```java
/*
Author Name: Joshua Hernandez
Email: joshua.r.hernandez@okstate.edu
Data: October 8, 2023
Program Description: CS3353 Assignment 02
*/

package assignment02;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Header header = null;

        boolean fileInputted = false; // Boolean for if fileInput_1.txt has been inputted.

        int option;
        do {
            // Print the menu options
            System.out.println("CS3353 Assignment 2 Main Menu:");
            System.out.println("1. Read the input data");
            System.out.println("2. Delete a course");
            System.out.println("3. Insert a new course");
            System.out.println("4. Delete a student");
            System.out.println("5. Insert a new student");
            System.out.println("6. Transfer a student from one course to another");
            System.out.println("7. Display the course list");
            System.out.println("8. Display the student list");
            System.out.println("9. Exit");
            System.out.print("Enter your option: ");

            // Get user input
            option = scanner.nextInt();
            scanner.nextLine();

            switch (option) {
                case 1: Worse-case analysis: O(n) the buffered reader and Scanner will have to go (n) lines
                    // Read the input data
                    header = readInputFile();
                    fileInputted = true;
                    displayHeaderSummary(header); // Display header summary
                    break;

                case 2: Worse-case analysis: O(1)
                    // Delete a course
        if (fileInputted) {
            System.out.println("Input file data first");
            break;
        }
        System.out.print("Enter the course number to delete: ");
        String deleteCourseNumber = scanner.next(); // Get inputted course number
        header.deleteCourse(deleteCourseNumber);
        displayHeaderSummary(header); // Display updated summary
        break;

    case 3:  Worse-case analysis: O(1)
        // Insert a new course
        if (!fileInputted) {
            System.out.println("Input file data first");
            break;
        }
        System.out.print("Enter the new course number to add: ");
        String newCourseNumber = scanner.nextLine();

        if (!newCourseNumber.isEmpty()) {
            System.out.print("Enter the new course name for " + newCourseNumber + ":");
            String newCourseName = scanner.nextLine(); // Get inputted course name

            if (!newCourseName.isEmpty()) {
                assert header != null;
                header.insertCourse(newCourseNumber, newCourseName);
                displayHeaderSummary(header); // Display updated summary
            } else {
                System.out.println("Course name must not be empty");
            }
        } else {
            System.out.println("Course number must be entered");
        }
        break;

    case 4:  Worse-case analysis: O(1)
        // Delete a student
        if (!fileInputted) {
            System.out.println("Input file data first");
            break;
        }

        System.out.print("Enter the student ID number to delete: ");
        String studentID = scanner.nextLine();

        System.out.print("Enter the course number from which the student is the be dropped from: ");
        String courseNumber = scanner.nextLine();

        boolean studentDeleted = header.deleteStudent(courseNumber, studentID);
```

```
            if (studentDeleted) {
                displayHeaderSummary(header);
            } else {
                System.out.println("Cannot locate student");
            }

            break;

        case 5:  Worse-case analysis: O(1)
            // Insert a new student
            if (!fileInputted) {
                System.out.println("Input file data first");
                break;
            }
            // Get course number to enroll student
            System.out.print("Enter the course number the student wants to enroll to: ");
            String courseNumberToEnroll = scanner.nextLine();

            // Check if the course exist
            if (header.getCourse(courseNumberToEnroll) != null) {
                // Get the student's information
                System.out.print("Enter the student's name: ");
                String studentNameEnroll = scanner.nextLine();
                System.out.print("Enter the student's ID: ");
                String studentIDEnroll = scanner.nextLine();
                System.out.print("Enter the student's email");
                String studentEmailEnroll = scanner.nextLine();
                System.out.print("Enter the student's emergency contact address: ");
                String studentAddressEnroll = scanner.nextLine();

                //Add the student to course
                header.addStudentToCourse(courseNumberToEnroll, studentNameEnroll, studentIDEnroll,
studentEmailEnroll, studentAddressEnroll);

                // Display updated summary
                displayHeaderSummary(header);
            }

            break;

        case 6:  Worse-case analysis: O(n): Will most likely need to locate the student in the list
            // Transfer a student from one course to another
            if (!fileInputted) {
                System.out.println("Input file data first");
                break;
            }
            // Ask for student's name
            System.out.print("Enter the student's name:");
```

```
    String studentName = scanner.nextLine();

    // Ask for the course to drop
    System.out.print("Enter the course number the student wants to drop from:");
    String droppedCourseNumber = scanner.nextLine();
    // Find the course from the course number
    Courses droppedCourse = header.getCourse(droppedCourseNumber);

    if (droppedCourse != null) {
        // Locate the student in the dropped course
        Students transferStudent = droppedCourse.findStudentName(studentName);

        if (transferStudent != null) {
            // Ask for the course to add
            System.out.print("Enter the course number the student wants to enroll in:");
            String addedCourseNumber = scanner.nextLine();

            Courses addedCourse = header.getCourse(addedCourseNumber);

            if (addedCourse != null) {
                // Remove student from the dropped course
                droppedCourse.removeStudent(String.valueOf(transferStudent));
                // Add student to new course
                addedCourse.addStudent(transferStudent);

                // Display updated header summary information
                displayHeaderSummary(header);
            } else {
                System.out.println("New course not found");
            }
        } else {
            System.out.println("Student not found in the dropped course");
        }
    } else {
        System.out.println("Dropped course not found");
    }
    break;

case 7:  Worse-case analysis: O(n): This is for display every course, so it will iterate n times.
    // Display Course List
    if (!fileInputted) {
        System.out.println("Input file data first");
        break;
    }
    header.displayCourseList();
    break;

case 8:  Worse-case analysis: O(n): This will need to iterate n times for each student in the list
    // Display Student List
```

```
                if (!fileInputted) {
                    System.out.println("Input file data first");
                    break;
                }
                // Prompt user for course code
                System.out.print("Enter the course code: ");
                String courseNumberForList = scanner.nextLine();

                Courses courseStudentList = header.getCourse(courseNumberForList);

                if(courseStudentList != null) {
                    courseStudentList.displayCoursesStudentList();
                } else {
                    System.out.println("Course not found.");
                }
                break;

            case 9:  Worse-case analysis: O(1)
                System.out.println("Exiting");
                System.exit(0);
                break;

            default:
                System.out.println("Invalid option. Must enter a number between 1 and 9.");


        }
    } while (option != 9);
}
```

Worst-case complexity: O(1) + O(1) + O(1) + O(1) + O(1) + O(n) + O(n) + O(n) = O(n)