

Due 1:30PM Wednesday, November 3

Name _____

Student ID _____

Instructions Follow instructions *carefully*, failure to do so will result in points being deducted. It is highly recommended that you write your homework using L^AT_EX or Word. Staple this cover page to the front of your assignment for easier grading. Be sure to submit a hardcopy of your problem answers and your program analysis in class. Clearly label each problem and submit the answers in order. Put your program analysis to the end. In addition, submit your homework including all the program files via the webhandin (<http://www.cse.unl.edu/~cse310/handin>). Late homework policy specified in course syllabus *will be strictly followed*. Be sure to show *sufficient work* to justify your answer(s). Numerical answers with no explanation will NOT be granted full points. If you are asked to prove something, you must give as formal, rigorous, and complete proof as possible. You are to work individually, and all work should be your own. The CSE academic dishonesty policy is in effect (http://cse.unl.edu/undergrads/academic_integrity.php).

Problem	Page	Points	Score
5.1.2	163	10	
5.1.3	163	10	
5.3.10	177	10	
5.4.4(a)(c)	182	10	
5.5.9	187	10	
Program			
Correctness		30	
Style/Documentation		10	
Analysis		10	
Total		100	

Program

Write a program (based on a graph traversal algorithm you've learned in this class) that, for a given undirected graph, outputs: (i) vertices of each connected component; (ii) a cycle or a message that the graph is acyclic (if there are more than one cycles in a graph, you are required to output just one of them).

Your program must be written in C++, execute correctly on the CSE unix environment, and compile using the `g++` compiler using a `makefile`. The `makefile` should compile your program into an executable called `graph_inspector`. Your `main.cpp` program should take input from a file via the command line (example: `prompt:>graph_inspector inputfile`) with the following structure in the input file (you may assume that an input file will be well structured so you don't have to error check the input).

Each line of the input file represents a graph. The first number in a line specifies the number of vertices in the graph. Then pairs of nodes define the edges.

An example of an input file is as follows:

```
5 (1,2) (3,4) (3,5) (4,5)
4 (1,2) (2,3) (1,4)
```

It specifies two graphs. The first graph has five vertices (1,2,3,4,5) and four edges. The second graph has four vertices (1,2,3,4) and three edges.

Proper output should look (something) like:

```
////////////////////////////////////
Graph1:
Two connected components: {1 2} {3 4 5}
Cycle detected: 3 - 4 - 5 - 3

Graph2:
One connected component: {1 2 3 4}
The graph is acyclic.
////////////////////////////////////
```

Points will be awarded based on the following categories:

- **Correctness** – Your code must compile and execute as expected. You must include a `makefile` that will compile your code into an executable called `graph_inspector`.
- **Style/Documentation** – Your code should be readable, properly indented and spaced with sufficient comments to explain. Good naming style should be followed.
- **Analysis** – For this homework, your analysis should include two parts. First, you need to explain your design of the algorithms. Second, you analyze the algorithms and give time and space efficiencies as a conclusion. Discuss anything you think relevant, pitfalls that you faced, etc. Turn in your analysis as a hard copy with the rest of your assignment.