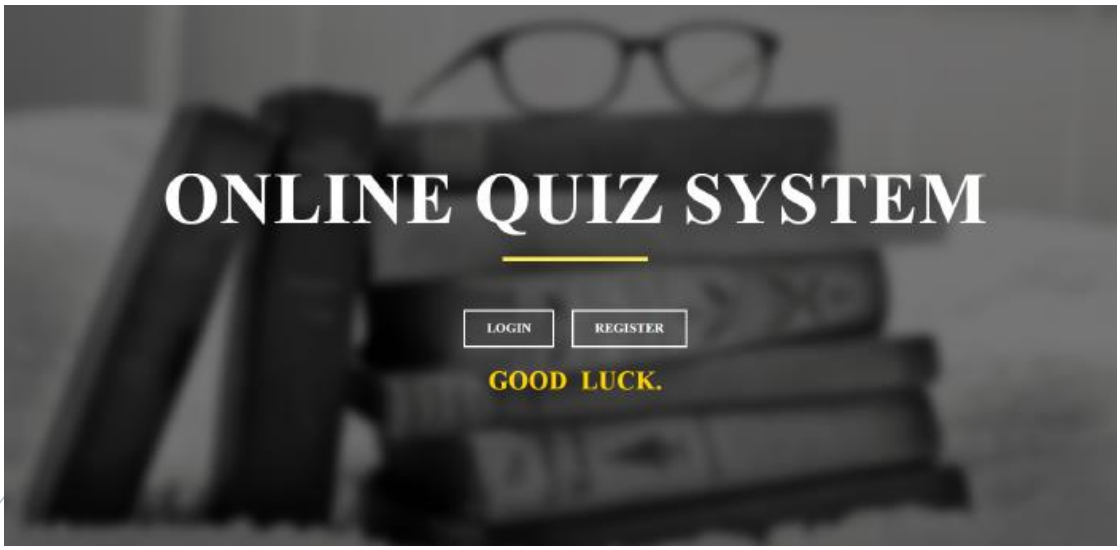
Project Report

[Online Quiz Management System]



------ Group 07 --------

G.W.Kaushalya
W.H.Sewwandi
G.V.N.D.Silva
J.A.R.Prabodha
R.V.P.I.Sewwandi

Java System Development Team

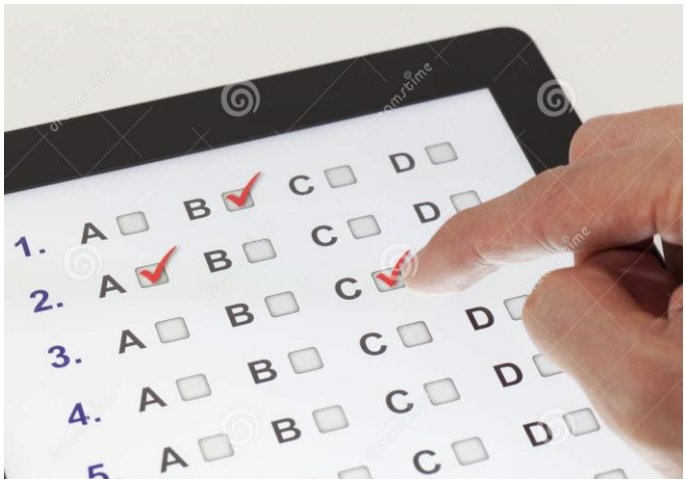
University of Kelaniya,

Project – Online Quiz Management System

CONTENT

INTRODUCTION TO THE PROJECT

Our system provides a common platform to connect teacher and student online. As well, it reduces huge work done by teacher and students because responses by the candidates will be check automatically and instantly. Furthermore, it reduces paper works and time to travel.



The users can sit at individual terminals and login to face the quiz in given duration. We are offering several types of questions and check answers finally give results as a percentage. This platform is fast, flexible, accurate and less ability to give errors. This generates the results as soon as answered by students and the result will be precise.

Our system decreases the plagiarism by shuffling questions and answers. We expect to encourage the students not to cheat when they are facing to the online quiz like this.

