

PART A.

```
/*
```

```
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this  
license
```

```
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
```

```
*/
```

```
/**
```

```
 *
```

```
 * @author user
```

```
 */
```

```
import java.util.Scanner;
```

```
public class StudentGrader {
```

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

```
        int[] gradeCounts;
```

```
        try (Scanner scanner = new Scanner(System.in)) {
```

```
            int numStudents = 5;
```

```
            int studentCount = 0;
```

```
            gradeCounts = new int[10];
```

```
            System.out.println("Automated Grading System");
```

```
            while (studentCount < numStudents) {
```

```
                System.out.print("Enter score for student " + (studentCount + 1) + " (out of 100): ");
```

```
                int score = scanner.nextInt();
```

```
                if (score < 0 || score > 100) {
```

```
                    System.out.println("Invalid score. Please enter a score between 0 and 100.");
```

```
                    continue;
```

```
                }
```

```
int grade = 0;
String remark = "";

if (score >= 80 && score <= 100) {
    grade = 1;
    remark = "D1";
} else if (score >= 75 && score <= 79) {
    grade = 2;
    remark = "D2";
} else if (score >= 66 && score <= 74) {
    grade = 3;
    remark = "C3";
} else if (score >= 60 && score <= 65) {
    grade = 4;
    remark = "C4";
} else if (score >= 50 && score <= 59) {
    grade = 5;
    remark = "C5";
} else if (score >= 45 && score <= 49) {
    grade = 6;
    remark = "C6";
} else if (score >= 35 && score <= 44) {
    grade = 7;
    remark = "P7";
} else if (score >= 30 && score <= 34) {
    grade = 8;
    remark = "P8";
} else {
    grade = 9;
```

```
        remark = "F";  
    }
```

```
    System.out.println("Score: " + score + ", Grade: " + grade + ", Remark: " + remark);
```

```
    if (grade >= 1 && grade <= 9) {  
        gradeCounts[grade]++;  
    }
```

```
        studentCount++;  
    }  
}
```

```
System.out.println("\n--- Grade Summary ---");  
for (int i = 1; i <= 9; i++) {  
    System.out.println("Grade " + i + ": " + gradeCounts[i] + " student(s)");  
}  
}  
}
```

PART B

```
/*
```

```
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this  
 license
```

```
*/
```

```
package com.mycompany.vehicleauctionsystem2;
```

```
/**
```

```

*
* @author user
*/
import java.util.Scanner;

public class VehicleAuctionSystem2 {

    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            int numBidders = 3;

            Bidder highestBidder = null;

            System.out.println("--- Bidder Registration ---");
            for (int i = 1; i <= numBidders; i++) {
                System.out.print("Enter name for bidder " + i + ": ");
                String name = scanner.nextLine();
                System.out.print("Enter bid price for bidder " + i + ": $");
                double bidPrice = scanner.nextDouble();
                scanner.nextLine(); // Consume newline

                Bidder currentBidder = new Bidder();
                currentBidder.name = name;
                currentBidder.bidPrice = bidPrice;

                if (highestBidder == null || currentBidder.bidPrice > highestBidder.bidPrice) {
                    highestBidder = currentBidder;
                }
            }
        }
    }
}

```

```
    if (highestBidder != null) {  
        System.out.println("\nAuction complete. The highest bidder is " + highestBidder.name + "  
with a bid of $" + highestBidder.bidPrice);  
    } else {  
        System.out.println("\nNo bidders were processed.");  
        return;  
    }  
}
```

```
System.out.println("\n--- Vehicle Details & Calculation ---");  
Vehicle vehicle = new Vehicle();  
vehicle.soldPrice = highestBidder.bidPrice;
```

```
System.out.print("Enter vehicle registration number: ");  
vehicle.registrationNumber = scanner.nextLine();
```

```
System.out.print("Enter original vehicle cost: $");  
vehicle.cost = scanner.nextDouble();
```

```
System.out.print("Enter total expenses incurred on vehicle: $");  
vehicle.expenses = scanner.nextDouble();
```

```
System.out.print("Enter total deposits made so far: $");  
vehicle.depositsMade = scanner.nextDouble();
```

```
System.out.print("Enter remaining balance left on vehicle: $");  
vehicle.balanceLeft = scanner.nextDouble();  
scanner.nextLine();
```

```
System.out.println("\n*** Vehicle Summary ***");
System.out.println("Registration Number: " + vehicle.registrationNumber);
System.out.println("Original Cost: $" + String.format("%.2f", vehicle.cost));
System.out.println("Total Expenses: $" + String.format("%.2f", vehicle.expenses));
System.out.println("Total Deposits Made: $" + String.format("%.2f", vehicle.depositsMade));
System.out.println("Remaining Balance: $" + String.format("%.2f", vehicle.balanceLeft));
System.out.println("Final Sale Price (Highest Bid): $" + String.format("%.2f", vehicle.soldPrice));
```

```
System.out.print("\nHas the bidder cleared the balance? (yes/no): ");
String balanceClearedInput = scanner.nextLine();
```

```
if (balanceClearedInput.equalsIgnoreCase("yes")) {
    System.out.println("\nCalculating final profit/loss...");
```

```
    double totalCost = vehicle.cost + vehicle.expenses;
    double profitLoss = vehicle.soldPrice - totalCost;
```

```
    System.out.println("Total Cost Incurred: $" + String.format("%.2f", totalCost));
    System.out.println("Total Revenue from Sale: $" + String.format("%.2f", vehicle.soldPrice));
```

```
    if (profitLoss >= 0) {
        System.out.println("Result: **Profit of $" + String.format("%.2f", profitLoss) + " **");
    } else {
        System.out.println("Result: **Loss of $" + String.format("%.2f", Math.abs(profitLoss)) +
" **");
    }
} else {
    System.out.println("Cannot compute final profit/loss until balance is cleared.");
```

```
    }  
    }  
}  
}
```

