LEBANESE AMERICAN UNIVERSITY School of Arts and Science

Department of Computer Science and Mathematics

CSC 310: Algorithms and Data Structures

Lab 5

Problem 1 –

Given an undirected graph, you are required to print the vertices of the graph using Breadth

First Search.

Input

Your program will be tested against multiple test cases. Each test case consists of two integers **n** and **e**, representing the number of vertices and edges respectively. The next **e** lines represent the vertices that are connected by an edge.

Output

For each test case, print the graph using BFS.

Sample Input Sample Output

6 6 0 1 3 2 4 5

0 1

0 3

1 2

2 4

3 4

3 5

7 5 0 1 2 3 4 5 6

0 1

0 2

1 2

3 4

5 6

Problem 2 – Graph Connectivity

Given an undirected graph, you are required to check if the graph is connected or not.

Input

Your program will be tested against multiple test cases. Each test case consists of two

integers **n** and **e**, representing the number of **vertices** and **edges** respectively. The next **e** lines represent the vertices that are connected by an edge.

Output

For each test case, print “connected” if the graph is connected; else print “not connected”.

Sample Input Sample Output

6 6 connected

0 1

0 3

1 2

2 4

3 4

3 5

7 5 not connected

0 1

0 2

1 2

3 4

5 6

Problem 3 – Connected Components of a Graph:

Given an undirected graph, you are required to find the number of connected components and print the vertices of each component.

Input

Your program will be tested against multiple test cases Each test case consists of two integers **n** and **e**, representing the number of **vertices** and **edges** respectively. The next **e** lines represent the vertices that are connected by an edge.

Output

For each test case, print the number of connected components.

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| 6 4 | 2 |
| 0 1 | 0 1 2 |
| 0 2   1. 4 2. 5 | 3 4 5 |
| 8 4 | 4 |
| 0 1 | 0 1 |
| 2 3 | 2 3 |
| 4 5 | 4 5 |
| 6 7 | 6 7 |

Problem 4 – Acyclic Graph:

Given an undirected graph, you are required to check if the graph has a cycle.

Input

Your program will be tested against multiple test cases. Each test case begins with two integers **n** and **e**, representing the number of **vertices** and **edges**. The next **e** lines represent the vertices that are connected by an edge.

Output

For each test case, print “no” if the graph contains a cycle; else print “yes”.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sample Input |  |  |  | Sample Output |
| **6 6**  0 1   1. 3 2. 2 3. 4 4. 4   3 5 | no |  |  |  |
| **6 5**  0 1  0 2   1. 3 2. 4 3. 5 | yes |  |  |  |

Problem 5– Bipartite Graphs:

Given an undirected graph, you are required to check if the graph is bipartite.

# Input

Your program will be tested against multiple test cases. Each test case begins with two integers n and e, representing the number of vertices and edges. The next e lines represent the vertices that are connected by an edge.

# Output

For each test case, print “bipartite” if the graph is bipartite; else print “not bipartite”.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| **6 6** | not bipartite |
| 0 1 |  |
| 0 3 |  |
| 1 2 |  |
| 2 4 |  |
| 3 4 |  |
| 3 5 |  |
| **4 4** | bipartite |
| 0 1 |  |
| 0 3 |  |
| 1 2 |  |
| 2 3 |  |

Problem 6– Tree Graphs:

Given an undirected graph, you are required to check if the graph is a Tree.

# Input

Your program will be tested against multiple test cases. Each test case begins with two integers n and e, representing the number of vertices and edges. The next e lines represent the vertices that are connected by an edge.

# Output

For each test case, print “tree” if the graph is tree; else print “not tree”.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| **5 4** | tree |
| 0 1 |  |
| 0 2 |  |
| 0 3 |  |
| 3 4 |  |
|  |
|  |
| **4 4** | not tree |
| 0 1 |  |
| 0 3 |  |
| 1 2 |  |
| 2 3 |  |