CLASS DEFINITIONS

In this system development program, there are classes as follower,

Class 01 → Login Class

Class 02 → User Class

Class 03 → Rule Class

Class 04 → Time Class

Class 05 → Quiz Class

Class 06 → Question Class

Class 07 → MultipleChoiceQuestion Class

Class 08 → TrueFalseQuestion Class

Class 09 → Result Class

Class 10 → Main Class

01) Login Class

There are no attributes in here.

The set of usernames and passwords of the users that have license to access stored in an array of User class objects. Then we use the constructor of the User class. The "***private final***" access modifiers indicate that the "users" array is only accessible within the class and cannot be modified outside the class.

```
// Hardcoded list of users for demonstration purposes
private final User[] users = {
        new User("kln2020", "abc123"),
        new User("ps2020", "def456")
};
```

As functionality ***authenticate()*** method used to identify users that request to access the system(like security method).

```
public User authenticate() {
    Scanner scanner = new Scanner(System.in);
    boolean isAuthenticated = false;
    User user = null;
    while (!isAuthenticated) {
        System.out.println("Please enter your username:");
        String username = scanner.nextLine();
        System.out.println("Please enter your password:");
        String password = scanner.nextLine();
        for (User u : users) {
            if (u.getUsername().equals(username) &&
                    u.getPassword().equals(password)) {
                user = u;
                isAuthenticated = true;
                break;
            }
        }
        if (!isAuthenticated) {
            System.out.println("Invalid username or password. Please
try again.");
        }
    }
    return user;
}
```

02) User Class

username & ***password*** are fields in this class. Both are final variables to avoid reuse and private to prevent access from outside.

```
private final String username;

private final String password;
```

A constructor takes two arguments: username and password, Which are passed as strings and used to initialize them.

```
 public User(String username, String password) {
    this.username = username;
    this.password = password;
}
```

As functions, ***getters*** used to read above data members.

```
public String getUsername() {
    return username;
}

public String getPassword() {
    return password;
}

public String getName() {
    return username;
}
```

03) Rule Class

Here data member ***"rules"*** is used to store rules which must followed by the applicant.

```
private final String rules = "Welcome to the quiz! Please read and
agree to the following rules:\n\n" +
        "1. You will have 1 minute to complete the quiz.\n" +
        "2. You may not use any external resources during the quiz.\n"
+
        "3. You may not collaborate with other individuals during the
quiz.\n" +
        "4. You may not attempt to cheat or plagiarize in any
way.\n\n" +"Do you agree to these rules? (yes/no)";
```

There are two functionalities. Those are ***showRules()*** and ***agreeToRules()***. First one used for showing rules to the user. And other one for accepting user to agree to the rules provided by system.

```
public void showRules() {
    System.out.println(rules);
}
public boolean agreeToRules(User user) {
    Scanner scanner = new Scanner(System.in);
    String response = scanner.nextLine();
    if (response.equalsIgnoreCase("yes")) {
        return true;
    } else {
        return false;
    }
}
```

04) Time Class

duration and ***timeRemainning*** are the attributes of this class.

```
private int timeRemaining,duration;
```

startTime(), takeDuration() and ***TimeTaken()*** are the methods of this class. In startTime() method we calculate the time and when the time is up, quiz is end. We can take the time duration of the quiz by takeDuration() method. And finally, we can calculate and display the time taken by TimeTaken() method.

```
public int takeDuration(){
    Quiz object =new Quiz("Java Quiz",60);
    duration=object.getDuration();
    return duration;
}
public void startTime(){
    timeRemaining =takeDuration() ;
    // Set up a timer to track the time remaining for the quiz
    java.util.Timer timer = new java.util.Timer();
    timer.schedule(new TimerTask() {
        public void run() {
            timeRemaining--;
            if (timeRemaining == 0) {
                timer.cancel();
                System.out.println("Time's up! Quiz submitted.");
            }
        }
    }, 0, 1000); // Update the timer every second
}
public void TimeTaken(){
    System.out.println("Time Taken : "+(duration-timeRemaining)+"
seconds");

}
```

