Many popular quiz apps, such as "Trivia Crack" and "HQ Trivia," have been downloaded millions of times and have spawned a new genre of mobile gaming.

Today, quiz apps continue to evolve and adapt to changing technology and user preferences. Some apps use artificial intelligence and machine learning algorithms to tailor questions to individual users' knowledge levels, while others offer social features that allow users to compete with friends or strangers. Quiz apps are now used not only for entertainment but also for education, marketing, and research purposes.

4. **HARDWARE AND SOFTWARE REQUIREMENTS:**

Hardware Requirements:

- A computer or mobile device capable of running the programming language used to develop the app.
- Adequate storage space for the app and any necessary data (e.g., questions and answers, user information)

Software Requirements:

- An integrated development environment (IDE) or text editor for writing and testing the code (e.g., Visual Studio Code, Eclipse, NetBeans)
- A programming language runtime or interpreter installed on the system (e.g. Java)

5. CODING:

```
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;

class Quiz implements ActionListener {

    String[] questions = {
    "What is the default value of an uninitialized boolean variable in Java?",
    "Which collection class in Java allows duplicate elements?",
    "Which keyword is used to prevent a class from being inherited in Java?",
    "What is the purpose of the "finally" block in Java exception handling?"
    "Which of the following is not a valid access modifier in Java?",
    "Which method is used to read input from the keyboard in Java?"
    "Which keyword is used to define a subclass in Java?",
    "How many primitive data types are there in Java?",
    "Which of the following operators is used to create an object in Java?",
    "Which of the following is a superclass of all Java classes?"
    };

    String[][] options = {
        {"true","false","0","null"},
        {"HashSet"," TreeSet"," LinkedHashSet","None of the above"},
        {"final"," static","abstract","private"},
        {"To catch and handle exceptions","To indicate that an error has
occurred","To provide a default case for a switch statement","To execute code
regardless of whether an exception is thrown"},
        {"public","private","protected","package-private"},
        {"System.in()","Console.read()","Scanner.nextLine()","
BufferedReader.readLine()"},
        {"extends"," implements","super","this"},
        {"6","7","8","9"},
        {"new","create"," make","build"},
{"Object","Class","String","System"}

    };

    char[] answers = {'D', 'A', 'A', 'D', 'D', 'A', 'A', 'C', 'A', 'A'};

    char guess;
```

```
char answer;
int index;
int correct_guesses = 0;
int total_questions = questions.length;
int result;
int seconds = 10;

JFrame frame = new JFrame();
JTextField textfield = new JTextField();
JTextArea textarea = new JTextArea();
JButton buttonA = new JButton();
JButton buttonB = new JButton();
JButton buttonC = new JButton();
JButton buttonD = new JButton();
JLabel answer_labelA = new JLabel();
JLabel answer_labelB = new JLabel();
JLabel answer_labelC = new JLabel();
JLabel answer_labelD = new JLabel();
JLabel time_label = new JLabel();
JLabel seconds_left = new JLabel();
JTextField number_right = new JTextField();
JTextField percentage = new JTextField();

Timer timer = new Timer(1000, new ActionListener() {
	@Override
	public void actionPerformed(ActionEvent e) {
		seconds--;
		seconds_left.setText(String.valueOf(seconds));
		if(seconds<=0) {
			displayAnswer();
		}
	}
});

public Quiz() {
	frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	frame.setSize(650,650);
	frame.getContentPane().setBackground(new Color(50,50,50));
	frame.setLayout(null);
	frame.setResizable(false);
```

```
textfield.setBounds(0,0,650,100);
textfield.setBackground(new Color(25,25,25));
textfield.setForeground(new Color(0,128,128));
textfield.setFont(new Font("Arial",Font.BOLD,50));
textfield.setBorder(BorderFactory.createBevelBorder(1));
textfield.setHorizontalAlignment(JTextField.CENTER);
textfield.setEditable(false);

textarea.setBounds(0,100,650,100);
textarea.setLineWrap(true);
textarea.setWrapStyleWord(true);
textarea.setBackground(new Color(25,25,25));
textarea.setForeground(new Color(225,225,0));
textarea.setFont(new Font("Arial",Font.BOLD,40));
textarea.setBorder(BorderFactory.createBevelBorder(1));
textarea.setEditable(false);

buttonA.setBounds(0,200,100,50);
buttonA.setFont(new Font("Arial",Font.BOLD,25));
buttonA.setFocusable(false);
buttonA.addActionListener(this);
buttonA.setText("A");

buttonB.setBounds(0,250,100,50);
buttonB.setFont(new Font("Arial",Font.BOLD,25));
buttonB.setFocusable(false);
buttonB.addActionListener(this);
buttonB.setText("B");

buttonC.setBounds(0,300,100,50);
buttonC.setFont(new Font("Arial",Font.BOLD,25));
buttonC.setFocusable(false);
buttonC.addActionListener(this);
buttonC.setText("C");

buttonD.setBounds(0,350,100,50);
buttonD.setFont(new Font("Arial",Font.BOLD,25));
buttonD.setFocusable(false);
buttonD.addActionListener(this);
```

```
buttonD.setText("D");

answer_labelA.setBounds(125,200,500,50);
answer_labelA.setBackground(new Color(50,50,50));
answer_labelA.setForeground(new Color(25,255,0));
answer_labelA.setFont(new Font("Arial",Font.PLAIN,25));

answer_labelB.setBounds(125,250,500,50);
answer_labelB.setBackground(new Color(50,50,50));
answer_labelB.setForeground(new Color(25,255,0));
answer_labelB.setFont(new Font("Arial",Font.PLAIN,25));

answer_labelC.setBounds(125,300,500,50);
answer_labelC.setBackground(new Color(50,50,50));
answer_labelC.setForeground(new Color(25,255,0));
answer_labelC.setFont(new Font("Arial",Font.PLAIN,25));

answer_labelD.setBounds(125,350,500,50);
answer_labelD.setBackground(new Color(50,50,50));
answer_labelD.setForeground(new Color(25,255,0));
answer_labelD.setFont(new Font("Arial",Font.PLAIN,25));

seconds_left.setBounds(535,510,100,100);
seconds_left.setBackground(new Color(25,25,25));
seconds_left.setForeground(new Color(255,0,0));
seconds_left.setFont(new Font("Arial",Font.BOLD,60));
seconds_left.setBorder(BorderFactory.createBevelBorder(1));
seconds_left.setOpaque(true);
seconds_left.setHorizontalAlignment(JTextField.CENTER);
seconds_left.setText(String.valueOf(seconds));

time_label.setBounds(535,475,100,25);
time_label.setBackground(new Color(50,50,50));
time_label.setForeground(new Color(255,0,0));
time_label.setFont(new Font("Arial",Font.BOLD,20));
time_label.setHorizontalAlignment(JTextField.CENTER);
time_label.setText("Timer");

number_right.setBounds(225,225,200,100);
number_right.setBackground(new Color(25,25,25));
```