QUESTION 2

```
/*  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this  
license  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  
 */  
package studentsregistrationapp3;  
  
/**  
 *  
 * @author user  
 */  
import javax.swing.*;  
import java.awt.*;  
import java.io.*;  
import java.time.LocalDate;  
import java.time.Period;  
  
public class StudentsRegistrationApp extends JFrame {  
  
    private final JTextField txtFirst;  
    private final JTextField txtLast;  
    private final JTextField txtEmail;  
    private final JTextField txtConfirmEmail;  
    private final JPasswordField txtPass;  
    private final JPasswordField txtConfirmPass;
```

```
private final JComboBox<Integer> cbYear;
private final JComboBox<Integer> cbDay;
private final JComboBox<String> cbMonth;
private final JRadioButton rbMale;
private final JRadioButton rbFemale;
private final JComboBox<String> cbDept;
private final JTextArea summaryArea;
private static int studentCounter = 1;

public StudentsRegistrationApp() {
    setTitle("New Student Registration");
    setSize(900, 500);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLocationRelativeTo(null);
    setLayout(new BorderLayout(10, 10));

    JPanel formPanel = new JPanel(new GridBagLayout());
    GridBagConstraints gbc = new GridBagConstraints();
    gbc.insets = new Insets(5, 10, 5, 10);
    gbc.fill = GridBagConstraints.HORIZONTAL;

    addLabelAndField(formPanel, "First Name:", txtFirst = new JTextField(15), gbc, 0);
    addLabelAndField(formPanel, "Last Name:", txtLast = new JTextField(15), gbc, 1);
    addLabelAndField(formPanel, "Email:", txtEmail = new JTextField(15), gbc, 2);
    addLabelAndField(formPanel, "Confirm Email:", txtConfirmEmail = new JTextField(15), gbc, 3);
    addLabelAndField(formPanel, "Password:", txtPass = new JPasswordField(15), gbc, 4);
    addLabelAndField(formPanel, "Confirm Password:", txtConfirmPass = new JPasswordField(15),
gbc, 5);
```

```

gbc.gridy = 6; gbc.gridx = 0; formPanel.add(new JLabel("DOB:"), gbc);
JPanel dobPanel = new JPanel(new FlowLayout(FlowLayout.LEFT, 5, 0));
cbYear = new JComboBox<>();
for (int i = 2026; i >= 1960; i--) cbYear.addItem(i);
cbMonth = new JComboBox<>(new String[]{"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug",
"Sep", "Oct", "Nov", "Dec"});
cbDay = new JComboBox<>();

cbYear.addActionListener(e -> updateDays());
cbMonth.addActionListener(e -> updateDays());
updateDays();

dobPanel.add(cbYear); dobPanel.add(cbMonth); dobPanel.add(cbDay);
gbc.gridx = 1; formPanel.add(dobPanel, gbc);


gbc.gridy = 7; gbc.gridx = 0; formPanel.add(new JLabel("Gender:"), gbc);
rbMale = new JRadioButton("Male"); rbFemale = new JRadioButton("Female");
ButtonGroup bg = new ButtonGroup(); bg.add(rbMale); bg.add(rbFemale);
JPanel genPanel = new JPanel(new FlowLayout(FlowLayout.LEFT));
genPanel.add(rbMale); genPanel.add(rbFemale);
gbc.gridx = 1; formPanel.add(genPanel, gbc);


gbc.gridy = 8; gbc.gridx = 0; formPanel.add(new JLabel("Department:"), gbc);
cbDept = new JComboBox<>(new String[]{"Civil", "CSE", "Electrical", "E&C", "Mechanical"});
gbc.gridx = 1; formPanel.add(cbDept, gbc);


JButton btnSubmit = new JButton("Submit");