05) Quiz Class

There are several attributes such as ***name, duration, question, choices*** and ***answer.***

```
private  String name;
private  int duration;
private String question;
private ArrayList<String> choices;
private String answer;
```

take() method used to take and welcome the user to the quiz system. As well, ***getters*** use to read data members that mentioned above. Avoid cheating as the main purpose of our system we used ***Quiz constructor*** to shuffle questions and answers.

```
public void take(User user) {
    System.out.println("\n//--------------//");
    System.out.print("Welcome ");
    System.out.println(user.getName()+"!");
    System.out.print("You have ");
    Time x=new Time();
    System.out.print(x.takeDuration());
    System.out.println(" seconds to complete the quiz.");
    System.out.println("Starting quiz: " +name);
    System.out.println("//-------------//\n");
}
```

```
public String getQuestion() {
    return question;
}

public ArrayList<String> getChoices() {
    return choices;
}

public String getAnswer() {
    return answer;
}

public int getDuration() {
    return duration;
}
```

```
public Quiz(String question,String[] choices,String answer){
    this.question=question;
    this.choices=new ArrayList<String>();
    for (int i=0;i<choices.length;i++){
        this.choices.add(choices[i]);
    }
    //make sure each game shows questions imn different order
    Collections.shuffle(this.choices);
    this.answer=answer;
}
```

06) Question Class

questionSet and ***numCorrect*** are fields in this class.

```
public ArrayList<Quiz> questionSet;
int numCorrect=0;
```

Start() method is used to check the answers, calculate the number of correct answers. In additional, we have used *Exception Handling* for invalid answers given by applicant.

```
public void start() {
    Scanner scan = new Scanner(System.in);
    //show questions from question set
    for (int question = 0; question < questionSet.size(); question++) {
        System.out.println(questionSet.get(question).getQuestion());
        int numChoices = questionSet.get(question).getChoices().size();
        //show choices from questions in question
        for (int choice = 0; choice < numChoices; choice++) {
            System.out.println((choice + 1) + ":" +
questionSet.get(question).getChoices().get(choice));
        }
        // get player answer and validate input
        int playerAnswer = 0;
        boolean validInput = false;
        while (!validInput) {
            try {
                playerAnswer = scan.nextInt();
                if (playerAnswer < 1 || playerAnswer > numChoices) {
                    System.out.println("Invalid input. Please enter a number
between 1 and " + numChoices + ".");
                } else {
                    validInput = true;
                }
            } catch (InputMismatchException e) {
                System.out.println("Invalid input. Please enter a number.");
                scan.next(); // consume the invalid input
            }
        }
        ArrayList<String> choiceSet = questionSet.get(question).getChoices();
        String correctAnswer = questionSet.get(question).getAnswer();
        int correctAnswerIndex = choiceSet.indexOf(correctAnswer);
        if (playerAnswer == correctAnswerIndex + 1) {
            numCorrect++;
        }
    }
}
```

07) MultipleChoiceQuestion Class

There are no attributes in here. But this class can inherit all the attributes as well as methods of parent class (Question class).

In a constructor, question set is made. Furthermore, *ArrayList* is used to store questions and answers.

Answers which given by the user are checked in ***checkAnswers()*** method. We can show the results from ***toString()*** method.

```
public MultipleChoiceQuestion(){
  //  super();
    questionSet=new ArrayList<Quiz>();
    String q="_____ is a Keyword";
    String[] a={"Final","String","Finally","Finalize"};
    questionSet.add(new Quiz(q,a,"Final"));
    q="Which of the following is not a Java Feature?";
    //must reuse variable with new a string[]
    a=new String[]{"Dynamic","Architecture Neutral","Use of
pointers","Object-oriented"};
    questionSet.add(new Quiz(q,a,"Use of pointers"));
    q="In which memory a String is stored, when we create a string
using new operator?";
    //must reuse variable with new a string[]
    a=new String[]{"Stack","String Memory","Heap memory","Random
storage space"};
    questionSet.add(new Quiz(q,a,"Heap memory"));
    Collections.shuffle(questionSet,new Random());

}
public void checkAnswers(){
    super.start();
}
```

```
public String toString() {
    return "Your  Results for MultipleChoice Questions\n----------
----------\nYou have got "+super.numCorrect+" correct Answers
"+"\n"+"You have got "+(questionSet.size()-super.numCorrect)+"
Incorrect answers\nyour grade is
"+(((double)super.numCorrect/questionSet.size())*100)+"%\n----------
----------";
}
```

08) TrueFalseQuestion class

This class also behaves as above MultipleChoiceQueston Class.

