**Assignment 1**

Name: null  
Class: null  
Roll No.: null

**Q.1 write a program to implement something:**

**Program:**

import edu.duke.DirectoryResource;  
import org.apache.poi.xwpf.usermodel.\*;  
  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.Scanner;  
  
public class LazyAssignment {  
 private XWPFDocument document;  
 private User user;  
 int noOfQuestions;  
 int assignmentNo;  
  
 // constructor for Lazy people @rushan  
 public LazyAssignment(XWPFDocument document, User user) {  
 this.document = document;  
 this.user = user;  
 initializeAssignment(this.document, this.user);  
  
 }  
  
 // initialize the document with user details @rushan / @Saubaan  
 private void initializeAssignment(XWPFDocument document, User user) {  
 Scanner scan = new Scanner(System.in);  
  
 System.out.print("Enter Assignment no.: ");  
 this.assignmentNo = scan.nextInt();  
  
 System.out.print("Enter no. of questions : ");  
 this.noOfQuestions = scan.nextInt();  
  
 try {  
 FileOutputStream outputStream = new FileOutputStream("Assignment-" + this.assignmentNo + ".docx");  
 XWPFParagraph title = document.createParagraph();  
 title.setAlignment(ParagraphAlignment.CENTER);  
 System.out.println("File Created");  
 XWPFRun titleRun = title.createRun();  
 titleRun.setText("Assignment " + this.assignmentNo);  
 titleRun.setFontSize(25);  
 titleRun.setCapitalized(true);  
 titleRun.setBold(true);  
 addSpace(titleRun);  
  
 // @Saubaan/@rushan initialize user info  
 XWPFParagraph userDetails = document.createParagraph();  
 XWPFRun userDetailsRun = userDetails.createRun();  
 userDetailsRun.setFontSize(14);  
 userDetailsRun.setText("Name: " + user.getUserName());  
 userDetailsRun.addBreak();  
 userDetailsRun.setText("Class: " + user.getUserClass());  
 userDetailsRun.addBreak();  
 userDetailsRun.setText("Roll No.: " + user.getUserRoll());  
 userDetailsRun.addBreak();  
 addSpace(userDetailsRun);  
  
 document.write(outputStream);  
// document.close();  
 } catch (IOException e) {  
 System.err.println("Document Error: " + e.getMessage());  
 }  
  
 }  
  
 public void generateAssignmentBody(XWPFDocument document) {  
  
  
 // @Saubaan Get question text from User  
 ArrayList<String> questions = getQuestions();  
  
  
 // @rushan getting file path of selected files  
 ArrayList<String> selectedFiles = getJavaFilePaths();  
  
 // @Saubaan getting image files path  
// ArrayList<String> selectedFiles = getJavaFilePaths();  
  
 try {  
 FileOutputStream outputStream = new FileOutputStream("Assignment-" + this.assignmentNo + ".docx");  
 // @Saubaan create body for questions  
 XWPFParagraph questionTextPara;  
 XWPFRun questionTextRun;  
 for (int i = 0; i < selectedFiles.size(); i++) {  
  
 // @Saubaan Create paragraph for question  
 questionTextPara = document.createParagraph();  
 questionTextPara.setAlignment(ParagraphAlignment.LEFT);  
 questionTextRun = questionTextPara.createRun();  
 questionTextRun.setBold(true);  
 questionTextRun.setFontSize(13);  
 questionTextRun.setText("Q." + (i+1) + " " + questions.get(i));  
 questionTextRun.addBreak();  
  
 // @Saubaan Create title for "Program: " text  
 XWPFParagraph programTitlePara = document.createParagraph();  
 programTitlePara.setAlignment(ParagraphAlignment.LEFT);  
 XWPFRun programTitleRun = programTitlePara.createRun();  
 programTitleRun.setBold(true);  
 programTitleRun.setText("Program: ");  
 programTitleRun.addBreak();  
  
 // @Saubaan Create paragraph for the program text  
 XWPFParagraph programTextPara = document.createParagraph();  
 programTextPara.setAlignment(ParagraphAlignment.LEFT);  
 XWPFRun programTextRun = programTextPara.createRun();  
 writeFileToDocument(programTextRun, selectedFiles.get(i));  
  
 // @rushan create output segment  
 XWPFParagraph outputTitlePara = document.createParagraph();  
 programTitlePara.setAlignment(ParagraphAlignment.LEFT);  
 XWPFRun outputTitleRun = outputTitlePara.createRun();  
 outputTitleRun.setBold(true);  
 outputTitleRun.setText("Output: ");  
 outputTitleRun.addBreak();  
  
 // @Saubaan Create a new page for each question  
 document.createParagraph().createRun().addBreak(BreakType.PAGE);  
 }  
  
 document.write(outputStream);  
 } catch (IOException e) {  
 System.err.println("Document Error: " + e.getMessage());  
 }  
 }  
  
 // writing the selected files to word document @rushan  
 private void writeFileToDocument(XWPFRun run, String filePath) {  
 try {  
 String line;  
 BufferedReader reader = new BufferedReader(new FileReader(filePath));  
 while ((line = reader.readLine()) != null) {  
 run.setText(line);  
 run.addBreak();  
 }  
 } catch (IOException e) {  
 System.err.println("An Error occurred: " + e.getMessage());  
 }  
 }  
  