```

```
btnSubmit.addActionListener(e -> validateForm());
JButton btnCancel = new JButton("Cancel");
btnCancel.addActionListener(e -> System.exit(0));
JPanel btnPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
btnPanel.add(btnSubmit); btnPanel.add(btnCancel);
gbc.gridy = 9; gbc.gridx = 1; formPanel.add(btnPanel, gbc);
```

```
summaryArea = new JTextArea(15, 35);
summaryArea.setEditable(false);
summaryArea.setFont(new Font("Monospaced", Font.PLAIN, 12));
JScrollPane scroll = new JScrollPane(summaryArea);
scroll.setBorder(BorderFactory.createTitledBorder("Your Data is Below:"));
```

```
add(formPanel, BorderLayout.WEST);
add(scroll, BorderLayout.CENTER);
}
```

```
private void addLabelAndField(JPanel p, String lbl, JComponent field, GridBagConstraints gbc, int
y) {
    gbc.gridy = y; gbc.gridx = 0; p.add(new JLabel(lbl), gbc);
    gbc.gridx = 1; p.add(field, gbc);
}
```

```
private void updateDays() {
    int year = (int) cbYear.getSelectedItem();
    int month = cbMonth.getSelectedIndex() + 1;
    int days = java.time.YearMonth.of(year, month).lengthOfMonth();
    Integer currentDay = (Integer) cbDay.getSelectedItem();

    cbDay.removeAllItems();
```

```

        for (int i = 1; i <= days; i++) cbDay.addItem(i);
        if (currentDay != null && currentDay <= days) cbDay.setSelectedItem(currentDay);
    }

    private void validateForm() {
        StringBuilder sb = new StringBuilder();
        String mail = txtEmail.getText().trim();
        String p1 = new String(txtPass.getPassword());
        String p2 = new String(txtConfirmPass.getPassword());

        if (txtFirst.getText().trim().isEmpty() || txtLast.getText().trim().isEmpty()) sb.append("- Names required\n");
        if (!mail.matches("^([\\w-\\.]+@([\\w-]+\\.)+[\\w-]{2,4})$") || !mail.equals(txtConfirmEmail.getText().trim())) sb.append("- Emails invalid or mismatch\n");
        if (!p1.matches("(?=[a-zA-Z])(?=.*\\d){8,20}$") || !p1.equals(p2)) sb.append("- Password must be 8-20 chars with 1 letter & 1 digit\n");

        LocalDate birth = LocalDate.of((int)cbYear.getSelectedItem(), cbMonth.getSelectedIndex()+1, (int)cbDay.getSelectedItem());
        int age = Period.between(birth, LocalDate.now()).getYears();
        if (age < 16 || age > 60) sb.append("- Age must be 16-60 (Current: ").append(age).append(")\n");
        if (!rbMale.isSelected() && !rbFemale.isSelected()) sb.append("- Select Gender\n");

        if (sb.length() > 0) {
            JOptionPane.showMessageDialog(this, sb.toString(), "Validation Errors",
            JOptionPane.ERROR_MESSAGE);
        } else {
            processSubmit(birth, mail);
        }
    }

    private void processSubmit(LocalDate dob, String email) {

```

```
String id = "2026-" + String.format("%05d", studentCounter++);
```

```
String gen = rbMale.isSelected() ? "M" : "F";
```

```
String record = String.format("ID: %s | %s %s | %s | %s | %s | %s",
```

```
    id, txtFirst.getText().trim(), txtLast.getText().trim(),
```

```
    gen, cbDept.getSelectedItem(), dob, email);
```

```
summaryArea.setText(record);
```

```
try (PrintWriter pw = new PrintWriter(new BufferedWriter(new FileWriter("students.csv", true)))) {
```

```
    pw.println(record);
```

```
} catch (IOException ex) {
```

```
    JOptionPane.showMessageDialog(this, "File Error: " + ex.getMessage());
```

```
}
```

```
}
```

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

```
    SwingUtilities.invokeLater(() -> new StudentsRegistrationApp().setVisible(true));
```

```
}
```

```
}
```