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Random;

public class TrueFalseQuestion extends Question{


    public TrueFalseQuestion(){
        //super();

        questionSet=new ArrayList<Quiz>();
        String q="Constructor overloading is not possible in java.";
        String[] a={"True","False"};
        questionSet.add(new Quiz(q,a,"False"));
        q="Assignment operator is evaluated Left toRight.";
        //must reuse variable with new a string[]
        a=new String[]{"True","False"};
        questionSet.add(new Quiz(q,a,"False"));
        q="In an instance method or a constructor,\"this\"is a
reference to the current object";
        //must reuse variable with new a string[]
        a=new String[]{"True","False"};
        questionSet.add(new Quiz(q,a,"True"));
        Collections.shuffle(questionSet,new Random());

    }
    public void checkAnswers() {
        super.start();
    }
    @Override
    public String toString() {
        return "Your Results for True False Questions\n--------------
------\nYou have got "+numCorrect+" correct Answers "+"\n"+"You have
got "+(questionSet.size()-numCorrect)+" Incorrect answers\nyour grade
is "+(((double)numCorrect/questionSet.size())*100)+"%";
    }
}
```

09) Result Class

No data members here.

A constructor is used to make objects of MultipleChoiceQuestion class and TrueFalseQuestion classes and print them. Then calculated results are displayed in toString() of those classes. The final mark is calculated using those objects.

```
public Result(){
    MultipleChoiceQuestion g=new MultipleChoiceQuestion();
    TrueFalseQuestion t=new TrueFalseQuestion();
    g.checkAnswers();
    t.checkAnswers();
    System.out.println(g);
    System.out.println(t);
    double final_mark= ((((double) g.numCorrect/g.questionSet.size())+
((double) t.numCorrect/t.questionSet.size()))/2)*100;
    System.out.println("----------------------\nYour Final average
mark is "+final_mark+"%");
}
```

10) Main class

There are no attributes here also.

As the functionalities we have created the following objects and called to the relevant constructors and methods respectively.

First, we create the **“Login object”** then create the **“user constructor”** & call to **authenticate() method** using this login object. In here, check whether the user is null or not by conditional statement. If null or user entered invalid username, password then print the message to force

the user to enter valid information. If not, display the rules to the user & check whether the agree or not for them. If user enter “no”, print the message & exit from the system.

Otherwise, user enter “yes”, call to the “**Quiz constructor”** and create **“Time object**”. Then, call to **startTime()** & **take()** methods respectively. After user answering the questions call to “**Result constructor**” & call to “**TimeTaken()”** method.

```
public class Main {
    public static void main(String[] args) {
        // Authenticate the user
        Login login = new Login();
        User user = login.authenticate();
        if (user != null) {
            // Show the quiz rules and conditions
            Rule rule = new Rule();
            rule.showRules();
            // Check if the user agrees to the rules
            if (rule.agreeToRules(user)) {
                Quiz quiz = new Quiz("Java Quiz", 60);
                Time disptime = new Time();
                //start the time
                disptime.startTime();
                //Take the quiz
                quiz.take(user);
                //Display the result
                Result obj=new Result();
                // Display time taken
                disptime.TimeTaken();
            } else {
                // User did not agree to the rules
                System.out.println("You must agree to the rules
to take the quiz.");
            }
        } else {
            // Authentication failed
            System.out.println("Invalid username or password.");
        }
    }
}
```

OOP CONCEPTS

Creating objects and classes

- There are Classes to the program which relevant to each section.
- For all those classes there are specific objects and some attributes and behaviors are implemented to those objects.
- As the variables and methods that some classes have, can access via these objects.
- Main method in another class
- To control the access for each class, method and, variable has been declared with the specific access modifiers.

Constructors

- To instantiate objects, classes have used constructors with particular parameters.

OOP Principles

❖ Inheritance

There are subclasses extended from each parent classes.
Question class extends MultipleChoiceQuestion class and TrueFalseQuestion class. MultipleChoiceQuestion ***is a*** Question and TrueFalseQuestion ***is a*** Question. Therefore, there is a hierarchical Inheritance between child and parent classes.