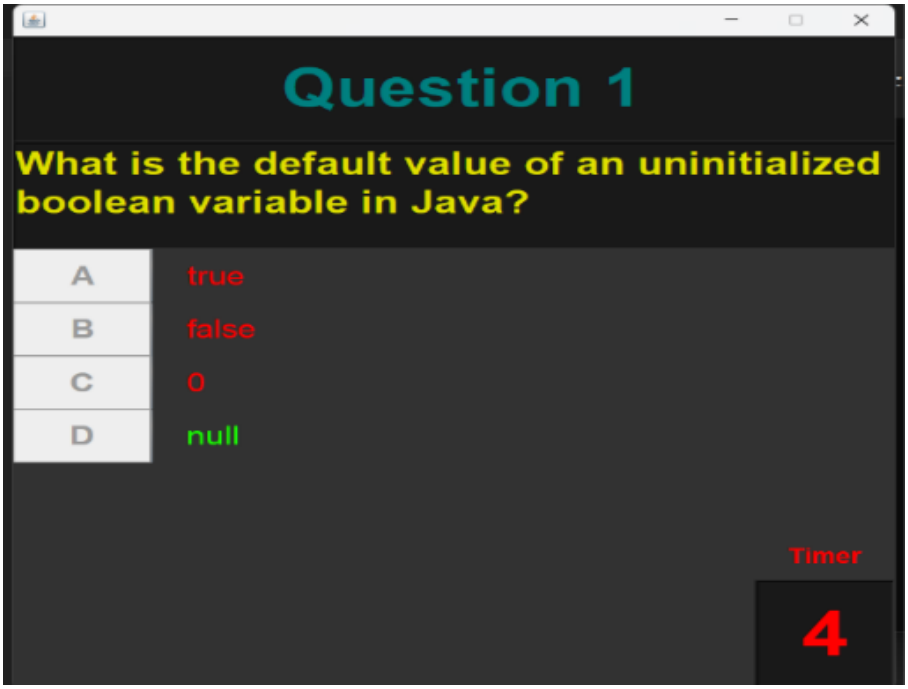
```
        number_right.setForeground(new Color(255,165,0));
        number_right.setFont(new Font("Arial",Font.BOLD,50));
        number_right.setBorder(BorderFactory.createBevelBorder(1));
        number_right.setHorizontalAlignment(JTextField.CENTER);
        number_right.setEditable(false);

        percentage.setBounds(225,325,200,100);
        percentage.setBackground(new Color(25,25,25));
        percentage.setForeground(new Color(255,165,0));
        percentage.setFont(new Font("Arial",Font.BOLD,50));
        percentage.setBorder(BorderFactory.createBevelBorder(1));
        percentage.setHorizontalAlignment(JTextField.CENTER);
        percentage.setEditable(false);

        frame.add(time_label);
        frame.add(seconds_left);
        frame.add(answer_labelA);
        frame.add(answer_labelB);
        frame.add(answer_labelC);
        frame.add(answer_labelD);
        frame.add(buttonA);
        frame.add(buttonB);
        frame.add(buttonC);
        frame.add(buttonD);
        frame.add(textarea);
        frame.add(textfield);
        frame.setVisible(true);

        nextQuestion();
}

public void nextQuestion() {
        if(index>=total_questions) {
                results();
        }
        else {
                textfield.setText("Question "+(index+1));
                textarea.setText(questions[index]);
                answer_labelA.setText(options[index][0]);
                answer_labelB.setText(options[index][1]);
```

