A blue and yellow logo

Description automatically generated

**STIA1113**

**PROGRAMMING-1 (I)**

**INDIVIDUAL**

**ASSIGNTMENT-2**

**Prepared By : ANDYDERIS P.A.S**

**Prepared For : Dr. Baharudin Bin Osman**

**Date : 7 of Januari 2024**

*visit my repository on:* [*https://github.com/andyderis36/Assignment-2.git*](https://github.com/andyderis36/Assignment-2.git)

**1. INPUT/PROGRAM CODE**

import java.util.InputMismatchException;

import java.util.Scanner;

public class App {

private static Scanner scanner = new Scanner(System.in);

// Set the maximum size for arrays storing student data

private static int maxSize = 100;

// Declare arrays to store student data: names, matric numbers, carry marks, and final exam marks

private static String[] names = new String[maxSize];

private static int[] matrics = new int[maxSize];

private static double[] carryMarks = new double[maxSize];

private static double[] finalExamMarks = new double[maxSize];

// Initialize a variable to keep track of the number of students

private static int count = 0;

public static void main(String[] args) {

// Initialize test data

initializeTestData();

boolean exit = false;

while (!exit) {

// Display the main menu

System.out.println("\n--- STUDENT MANAGEMENT SYSTEM ---\n");

System.out.println("0. Exit");

System.out.println("1. Input Student Data");

System.out.println("2. View Student Data");

System.out.println("3. Edit Marks");

int choice = getUserChoice();

// Switch for menu choice

switch (choice) {

case 0:

exitProgram(); // Call method to exit the program

exit = true;

break;

case 1:

System.out.println("\n--- INPUT STUDENT DATA ---\n");

inputStudentData(); // Call method to input student data

break;

case 2:

viewAllNamesAndMatrics(names, matrics, count);

viewStudentData(); // Call method to view student data

break;

case 3:

System.out.println("\n--- EDIT MARKS ---\n");

editMarks(); // Call method to edit marks

break;

default:

invalidChoice(); // Call method for invalid choice

break;

}

}

// Close the scanner

scanner.close();

}

// Method to initialize test data

private static void initializeTestData() {

names[0] = "ANDYDERIS P.A.S";

matrics[0] = 296530;

carryMarks[0] = 50.0;

finalExamMarks[0] = 50.0;

count = 1;

}

// Method for invalid choice

private static void invalidChoice() {

System.out.println("\nInvalid choice!!! Please enter a valid choice from menu list given.");

System.out.println("Press Enter to try again...");

scanner.nextLine();

scanner.nextLine();

}

// Method for invalid input not integer

private static int getUserChoice() {

while (true) {

System.out.print("Enter your choice: ");

try {

return scanner.nextInt();

} catch (InputMismatchException e) {

System.out.println("\nInvalid input!!! Please enter integer/number");

System.out.println("Press Enter to try again...");

scanner.nextLine(); // Consume the newline character

scanner.nextLine();

}

}

}

// Method to input student data

private static void inputStudentData() {

if (count < maxSize) {

System.out.print("Enter Name: ");

scanner.nextLine(); // Consume the newline character left by previous nextInt

names[count] = scanner.nextLine();

System.out.print("Enter Matric: ");

matrics[count] = scanner.nextInt();

System.out.print("Enter Carry Mark (60%): ");

carryMarks[count] = scanner.nextDouble();

System.out.print("Enter Final Exam Mark (40%): ");

finalExamMarks[count] = scanner.nextDouble();

System.out.println("\nDATA INPUTED. Press Enter to show...");

scanner.nextLine(); // Consume the newline character

scanner.nextLine(); // Wait for the user to press Enter

// Displaying student data

double totalMark = calculateTotalMark(carryMarks[count], finalExamMarks[count]);

String grade = calculateGrade(totalMark);

System.out.println("--- STUDENT DATA ---\n");

System.out.println("Name: " + names[count]);

System.out.println("Matric: " + matrics[count]);

System.out.println("Carry Mark: " + carryMarks[count]);