```
public class MultipleChoiceQuestion extends Question {
```

```
public class TrueFalseQuestion extends Question{
```

```
public void checkAnswers(){
    super.start();
}
```

❖ Polymorphism

➢ Method Overriding

This is applied by using toString() method to override the object which creates by java compiler when we try to print an object of the class. This is used to get understandable information which is string representation of the object.

In TrueFalseQuestion class,

```
@Override
public String toString() {
    return "Your Results for True False Questions\n----------------------\nYou
have got "+numCorrect+" correct Answers "+"\n"+"You have got
"+(questionSet.size()-numCorrect)+" Incorrect answers\nyour grade is
"+(((double)numCorrect/questionSet.size())*100)+"%";
}
```

In MultipleChoiceQuestion class,

```
public String toString() {
    return "Your  Results for MultipleChoice Questions\n--------------------
---\nYou have got "+super.numCorrect+" correct Answers "+"\n"+"You have got
"+(questionSet.size()-super.numCorrect)+" Incorrect answers\nyour grade is
"+(((double)super.numCorrect/questionSet.size())*100)+"%\n--------------------
---";
}
```

➢ Method Overloading

```
public Quiz(String name, int duration){
    this.name=name;
    this.duration=duration;
}
public Quiz(String question,String[] choices,String answer){
    this.question=question;
    this.choices=new ArrayList<String>();
    for (int i=0;i<choices.length;i++){
        this.choices.add(choices[i]);
    }
    //make sure each game shows questions imn different order
    Collections.shuffle(this.choices);
    this.answer=answer;
}
```

we have two constructors with different parameter lists. First constructor is use to initialize user information and second is to initialize data members related to questions (question, choices and answers)

❖ Encapsulation

In User class, getters are used to access to read private data members.

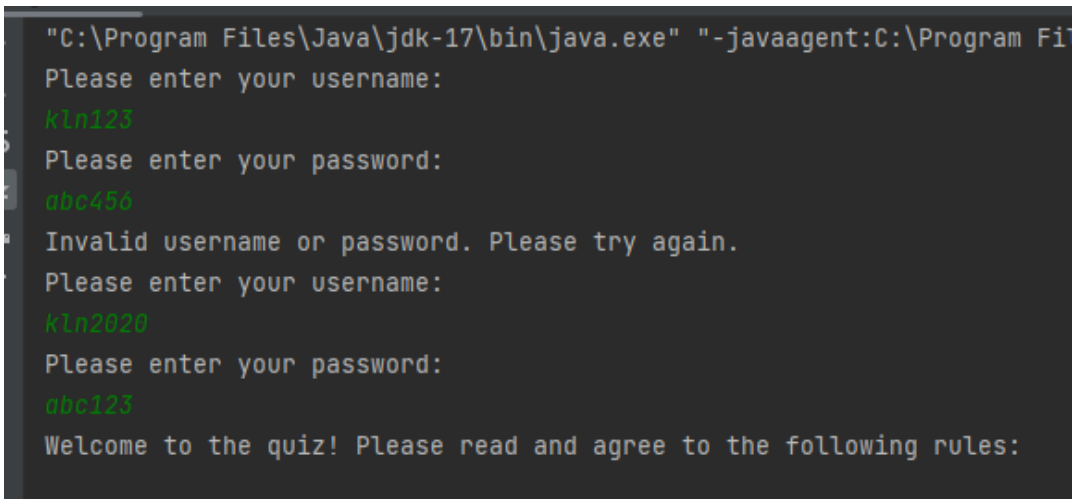
```
public String getUsername() {
    return username;
}