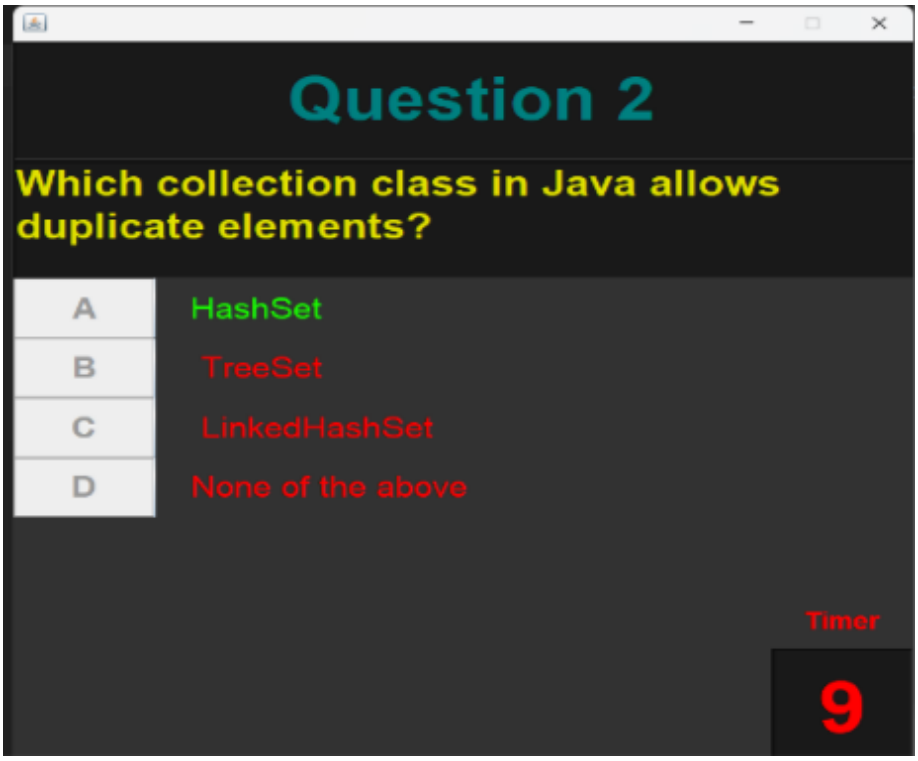
```
                answer_labelC.setText(options[index][2]);
                answer_labelD.setText(options[index][3]);
                timer.start();
        }
}

@Override
public void actionPerformed(ActionEvent e) {

        buttonA.setEnabled(false);
        buttonB.setEnabled(false);
        buttonC.setEnabled(false);
        buttonD.setEnabled(false);

        if(e.getSource()==buttonA) {
                answer= 'A';
                if(answer == answers[index]) {
                        correct_guesses++;
                }
        }
        if(e.getSource()==buttonB) {
                answer= 'B';
                if(answer == answers[index]) {
                        correct_guesses++;
                }
        }
        if(e.getSource()==buttonC) {
                answer= 'C';
                if(answer == answers[index]) {
                        correct_guesses++;
                }
        }
        if(e.getSource()==buttonD) {
                answer= 'D';
                if(answer == answers[index]) {
                        correct_guesses++;
                }
        }
        displayAnswer();
}
```