System.out.println("Final Exam Mark: " + finalExamMarks[count]);

System.out.println("Total Mark: " + totalMark);

System.out.println("Grade: " + grade);

System.out.println("\nPress Enter to return Main Menu...");

scanner.nextLine(); // Wait for the user to press Enter

count++;

} else {

System.out.println("Array is full. Cannot add more students.");

}

}

// Method to view student data

private static void viewStudentData() {

System.out.print("\nEnter Matric to view full data: ");

int matricToView = scanner.nextInt();

int index = findStudentIndex(matricToView, matrics, count);

if (index != -1) {

viewFullData(names[index], matrics[index], carryMarks[index], finalExamMarks[index]);

} else {

System.out.println("\nSTUDENT NOT FOUND!!!");

viewStudentData();

}

System.out.print("\nPress Enter to return to the Main Menu.\n");

scanner.nextLine(); // Consume the newline character

scanner.nextLine(); // Consume the newline character

}

// Method to edit marks

private static void editMarks() {

System.out.print("Enter Matric to edit marks: ");

int matricToEdit = scanner.nextInt();

int index = findStudentIndex(matricToEdit, matrics, count);

if (index != -1) {

System.out.print("\nEnter new Carry Mark (60%): ");

carryMarks[index] = scanner.nextDouble();

System.out.print("Enter new Final Exam Mark (40%): ");

finalExamMarks[index] = scanner.nextDouble();

System.out.println("\nMarks Edited. Press Enter to show data...");

scanner.nextLine();

scanner.nextLine();

viewFullData(names[index], matrics[index], carryMarks[index], finalExamMarks[index]);

System.out.println("\nPress Enter to return Main Menu...");

scanner.nextLine();

} else {

System.out.println("\nStudent not found!!! Try to input listed Student...\n");

editMarks();

}

}

// Method to exit the program

private static void exitProgram() {

System.out.println("\nExiting the program. Goodbye!\n");

}

// Method to view all names and matrics

private static void viewAllNamesAndMatrics(String[] names, int[] matrics, int count) {

System.out.println("\n--- ALL STUDENTS LIST ---\n");

for (int i = 0; i < count; i++) {

System.out.println((i + 1) + ". |Name: " + names[i] + "|Matric: " + matrics[i] + "|");

}

}

// Method to view full student data with grade

private static void viewFullData(String name, int matric, double carryMark, double finalExamMark) {

double totalMark = calculateTotalMark(carryMark, finalExamMark);

String grade = calculateGrade(totalMark);

System.out.println("\n--- FULL DATA OF " + matric + " ---\n");

System.out.println("Name: " + name);

System.out.println("Matric: " + matric);

System.out.println("Carry Mark: " + carryMark);

System.out.println("Final Exam Mark: " + finalExamMark);

System.out.println("Total Mark: " + totalMark);

System.out.println("Grade: " + grade);

}

// Method to calculate total mark

private static double calculateTotalMark(double carryMark, double finalExamMark) {

return 0.6 \* carryMark + 0.4 \* finalExamMark;

}

// Method to calculate grade based on total mark

private static String calculateGrade(double totalMark) {

if (totalMark > 100) {

return "S+++";

} else if (totalMark >= 95) {

return "A+";

} else if (totalMark >= 90) {

return "A";

} else if (totalMark >= 85) {

return "B+";

} else if (totalMark >= 80) {

return "B";

} else if (totalMark >= 75) {

return "C+";

} else if (totalMark >= 70) {

return "C";

} else if (totalMark >= 65) {

return "D+";

} else if (totalMark >= 60) {

return "D";

} else if (totalMark >= 55) {

return "E+";

} else if (totalMark >= 50) {

return "E";

} else {

return "FAIL";

}

}

// Method to find the index of a student based on matric number

private static int findStudentIndex(int matric, int[] matrics, int count) {

for (int i = 0; i < count; i++) {

if (matrics[i] == matric) {

return i;

}

}

return -1;

