Our Solution(s) Run Code

Solution 1 Solution 2

```cpp
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

#include <vector>
#include <unordered_map>
using namespace std;

class JobNode {
public:
  int job;
  vector<JobNode *> prereqs;
  bool visited;
  bool visiting;

  JobNode(int job);
};

class JobGraph {
public:
  vector<JobNode *> nodes;
  unordered_map<int, JobNode *> graph;

  JobGraph(vector<int> jobs);
  void addPrereq(int job, int prereq);
  void addNode(int job);
  JobNode *getNode(int job);
};

JobGraph *createJobGraph(vector<int> jobs, vector<vector<int>> deps);
vector<int> getOrderedJobs(JobGraph *graph);
bool depthFirstTraverse(JobNode *node, vector<int> *orderedJobs);

// O(j + d) time | O(j + d) space
vector<int> topologicalSort(vector<int> jobs, vector<vector<int>> deps
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```cpp
#include <vector>
using namespace std;

vector<int> topologicalSort(vector<int> jobs, vector<vector<int>> deps
  // Write your code here.
  return {};
}

```

Our Tests

Custom Output Submit Code

```
void Run() {
  RunTests("Test Case 1", [&]() {
    vector<int> jobs = {1, 2, 3, 4, 5, 6, 7, 8};
    vector<vector<int>> deps = {{3, 4}, {5, 6}, {6, 7}, {5, 7}, {8,
                                {4, 8}, {3, 8}, {1, 2}, {7, 8}};
    vector<int> order = topologicalSort(jobs, deps);
    assert(isValidTopologicalOrder(order, jobs, deps) == true);
  });

  RunTests("Test Case 2", [&]() {
    vector<int> jobs = {1, 2, 3, 4, 5, 6, 7, 8};
    vector<vector<int>> deps = {{3, 4}, {5, 6}, {6, 7}, {5, 7}, {8,
                                {4, 8}, {6, 7}, {3, 8}, {7, 8}};
    vector<int> order = topologicalSort(jobs, deps);
    assert(order == vector<int>{});
```

Run or submit code when you're ready.