Title: Source Scheduling Optimization

The Objectives for this project:

Students should be able to demonstrate real life application of Using objects, Boolean expression, if statements, iteration,user input/reading a file, arraylist and write a class.

- Develop code to declare instance variables for the attributes to be initialized in the body of the constructors of a class.
- Develop code to define behaviors of a class through class methods.
- Develop code to declare the class variables that belong to the class.
- Develop code to designate access and visibility constraints to classes, data, constructors, and methods.
- A Develop code to define behaviors of an object through methods written in a class using primitive values and determine the result of calling these methods.
- Write program code to satisfy method specifications using expressions,conditional statements, and iterative statements.
- Develop code used to traverse the elements of an ArrayList and determine the results of these traversals.
- Develop code used to represent collections of related data using two dimensional(2D) array objects.
- Develop code to create Boolean expressions with relational operators and determine the result of these expressions.
- Develop code to represent iterative processes using for loops and determine the result of these processes.

Day 1 - Program Purpose & Overview

Objective: Understand the *why* behind the program

Topics:

- What the program does:
    - Reads in course data (name, credits, prerequisites).
    - Decides when each course can be taken based on prerequisites and credit limits.
    - Produces a schedule for multiple semesters.
- Real-world connections:
    - University degree planners.

Activity:

- Show the program output for a small example (5–6 courses).

  The student will manually create a schedule output by hand.

- Discuss: Why do we need prerequisites? What happens if we ignore them?

Day 2 - Input Handling

Objective: Learn how the program collects data safely from the user.

Topics:

- Using Scanner to get input from the console.
- Reading strings, numbers, and avoiding InputMismatchException.
- Difference between nextInt() and nextLine().
- Error handling for wrong inputs.

Simple code:

System.out.print("Enter the number of semesters: ");

int totalSemesters = scanner.nextInt();

scanner.nextLine();

Activity:

- Students practice entering valid and invalid data using the code below.

  https://codehs.com/sandbox/mathteachingdavid/coursescheduler/run

- Students start building their first section of the code using the scanner class and ask the and
  Start working on Step 1,2 and 3

Day 3 - Data Structures for Courses

Objective: Understand how to represent courses and prerequisites.

Topics:

- Creating a Course class with:
  - Name (String)
  - Credits (int)
  - Prerequisites (ArrayList<Integer> or course names)
- Storing multiple courses in an array or ArrayList.

Code Focus:

```
class Course {
   String name;
   int credits;
```

```
    ArrayList<String> prerequisites = new ArrayList<>();
}
```

Activity:

- Student will be working on step 4a to 4c on

  https://codehs.com/sandbox/mathteachingdavid/courseschedule

Day 4- Topological Sorting

Objective: Introduce Topological Sorting

Topics:

- Understanding Topological Sorting using a graph
- Use Topological Sorting to sort courses manually

1. Record the in-degree of each vertex in an array.
2. Place all vertices with an in-degree of zero into a queue.
3. While the queue is not empty:
    - Remove a vertex from the queue and output it.
    - Decrease the in-degree of each of its adjacent vertices by 1.
    - If any adjacent vertex's in-degree becomes zero, add it to the queue.

Resource :
https://opendsa-server.cs.vt.edu/ODSA/Books/Everything/html/GraphTopsort.html

Activity

Student will be working on Step 5 and 8
https://codehs.com/sandbox/mathteachingdavid/courseschedule

Day 5 – Scheduling Logic

Objective: Learn how the program assigns courses to semesters.

Topics:

- Loops for checking prerequisites.
- Tracking completed courses with a boolean[] or list.
- Balancing semesters so credit limits aren't exceeded.

Activity

Students will be working on Step 6

https://codehs.com/sandbox/mathteachingdavid/courseschedule

Code Focus:

```
boolean allMet = true;
for (String pre : course.prerequisites) {
   if (!completed.contains(pre)) {
      allMet = false;
      break;
   }
}
```

Activity:

- Students manually schedule 6–8 courses on paper using prerequisites.
- Compare with the program's output.

Day 6

Final Project Challenge

- **Have students create their own course list (10–12 courses, with prerequisites).**
- **Run the program and compare both options with their classmate.**
- **Discuss which schedule they'd prefer and why.**