CSCI 241 Data Structures
Project 2

Objectives

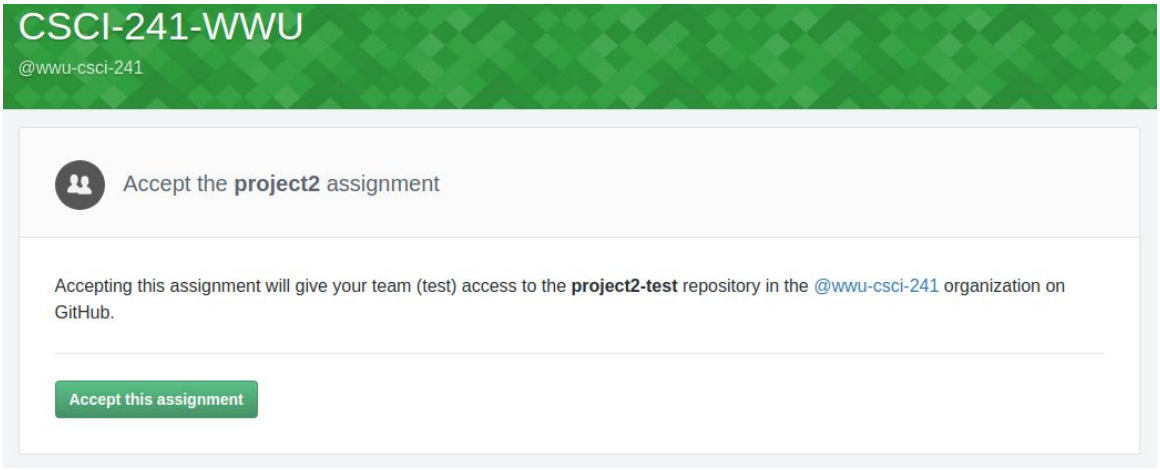
1. Practice DFS/BFS
2. Practice Topological Sorting

Part I Set Up a Team

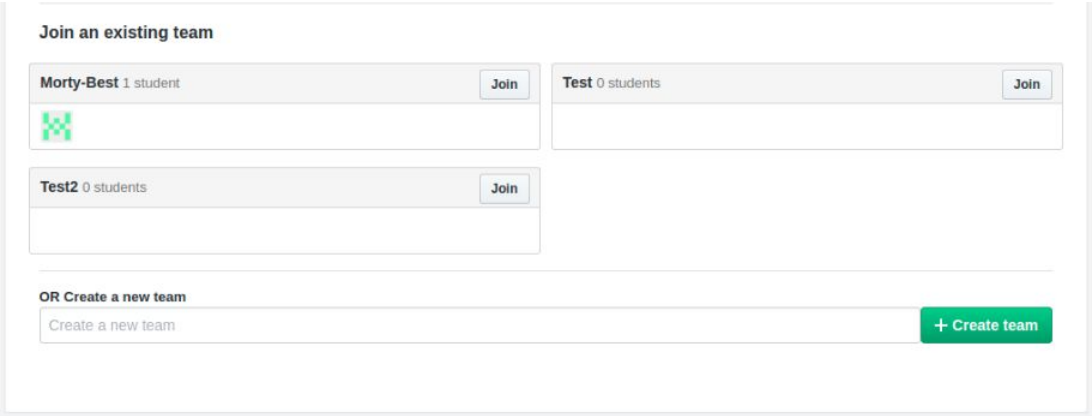
In this project, you will be working with other students as a team. A team can have a maximum of 3 members, and one member has to take the role as the team leader. Team leader will

- initiate meetings
- make sure every member takes a fair share of tasks
- lead the team to beat the deadline.

If you are a team leader, you will need to click this link (https://classroom.github.com/g/y8Yojmlt) to initialize a repository for your team. You will be asked to give a name to your team. In the following example, you will see that the team is named "Morty-Best".



If you are a team member, you will need to confirm that your team leader has done the above step. After that, you will need to click this link (https://classroom.github.com/g/y8Yojmlt) and choose your team among many teams, and click Join. In the following example, we will join the team named "Morty-Best".



Part II Automatic Course Planner

Suppose you are working on creating an automatic course planner for WWU students. A student can get a total of n courses, labeled from 0 to n - 1.

Some courses may have prerequisites, for example, to take course 241 you have to first take course 145, which is expressed as a pair: [241, 145] (*the current course, the prerequisite of the current course*).

Given the total number of courses and a list of prerequisite pairs, you need to:

- Determine if it is possible to finish taking the n courses. If not, print "It is impossible to take all the given courses." **If yes, print "It is possible to take all the given courses.", and move to the next step**.
- Generate one schedule of the given courses (prerequisites have to be scheduled first for any courses). Print the courses out with white spaces as separators. *There might be multiple valid schedules, but you only need to generate one of them*.

Example 1:

```
4, [[1,0], [2,0], [3,1]]
```

Output:

```
It is possible to take all the given courses.
A possible schedule is 0, 1, 2, 3.
```

Explanation:

There are a total of 4 courses to take. To take course 1 or 2 you should take course 0 first. To take course 3, you should take 1 first.

Example 2:

```
2, [[1,0],[0,1]]
```

Output:

```
It is impossible to take all the given courses.
```

Explanation:

There are a total of 2 courses to take. To take course 1 you should have finished course 0, and to take course 0 you should also have finished course 1.

This class needs to be named as "**CoursePlanner**". The methods that solve the problem are arranged as

- **plan**: this method returns no value and prints out (b) whether it is possible to take all the given courses and (b) what a possible schedule looks like.
- **check**: this method returns a boolean value to indicate whether it is possible to take all the given courses. This method can be seen as a helper for **plan**

Part III Test your planner

Create a testing class named "**PlanTest**". Under this class, there needs to be another static method besides the main method:

1. You will need to create a static method named "**prerequisiteGenerator**" under "**PlanTest**", which return a random 2D array containing prerequisite pairs (e.g., {{1, 3}, {2, 3}, {4, 2}}). The value range of a random integer is [0, 10] (inclusive). This method will accept an integer parameter, which specifies the length of the returned array:

   ```
   int[][] pre = prerequisiteGenerator(3); // possibly {{1, 3}, {2, 3}, {4, 2}}
   ```

2. Under the main method, you will need to use *prerequisiteGenerator* to generate random prerequisite pair lists, and perform tests on `plan` method at least three examples by printing proper messages.

   ```
   Given 4 courses and prerequisites as [[1,0], [2,0], [3,1]]
   It is possible to take all the given courses.
   A possible schedule is 0, 1, 2, 3.
   ```

Submission and Grading

You should open your browser and double check if such changes are reflected in your Github repo. Programs that don't compile will risk getting a zero. Late assignments are not accepted. Please use comments to explain your program or any sections that are possibly difficult to understand. **10% of the total lab points would be deducted if no comments can be found in your java files**. **10% of the total point would be deducted if there is no README file.**