public String getPassword() {
    return password;
}

public String getName() {
    return username;
}
```

In Quiz class getters are used to read private data members such as question, choices, answer and duration.

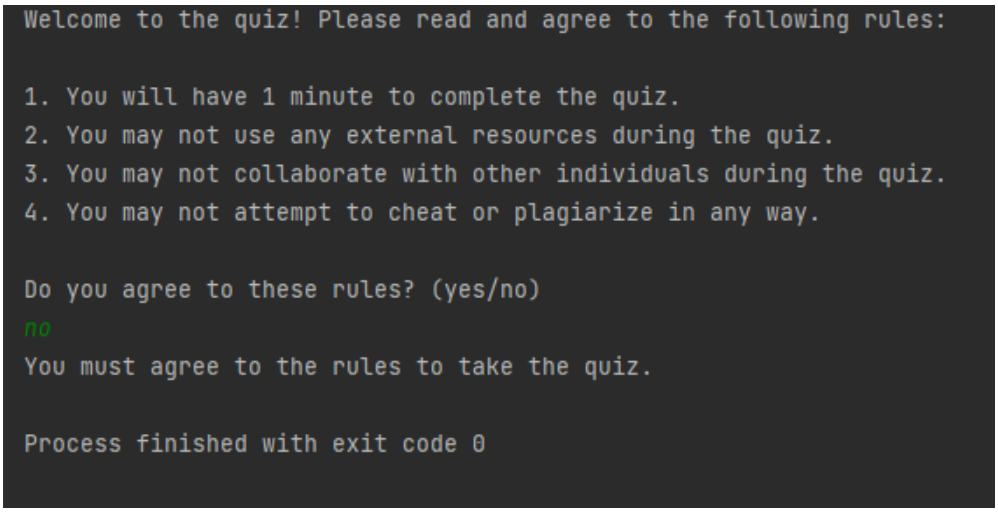
```
public String getQuestion() {
    return question;
}