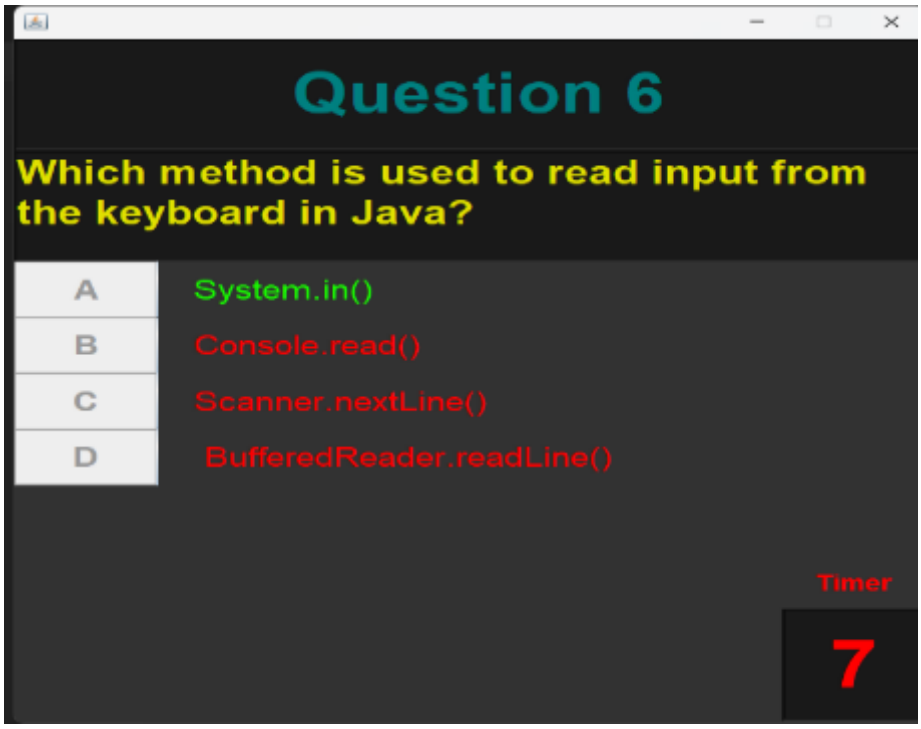
```
public void displayAnswer() {

        timer.stop();

        buttonA.setEnabled(false);
        buttonB.setEnabled(false);
        buttonC.setEnabled(false);
        buttonD.setEnabled(false);

        if(answers[index] != 'A')
                answer_labelA.setForeground(new Color(255,0,0));
        if(answers[index] != 'B')
                answer_labelB.setForeground(new Color(255,0,0));
        if(answers[index] != 'C')
                answer_labelC.setForeground(new Color(255,0,0));
        if(answers[index] != 'D')
                answer_labelD.setForeground(new Color(255,0,0));

        Timer pause = new Timer(2000, new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {

                answer_labelA.setForeground(new Color(25,255,0));
                answer_labelB.setForeground(new Color(25,255,0));
                answer_labelC.setForeground(new Color(25,255,0));
                answer_labelD.setForeground(new Color(25,255,0));

                answer = ' ';
                seconds=10;
                seconds_left.setText(String.valueOf(seconds));
                buttonA.setEnabled(true);
                buttonB.setEnabled(true);
                buttonC.setEnabled(true);
                buttonD.setEnabled(true);
                index++;
                nextQuestion();
        }
        });
```