 // get multiple file paths to manipulate as per our requirement @rushan  
 private ArrayList<String> getJavaFilePaths() {  
 ArrayList<String> selectedFilePaths = new ArrayList<String>();  
 DirectoryResource dr = new DirectoryResource();  
 for (File f : dr.selectedFiles()) {  
 selectedFilePaths.add(f.getAbsolutePath());  
 }  
  
 return selectedFilePaths;  
 }  
  
 // @rushan add space between segments or blocks  
 private void addSpace(XWPFRun run) {  
 for (int i = 0; i < 2; i++) {  
 run.addBreak();  
 }  
 }  
  
 // @Saubaan get questions from user using scanner  
 public ArrayList<String> getQuestions(){  
 ArrayList<String> questions = new ArrayList<>();  
 Scanner scan = new Scanner(System.in);  
 for (int i = 0; i < this.noOfQuestions; i++) {  
 System.out.printf("Enter question no %d: ", i+1);  
 String question = scan.nextLine();  
 questions.add(question);  
 }  
 return questions;  
 }  
  
 // @rushan main initialized  
 public static void main(String[] args) {  
 XWPFDocument document = new XWPFDocument();  
 User user = new User();  
 LazyAssignment main = new LazyAssignment(document, user);  
 main.generateAssignmentBody(document);  
 System.out.println("Operation completed successfully! ");  
 }  
}

**Output:**

**Q.2 write another program to implement something:**

**Program:**

// User class @rushan  
  
import java.io.\*;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.util.Scanner;  
  
public class User {  
  
 private String[] information = new String[3];  
  
 public User() {  
 File file = new File(".user\_config");  
  
 try {  
 Path filePath = Paths.get(".user\_config");  
 Files.setAttribute(filePath, "dos:hidden", true);  
 } catch(IOException e){  
 System.out.println("let's create a new one. ");  
 }  
  
 try {  
 if (!fileLengthZero(file)) {  
 BufferedReader reader = new BufferedReader(new FileReader(file));  
 information = reader.readLine().split(",");  
 reader.close();  
 } else {  
 BufferedWriter writer = new BufferedWriter(new FileWriter(file));  
 inputUserDetails(writer);  
 try {  
 Path filePath = Paths.get(".user\_config");  
 Files.setAttribute(filePath, "dos:hidden", true);  
 } catch(IOException e){  
 System.out.println("You have nothing in user\_config");  
 }  
 }  
 } catch (IOException ioException) {  
 System.out.println("Error: You have nothing in user\_config");  
 file.delete();  
 new User();  
 }  
 }  
 public String getUserName() { return information[0]; }  
 public String getUserRoll() { return information[1]; }  
 public String getUserClass() { return information[2]; }  
  
 public void setUserName(String value) { information[0] = value; }  
 public void setUserRoll(String value) { information[1] = value; }  
 public void setUserClass(String value) { information[2] = value; }  
  
 // method to print details of the user @rushan  
 public void getUserDetails() {  
 for (String value : information) {  
 System.out.println(value);  
 }  
 }  
  
 private boolean fileLengthZero(File file) {  
 return file.length() == 0;  
 }  
  
 private void inputUserDetails(BufferedWriter writer) {  
 String[] inputInformation = new String[information.length];  
 String[] questions = {"Name : ", "Roll No. : ", "Class : "};  
 Scanner scan = new Scanner(System.in);  
 for (int i = 0; i < information.length; i++) {  
 System.out.print(questions[i]);  
 inputInformation[i] = scan.nextLine();  
 }  
  
 String addThisLineToFile = String.join(",", inputInformation);  
 try {  
 writer.write(addThisLineToFile);  
 writer.close();  
 } catch (IOException e) {  
 System.out.println("Error: You successfully managed to delete user\_info.txt ");  
 }  
 }  
  
 public void updateUserDetails(File file) {  
 if (fileLengthZero(file)) {  
 try {  
 BufferedWriter writer = new BufferedWriter(new FileWriter(file));  
 inputUserDetails(writer);  
  
 } catch (IOException e) {  
  
 }  
 } else {  
 Scanner scan = new Scanner(System.in);  
 while (true) {  
 System.out.println("1.Name 2.Roll 3.Class 4.Save and Exit");  
 System.out.print("Enter the value to update (1, 2, 3, 4) ; ");  
 String choice = scan.next();  
 switch (choice) {  
 case "1":  
 setUserName(scan.nextLine());  
 break;  
 case "2":  
 setUserRoll(scan.nextLine());  
 break;  
 case "3":  
 setUserClass(scan.nextLine());  
 break;  
 case "4":  
 writeUpdatesToFile(information);  
 System.exit(1);  
 default:  
 System.out.println("Invalid choice");  
 }  
 }  
 }  
 }  
  
 private void writeUpdatesToFile(String[] changedInformation) {  
 String addThisLineToFile = String.join(",", changedInformation);  
 try {  
 BufferedWriter writer = new BufferedWriter(new FileWriter(".input.txt"));  
 writer.write(addThisLineToFile);  
 writer.close();  
 } catch (IOException e) {  
  
 }  
 }  
  
}

**Output:**