public ArrayList<String> getChoices() {
    return choices;
}

public String getAnswer() {
    return answer;
}

public int getDuration() {
    return duration;
}
```

SAMPLE PROCESS OF THE PROGRAM

First User have to log in to the system. So, user should enter invalid information.

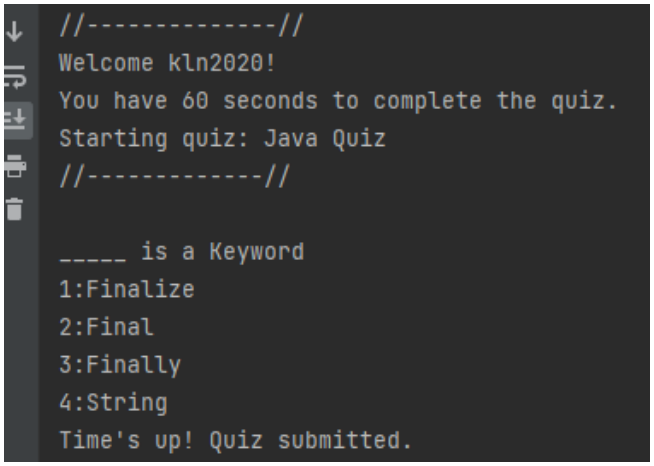


After login to the system, System displays the rules which should agree by the user. User have to agree to those if not system gives a warning and exit from it. That's how we provide very tough exams using our online quiz management system.



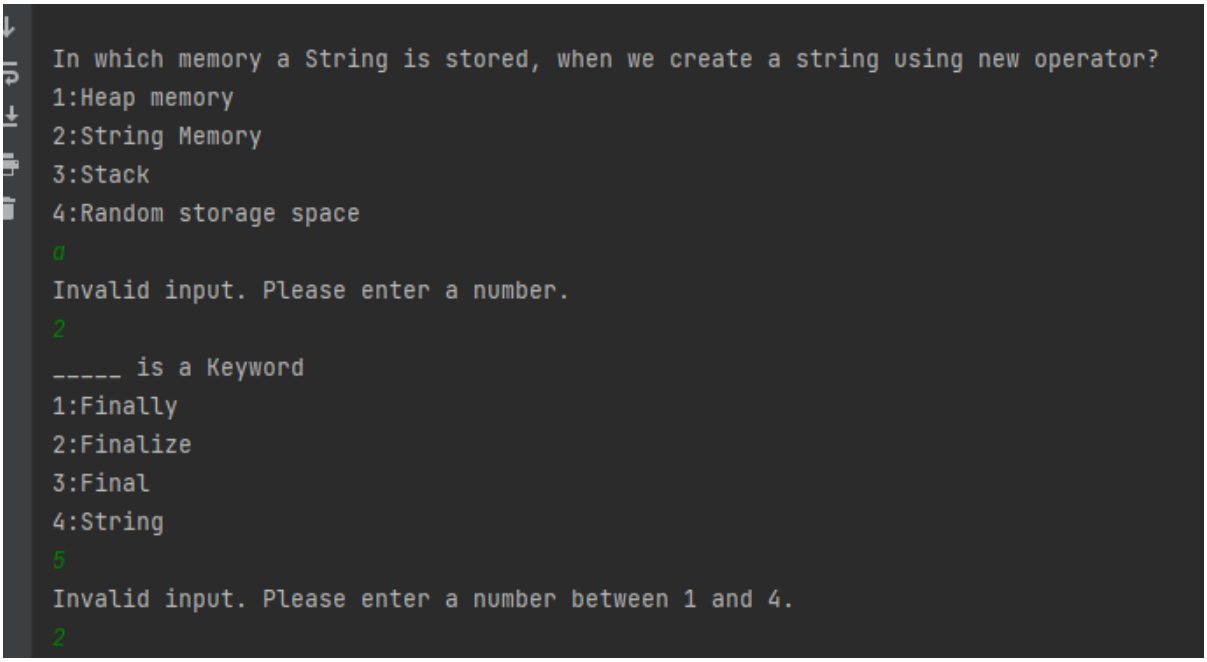
If user agree to the rules, there can be seen a welcome note. Then he can do the questionnaire. User can read questions carefully and answer them.

When answering user must enter valid answers if not the system incites to enter the valid answers.



If applicant is answering multiple choice questions, user must enter 1 or 2 or 3 or 4. If applicant answering true false questions user must enter 1 or 2.

If user is able to answer all the questions within the time, he can get results immediately. If not user unable to get it and display a note and quiz ends.



```
Which of the following is not a Java Feature?
1:Architecture Neutral
2:Use of pointers
3:Object-oriented
4:Dynamic
4
In an instance method or a constructor,"this"is a reference to the current object
1:True
2:False
1
Constructor overloading is not possible in java.
1:False
2:True
1
Assignment operator is evaluated Left toRight.
1:True
2:False
2
```

The result will calculate for multiple choice questions and true false questions separately. Then the total mark for both type of questions is calculated and also time taken for answering is displayed.

```
Your  Results for MultipleChoice Questions
------------------------
You have got 0 correct Answers
You have got 3 Incorrect answers
your grade is 0.0%
----------------------
Your Results for True False Questions
--------------------
You have got 3 correct Answers
You have got 0 Incorrect answers
your grade is 100.0%
---------------------
Your Final average mark is 50.0%
Time Taken : 25 seconds
Time's up! Quiz submitted.

Process finished with exit code 0
```

MODIFICATION FOR THE INITIAL PLAN

We have to add 6 new classes such as **User, Time, MultipleChoiceQuestion, TrueFalseQuestion** and **Main.** In addition, we remove 2 classes namely *PlagiarismChecking* and *AnswerChecking* from initial plan.

The functionalities of removed classes are done within the program in **Result class** and **each child class of the Question class**. As well, we add the concepts of removed classes to the new classes.

Furthermore, we add more OOP concepts. **(Inheritance, Encapsulation and Polymorphism)**

Additionally, we used **Exception Handling (try – catch) block** in **Question** class to print message for user entered invalid answers.

