hasCycle

Your task is to write a function, hasCycle, that determines whether or not the given graph contains a cycle. It should return true if the graph contains a cycle, and false otherwise.

Note: You are provided with a stack ADT, but you are not required to make use of it.

Download

Click here to download a zip of the files.

The Files

Graph.c	Contains the implementation of a graph ADT
Graph.h	Contains the interface of the graph ADT
Stack.c	Contains the implementation of a stack ADT
Stack.h	Contains the interface of the stack ADT
testHasCycle.c	Contains the main function, which reads in a graph from standard input, calls hasCycle, and prints out the result.
hasCycle.c	Contains hasCycle, the function you must implement
Makefile	A makefile to compile your code
tests/	A directory containing the inputs and expected outputs for some basic tests
autotest	A script that uses the tests in the tests directory to autotest your solution. You should only run this after you have tested your solution manually.

Examples

Your program should behave like these examples:

```
$ ./testHasCycle
Enter number of vertices: 8
Enter number of edges: 7
Enter edges in the form v-w: 0-1 0-2 0-3 1-6 2-5 3-4 3-7

Graph: nV = 8
Edges:
 0: 0-1 0-2 0-3
 1: 1-0 1-6
 2: 2-0 2-5
 3: 3-0 3-4 3-7
 4: 4-3
 5: 5-2
 6: 6-1
 7: 7-3

hasCycle returned: FALSE
```

```
$ ./testHasCycle
Enter number of vertices: 9
Enter number of edges: 9
Enter edges in the form v-w: 0-1 0-5 1-3 1-6 2-8 3-4 5-8 6-7 6-8

Graph: nV = 9
Edges:
 0: 0-1 0-5
 1: 1-0 1-3 1-6
 2: 2-8
 3: 3-1 3-4
 4: 4-3
 5: 5-0 5-8
 6: 6-1 6-7 6-8
 7: 7-6
 8: 8-2 8-5 8-6

hasCycle returned: TRUE
```

Hints

Only look at these hints if you are stuck.

Hint 1

Testing

You can test your program manually by compiling your code using `make`, and then running `./testHasCycle`, as shown above. After you are satisfied with your solution, you can autotest it by running `./autotest`. This will run some basic tests on your program.