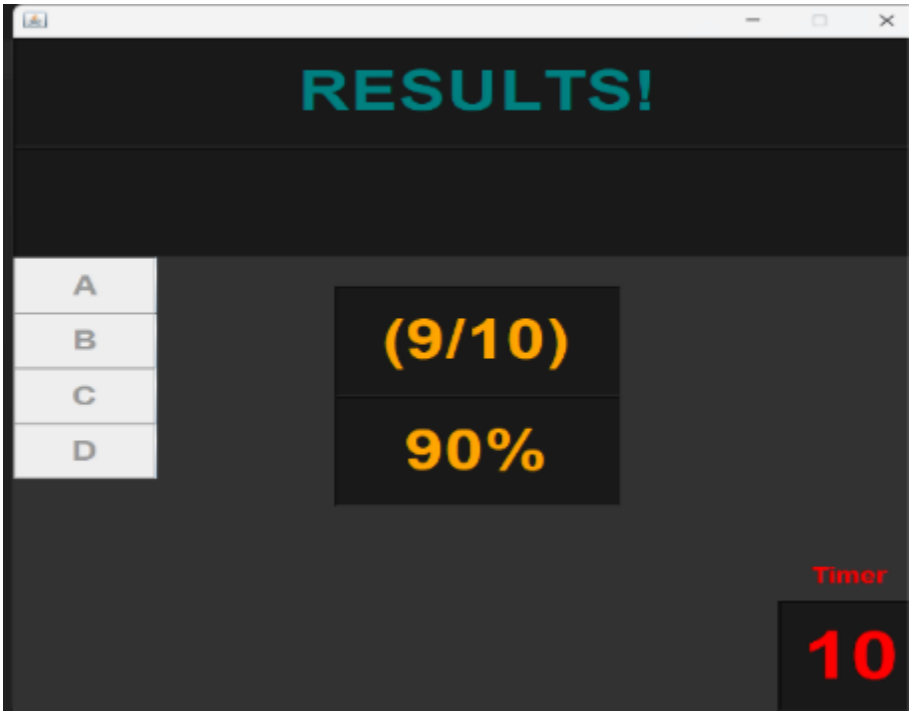
```
            pause.setRepeats(false);
            pause.start();
        }

public void results()
        {

            buttonA.setEnabled(false);
            buttonB.setEnabled(false);
            buttonC.setEnabled(false);
            buttonD.setEnabled(false);

            result = (int)((correct_guesses/(double)total_questions)*100);

            textfield.setText("RESULTS!");
            textarea.setText("");
            answer_labelA.setText("");
            answer_labelB.setText("");
            answer_labelC.setText("");
            answer_labelD.setText("");

            number_right.setText("("+correct_guesses+"/"+total_questions+")");
            percentage.setText(result+"%");

            frame.add(number_right);
            frame.add(percentage);

            }
        }

        public class Project {

            public static void main(String[] args) {

                Quiz quiz = new Quiz();

        }
        }
```

6. OUTPUT SCREENSHOT:






Question 6
Which method is used to read input from the keyboard in Java?
A
System.in()
B
Console.read()
C
Scanner.nextLine()
D
BufferedReader.readLine()
Timer
7


RESULTS!
A
B
C
D
(9/10)
90%
Timer
10

7. FUTURE SCOPE:

1. Expand the quiz categories: Consider adding more quiz categories to cater to a wider audience. For example, you could add categories such as sports, technology, history, or politics.

2. Add multimedia support: Enhance the user experience by adding images, videos, and audio files to your quiz questions. This can make the quiz more engaging and interactive.

3. Improve the scoring system: You can make your quiz app more competitive by adding a high score feature. This can encourage users to take the quiz multiple times and try to beat their own or others' high scores.

4. Implement user accounts: Consider allowing users to create accounts so that they can save their progress, view their scores, and share their results on social media.

5. Localization: Consider localizing your app to reach a wider audience. This can involve translating the app into multiple languages and adapting it to different cultural norms and practices.

6. Mobile App: You can also consider building a mobile app version of your quiz app, which can make it more accessible to users on-the-go and increase its reach.

8. CONCLUSION:

The code demonstrates the implementation of a simple multiple-choice quiz using Java Swing GUI components. The quiz consists of ten questions, and the user has ten seconds to answer each question. The user interface of the quiz consists of a text field for displaying the question, four buttons for selecting the answer, and labels for displaying the results and remaining time.

The code utilizes several Java concepts, such as arrays, timers, and event handling, to create an interactive quiz application. It also showcases the use of various Swing components to create a visually appealing user interface.

Overall, this code can serve as a starting point for creating more complex quiz applications or other interactive educational programs. With some modifications and enhancements, it could be used for a variety of purposes, such as language learning, test preparation, or gamification of learning materials.

9. REFERENCES AND BIBLIOGRAPHY:

1. BLACK BOOK:
   https://docs.google.com/file/d/0B1lryJO1ubTxbDhuZG42WEVTV0k/edit?usp=sharing.

2. GUI in Java:
   https://www.guru99.com/java-swing-gui.html

3. Swings:
   https://www.javatpoint.com/java-swing

4. Reference Video:
   https://youtu.be/utC-8xeEQQA

5. Packages in Java:
   https://www.w3schools.com/java/java_packages.asp