COMP2521 Data Structures & Algorithms Final Exam

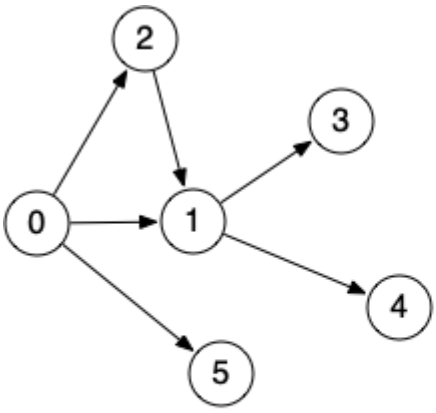
[Instructions] [Website] [C]
[Q1] **[Q2]** [Q3] [Q4] [Q5] [Q6] [Q7] [Q8]

Question 2 (15 marks)

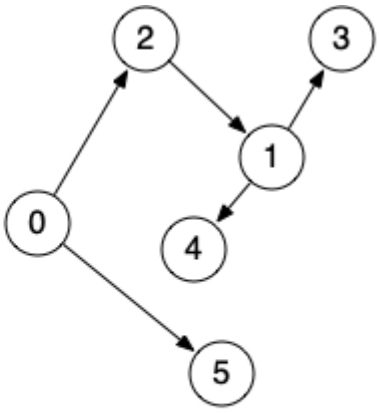
Trees can be viewed as a special case of directed graphs. A directed graph (digraph) is a binary tree provided it satisfies the following conditions:

- only one node has no incoming edges
- all other nodes have exactly one incoming edge
- nodes may have 0, 1 or 2 outgoing edges

The following diagram shows two example directed graphs, only one of which is an binary tree:



DiGraph is not a binary tree
0 has three outgoing edges
1 has two incoming edges



DiGraph is a binary tree
All nodes have 1 incoming edge
All nodes have <=2 outgoing edges

Graphs are represented internally using a `DiGraph` data structure. Graphs are read from files which look like:

```
0  1 2 5
1  3 4
2  1
3  -
4  -
5  -
```

This is the readable representation of the graph on the left above. Each line contains a node number and then a list of all the nodes that this node has outgoing edges to.

We have provided a program that reads in a representation of a graph, builds a `DiGraph` data structure (adjacency matrix), dispay the graph structure on standard output, and then calls a function to determine whether the graph is a binary tree. Your task is to complete the `graphIsBinTree()` function, which takes a `DiGraph`, and returns `true` if the graph satisfies the three conditions given above, and returns `false` otherwise.

The `graphIsBinTree()` function is defined as:

```
bool graphIsBinTree(DiGraph g)
```

Note that the `DiGraph` object is not passed to the function via a pointer. The whole structure is copied to the function to be tested.

The code for this question is in the `q2` directory, which contains:

- `gist.c` ... program for testing graph-is-binary-tree
- `Makefile` ...for building the program
- `tests/` ... directory containing test cases
- `run_tests.sh` .. script for runnign the tests

You ought to look at the data structures, the main program and the reading and writing functions, to familiarise yourself with the enviroment in which the `graphIsBinTree()` function operates.

You can compile the program using the `make` command. This will give you an executable file called `gist`. As supplied, the `graphIsBinTree()` function always returns `false`, regardless of the `DiGraph`.

You should complete the `graphIsBinTree()` function. You can define as many auxiliary functions as you want. It requires around 15-20 lines of code to solve this problem.

You should *not* change any other part of `gist.c` except for the `graphIsBinTree()` function, and any extra helper functions that you want to add. If you change the input and output functions or the `main()` function, then you will probably fail all of the tests.

Submissions that don't compile are worth zero marks. Submissions that compile, but fail all tests are worth up to 7 marks. Submissions that compile and pass some tests are worth between 8 and 14 marks, depending on how close they are to a working solution. Submissions that compile and pass all tests are worth 15 marks

Examples of how the program ought to behave when working correctly:

```
$ ./gist tests/g1
V    Connected to
--   ------------
0    1 2 5
1    3 4
2    1
3    -
4    -
5    -

Graph is not a tree

$ ./gist tests/g2
V    Connected to
--   ------------
0    2 5
1    3 4
2    1
3    -
4    -
5    -

Graph is a tree
```

To help you check whether your program is working correctly, there is a script called `run_tests.sh` which will run the program against all of the tests and report the results. It will also add the output from your program into the `tests` directory; comparing your output against the expected output might help you to debug your code. You can run the testing script as:

```
$ sh run_tests.sh
```

Once your function is working (passes all tests), follow the submission instructions below. Even if it fails some (or even all) tests, you should submit because you can get *some* marks. If your program does not compile, or if you simply submit the supplied code, then your "answer" is worth zero marks.

Submission Instructions:

- Type your answer to this question into the file called `gist.c`
- Submit via: **`give cs2521 exam_q2 gist.c`**
  or via: Webcms3 > exams > Final Exam > Submit Q2 > Make Submission

End of Question