EE599 - HW7
Computing and Software for Systems Engineers

- Unless specified: for each question that you write code:
    - Provide GTest.
    - Provide runtime analysis.
    - Proof of correctness is not necessary unless specified.
- For submission, please create a zip file of all of your assignments and only submit one file.
    - PLEASE REMOVE ALL FOLDERS STARTING WITH **bazel-*** before submitting. **To do this run : `bazel clean`**
    - Leave any extra instructions for the graders in a README text file.
    - Our grader should be able to call blaze run/test ... and run your code/test.
    - **For the purpose of faster grading, please put your code in one solution.cc / solution.h, test.cc, and main.cc.**
- Deadline: March 14, before 6pm.
- Total: 120 points. 100 points is considered full credit.
- **Academic Conduct**
    - Students are encouraged to collaborate on general solution strategies for homework. The writeup, however, must be your own - **you may not copy someone else's solution**.
    - In addition, your homework should list all the fellow students with whom you discussed the solutions.

For all the questions, If not specified, assume a graph is given using this class:

```
class Graph {
public:
 Graph(std::map<int, std::set<int>> &vertices) :
v_(vertices) {}
 std::map<int, std::set<int>> v_;
};

int main() {
 std::map<int, std::set<int>> vertices{
      {1, {2, 3}},
      {2, {1, 3, 4, 5}},
      {3, {1, 2, 4}},
      {4, {2, 3, 4}}
 };
 Graph g(vertices);
}
```

Question 1 (10 Points. Easy)

Complete or select the correct word in the following statements:

- What is the definition of a path in a graph?
- A simple path is a path that _____
- A cycle is a path in which _____
- Topological sort is defined in graphs that are _____

Question 2 (20 Points. Easy)

In the lecture we saw the steps of the Floyd-Warshall algorithm for a graph with 5 nodes:

	0	1	2	3	4	5	6
0	**0**	3	10	∞	∞	1	∞
1	3	**0**	6	1	∞	∞	∞
2	10	6	**0**	1	∞	∞	∞
3	∞	1	1	**0**	1	∞	4
4	∞	∞	∞	1	**0**	∞	∞
5	1	∞	∞	∞	∞	**0**	1
6	∞	∞	∞	4	∞	1	**0**

k=0

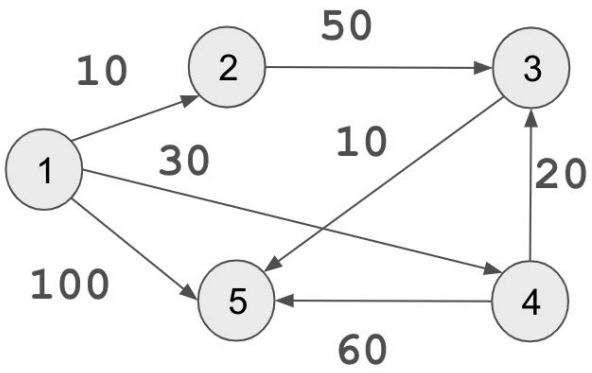
	0	1	2	3	4	5	6
0	0	3	10	∞	∞	1	∞
1	3	0	6	1	∞	**4**	∞
2	10	6	0	1	∞	**11**	∞
3	∞	1	1	0	1	∞	4
4	∞	∞	∞	1	0	∞	∞
5	1	**4**	**11**	∞	∞	0	1
6	∞	∞	∞	4	∞	1	0

k=1

	0	1	2	3	4	5	6
0	0	3	**9**	**4**	∞	1	∞
1	3	0	6	1	∞	4	∞
2	**9**	6	0	1	∞	**10**	∞
3	**4**	1	1	0	1	**5**	4
4	∞	∞	∞	1	0	∞	∞
5	1	4	**10**	**5**	∞	0	1
6	∞	∞	∞	4	∞	1	0

k=2

Similarly, please write down the steps for the following graph:



You should create a 5x5 matrix and modify it for k=0 to k=5. Please create these tables manually!

No GTest necessary. Please submit an image of the tables.