```
// get player answer and validate input
int playerAnswer = 0;
boolean validInput = false;
while (!validInput) {
    try {
        playerAnswer = scan.nextInt();
        if (playerAnswer < 1 || playerAnswer > numChoices) {
            System.out.println("Invalid input. Please enter a number between
1 and " + numChoices + ".");
        } else {
            validInput = true;
        }
    } catch (InputMismatchException e) {
        System.out.println("Invalid input. Please enter a number.");
        scan.next(); // consume the invalid input
    }
}
```

Reasons for above modifications

We except to do some additional methods (such as display time countdown, record the video, avoid getting screenshots and share scree). But we couldn't do it from java since it's very complex. As well, we had not clear idea about the program at the beginning.

CHALLENGES AND SOLUTIONS

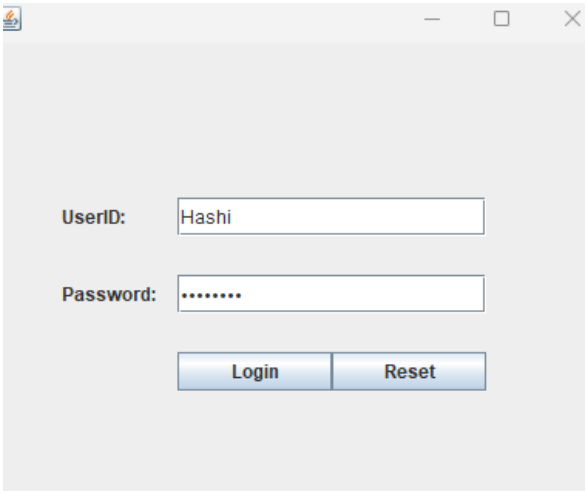
Within the time of working with project we have to face some challenges.

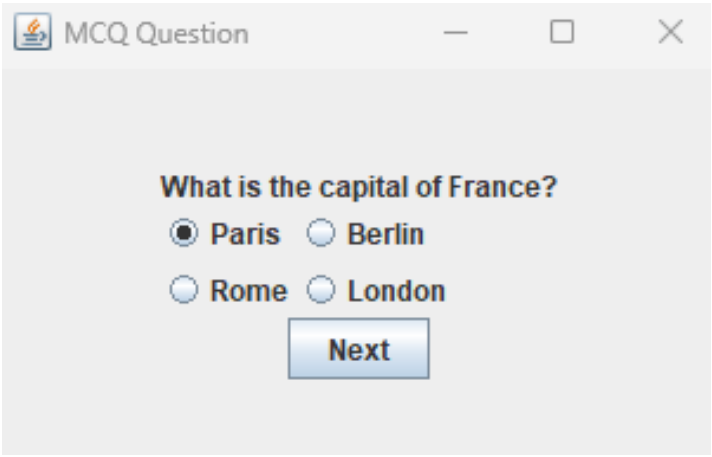
At the beginning we have no data base to log large number of users to give data to the system to check the validity when login. We used ***arrays*** to store several data members of users as the solution for this.

```
private final User[] users = {
        new User("kln2020", "abc123"),
        new User("ps2020", "def456")
};
```

Furthermore, we couldn't calculate the result & time taken if the user's answers are null.

Then we try to give user friendly interface (**GUI**) for this system (As below). But it was very hard.





So, we run the program in *IntelliJ IDEA*.

TEAMWORK

Task of each member

- PS/2020/290 → Introduction, Class Definitions, Challenges and Solutions
- PS/2020/269 → OOP concepts & Sample process of the program
- PS/2020/027 → Modifications for the initial plan & gather information for the report
- PS/2020/252 → Copy the codes & find the pictures, screenshots
- PS/2020/209 → Reedition & modify the report

APPENDIX(SOURCE CODE)

Our Source Code is uploaded to a google drive and anyone on the internet with the link can view through following link.

https://drive.google.com/drive/folders/17-MwALmGURXjW_Jnhx3HHJJi09mCZSBN?usp=sharing

Anyone can refer this code of our program and contribute to the project if you wish.

……//////-----THE END-----//////……