Description | Solution | Discuss (999+) | Submissions

C#

207. Course Schedule

Medium   👍 8144   👎 318   ♡ Add to List   ⭲ Share

There are a total of `numCourses` courses you have to take, labeled from `0` to `numCourses - 1`. You are given an array `prerequisites` where `prerequisites[i] = [a_i, b_i]` indicates that you **must** take course `b_i` first if you want to take course `a_i`.

- For example, the pair `[0, 1]`, indicates that to take course `0` you have to first take course `1`.

Return `true` if you can finish all courses. Otherwise, return `false`.

Example 1:

```
Input: numCourses = 2, prerequisites = [[1,0]]
Output: true
Explanation: There are a total of 2 courses to take.
To take course 1 you should have finished course 0. So it is
possible.
```

Example 2:

```
Input: numCourses = 2, prerequisites = [[1,0],[0,1]]
Output: false
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. So it is
impossible.
```

Constraints:

- $1 <= numCourses <= 10^5$
- $0 <= prerequisites.length <= 5000$
- `prerequisites[i].length == 2`
- $0 <= a_i, b_i < numCourses$
- All the pairs prerequisites[i] are **unique**.

≣ Problems   ⤬ Pick One   ‹ Prev   7/30   Next ›

```csharp
public class Solution {



    public bool CanFini
numCourses, int[][]
prerequisites) {


        Dictionary<int,
List<int>> courses = ne
Dictionary<int, List<in

        for(int i=0; i<
prerequisites.Length; i
        {

    if(courses.ContainsKey
sites[i][1]))
            {

    courses[prerequisites[
[1]].Add(prerequisites[
            }
            else{

    courses.Add(prerequisi
[1], new List<int> {
prerequisites[i][0] });
            }
        }

        bool[] check =
bool[numCourses];
            bool[] path = n
bool[numCourses];

        for (int currCo
currCourse < numCourses
++currCourse) {
```

Testcase | Run Code Result

Accepted   Runtime: 164 ms

Your input
```
2
[[1,0]]
```

Output   `true`

Expected   `true`

Example cases

Run Code ∧