Question 3 (40 Points. Medium)

Given the following code for BFS:

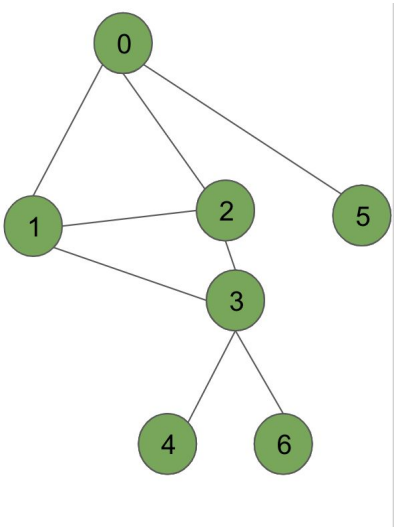
```
void Graph::BFS(int root) {
 std::map<int, int> marks;
 std::queue<int> q;
 q.push(root);
 marks[root] = 1;
 while (!q.empty()) {
   int cur = q.front();
```

```
    q.pop();
    for (auto &n : edge_map_[cur]) {
      if (!marks[n]) {
        marks[n] = 1;
        q.push(n);
      }
    }
  }
}
```

1. Enhance it to calculate the **shortest distance** for each node. The output should be a map that maps each node to its distance.
2. Enhance it to find the **actual shortest path** from the root to each node. The output should be a map that maps each node to a vector that represents the shortest path.

Example:

Input: the following graph, root = 0



Output:

1. Shortest distances: { (0,0), (1,1), (2,1), (5,1), (3, 2), (4, 3), (6,3) }

2. Shortest paths: { (0, [0]), (1, [0,1]}, (2, [0, 2]), (5, [0,5]), (3, [0,1,3]), (4, [0,1,3,4]), (6, [0,1,3,6])}

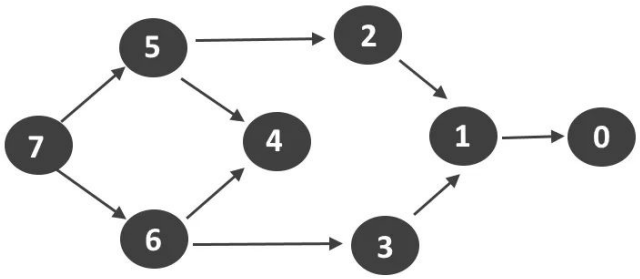
Question 4 (20 Points. Easy)

For the Graph class defined in the beginning of the assignment, write a function that:

1. finds root nodes and report them in a set. A root node is a node that doesn't have any incoming edges.
2. outputs a vector of nodes sorted in topological order.

You can assume that the graph is directed and acyclic.

Example:
Input:



Output:

1. Root nodes: {7}
2. Topological order: [7, 6, 5, 4, 3, 2, 1, 0]

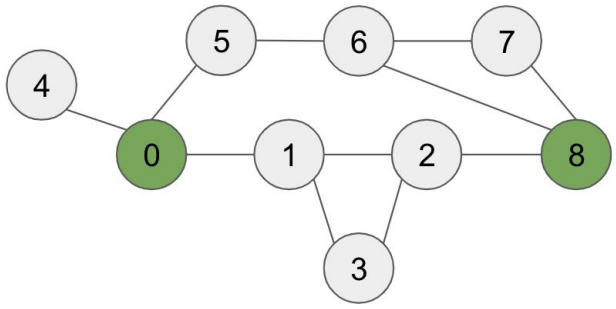
Question 5 (30 Points. Medium-Hard)

You are given an undirected graph G with N nodes indexed from 0 to N-1. For each node i in the graph G, determine if there exists one shortest path from node 0 to node N-1 that passes through i:

- Write a function that takes the graph G as the input
- Output a vector of bool B, where B[i] stands for whether there exists a shortest path from 0 to N-1 that passes through node i.