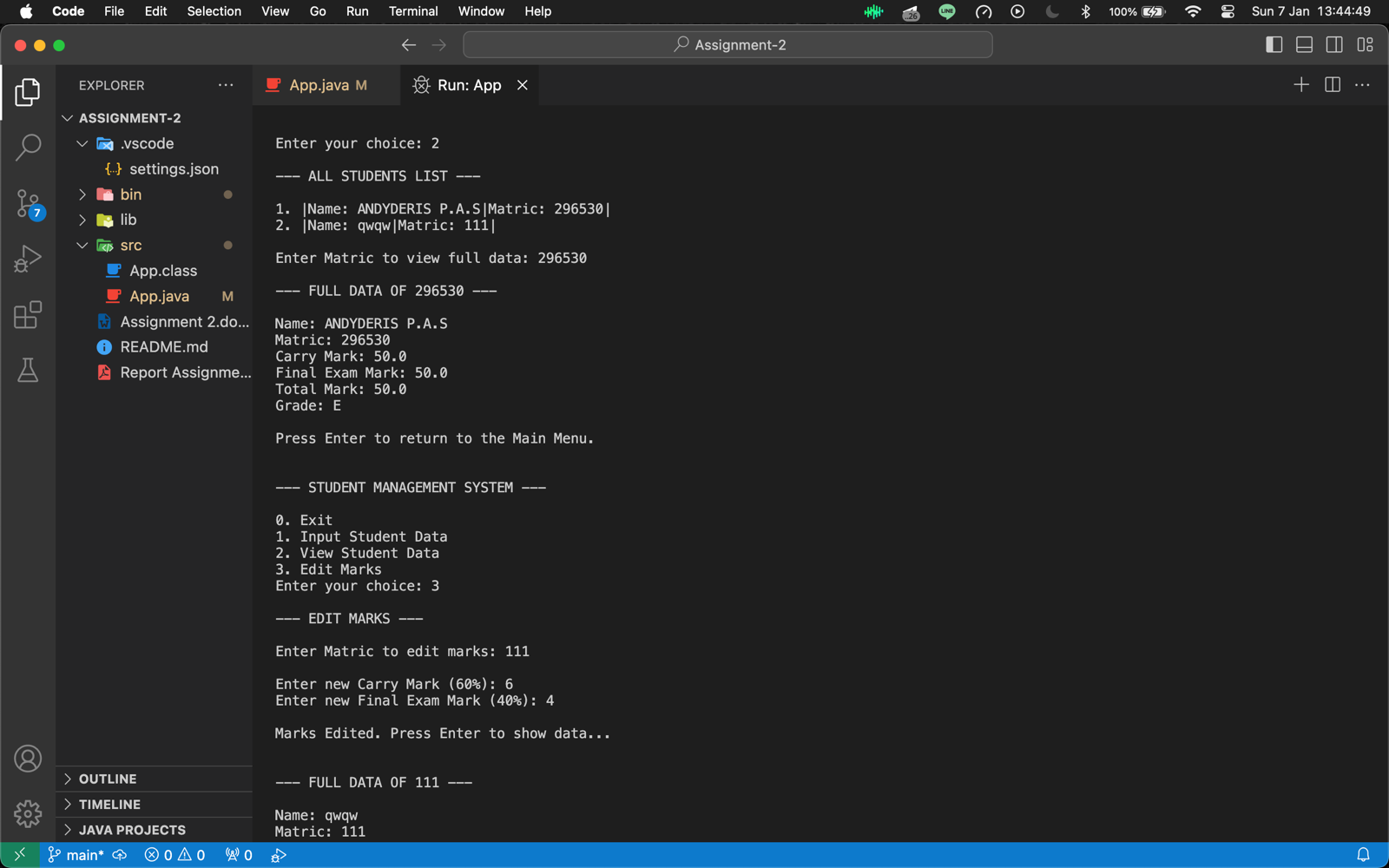
}

}

**2.OUTPUT**

**A screenshot of a computer

Description automatically generated**

****

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**