Course Schedule II

Try to solve the Course Schedule II problem.

We'll cover the following

Statement

Let’s assume that there are a total of n courses labeled from 0 to $n - 1$. Some courses may have prerequisites. A list of prerequisites is specified such that if $Prerequisites_i = a, b$, you must take course b before course a.

Given the total number of courses n and a list of the prerequisite pairs, return the course order a student should take to finish all of the courses. If there are multiple valid orderings of courses, then the return any one of them.

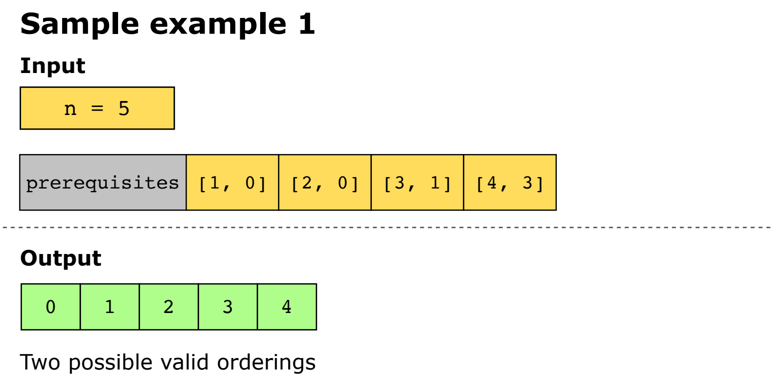
> **Note:** There can be a course in the 0 to $n - 1$ range with no prerequisites.

Constraints:

Let n be the number of courses.

- $1 \leq n \leq 2000$
- $0 \leq$ `prerequisites.length` $\leq n * (n - 1)$
- `prerequisites[i].length` $== 2$
- $0 \leq a,\ b < n$
- $a \neq b$
- All the pairs $[a,\ b]$ are distinct.

Examples



Sample example 1

Input

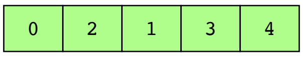
n = 5

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

Output

0 1 2 3 4

Two possible valid orderings



0	2	1	3	4

1 of 5

Understand the problem

Let’s take a moment to make sure you’ve correctly understood the problem. The quiz below helps you check if you’re solving the correct problem:

Course Schedule II

1 What is the course order if the following list of prerequisites and an integer n are given as input?

Select all that apply.

n = 7

prerequisites = [[1, 0], [1, 2], [2, 3], [3, 4], [4, 5]]

□ **A)** [1, 0, 2, 3, 4, 5, 6]

□ **B)** [0, 2, 1, 6, 5, 4, 3]

□ **C)** [0, 5, 6, 4, 3, 2, 1]

□ **D)** [6, 5, 4, 3, 2, 0, 1]

Submit Answer

Question 1 of 3
0 attempted

Reset Quiz

Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Drag and drop the cards to rearrange them in the correct sequence.

Create a graph with a node for each course and edges representing the dependencies. Store the indegrees of each node in a separate data structure.

Pick a node with in-degree equal to zero and add it to the output list.

Decrement the in-degree of the node picked in the previous step.

Repeat for all nodes with in-degree equal to zero.

Reset Show Solution Submit

Try it yourself

Implement your solution in the following coding playground:

Java

```java
import java.util.*;
class CourseSchedule{
    public static List <Integer> findOrder(int n, int[][] prerequisites) {
      // Write your code here

      return new ArrayList<>();
    }
}
```

Powered by AI Submit

Test Cases Results

Case 1 Case 2 Case 3

Input #1

3

Input #2

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

Course Schedule II

← Back Next →

☑ Mark as Completed