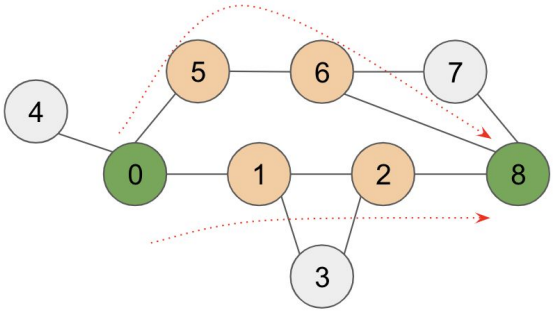
Example:

Input:



Output:
[**true**, **true**, **true**, false, false, **true**, **true**, false, **true**]

Explanation:
There are two possible shortest path from 0 to 8 each having length of 3:



0 - 1 - 2 - 8
And
0 - 5 - 6 - 8

Therefore, the output vector at indices 0, 1, 2, 5, 6, 8 is set to true because the shortest path between 0 and 8 passes through these nodes.

👉**Hint:** Try to think of an efficient algorithm for the shortest path when the graph is unweighted.

Optional Questions

The goal of this section is to introduce you to more challenging questions and some common problems in coding and algorithms.

- These questions don't have any credits.
- We may not provide complete solutions or grading for them.

- Solving them is completely optional.

Question 1 (Medium)

Given two integers ‘n’ and ‘m’, find all the stepping numbers in the range [n, m]. A number is called a stepping **number** if all adjacent digits have an absolute difference of 1.

For example: 321 is a Stepping Number while 421 is not.

👉**Hint:** Use BFS/DFS

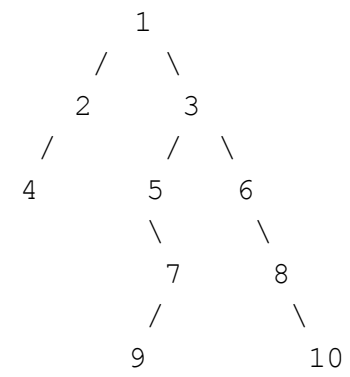
Examples:

Input : n = 0, m = 21
Output : 0 1 2 3 4 5 6 7 8 9 10 12 21

Input : n = 10, m = 15
Output : 10, 12

Question 2 (Medium)

Given a binary tree, find the leftmost value in the last row of the tree.
Example :



Output : 9

Question 3 (20 Points. Medium)

In a grid each cell has one of the three values - 0 represents an empty cell, 1 represents a fresh orange, 2 represents a rotten orange.
Every minute, any fresh orange adjacent (4-directionally: top, bottom, left and right) to a rotten orange becomes rotten. Write a C++ program that returns the minimum number of minutes that must elapse until no cell has a fresh orange. If this is impossible return -1.

Example 1:
Input: [[2,1,1], [1,1,0], [0,1,1]]
Output: 4

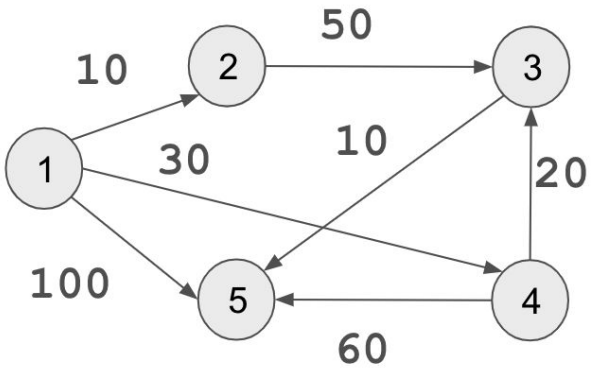
Example 2:
Input: [[2,1,1], [0,1,1], [1,0,1]]
Output: -1

Question 4 (Medium)

Did you know that in a directed acyclic graph (DAG), you can find the shortest distance from a source to all other nodes in linear time? Write a function that takes a directed acyclic graph, a root node, and outputs the distance from that node to all other nodes.

👉**Hint:** Use topological sort

Example :



`Root : 1`

Output : 2: 10, 3: 50, 4: 30, 5: 60.
Explanation: The shortest distance from node 1 to node 2 is 10.
Similarly, the shortest distance from node 1 to node 3 is 50.
From node 1 to node 4 is 30 and from node 1 to node 5 is 60.