CS271: Data Structures

Instructor: Dr. Stacey Truex

Project #7

This project is meant to be completed in groups. You should work in your Unit 4 groups. Implementation solutions should be written in `C++`. Exercise solutions should be included in `.txt` files. Only one submission (the last submission uploaded to canvas) will be graded per group. Submissions should be a compressed file following the naming convention: `NAMES_cs271_project7.zip` where `NAMES` is replaced by the first initial and last name of each group member. For example, if Dr. Truex and Dr. Chavrimootoo were in a group they would submit one file titled `STruexMChavrimootoo_cs271_project7.zip`. **You will lose points if you do not follow the course naming convention**. Your `.zip` file should contain a *minimum* of 4 files:

1. `proof.pdf`
2. `coloring.txt`
3. `bipartite.cpp`
4. `commits.pdf`: a commit history for your GitHub project

Additional files such as a `README.md` are welcome. The above merely represent the minimum files required for project completion. Your code is expected to implement an `is_bipartite` function for the provided `Graph` class. Details for each part of the project are as follows.

Definitions

Definition 1. [Proper Coloring] A proper coloring is an assignment of colors to the vertices of a graph so that no two adjacent vertices have the same color.

Definition 2. [Chromatic Number] The chromatic number of a graph is the minimum number of colors in a proper coloring of that graph.

Definition 3. [Clique] A clique is a set of mutually adjacent vertices. Specifically, if a graph G is a clique, then $\forall u \in G.V$

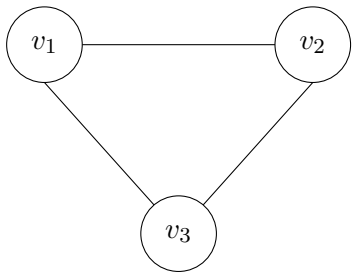
$$G.Adj[u] = G.V - \{u\} \ .$$

Definition 4. [Bipartite Graph] A bipartite graph is a graph $G = (V, E)$ whose vertices can be partitioned into two sets ($V = V_1 \cup V_2$ and $V_1 \cap V_2 = \emptyset$) such that there are no edges between vertices of the same set (for instance, if $u, v \in V_1$, then there is no edge between u and v).

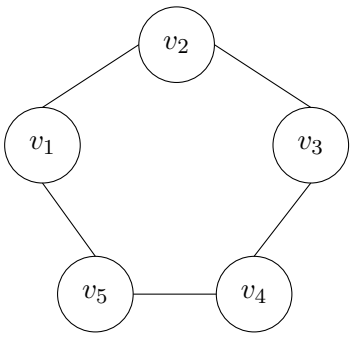
Coloring

1. Consider each of the following graphs. Edit the `coloring.txt` file to show a proper coloring of each using only the chromatic number of colors. After each vertex colon, add a number value to indicate its color. Do not change the file in any other way.

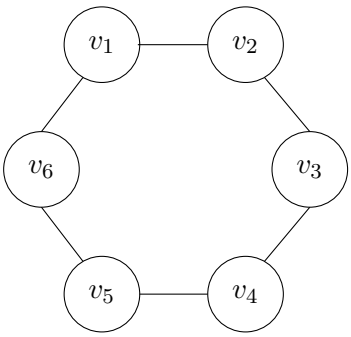
(a)



(b)

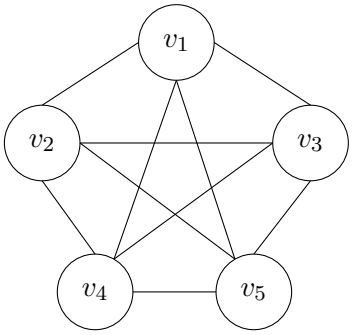


(c)



2. Coloring of cliques.

   (a) Edit the `coloring.txt` file to show a proper coloring of the following clique using only the chromatic number of colors. After each vertex colon, add a number value to indicate its color. Do not change the file in any other way.



   (b) What can you say about the relationship between $|V|$ in a graph $G = (V, E)$ and G's chromatic number if G is a clique? Respond in `coloring.txt` following the prompt `ANSWER:`

Implementation

Design an algorithm that has an average-case linear time complexity to determine whether an undirected graph G is bipartite. Implement your solution in `C++` in the provided `bipartite.cpp` file. It may be helpful to review *formal documentation* for an `unordered_set` and `unordered_map` to understand the complexity of their operations.

Unit Testing

An example test file `test_bipartite_example.cpp` has been provided along with example test files in `bipartite` and `non_bipartite_graphs` folders. Syntax is consistent with the graphs in Project 6.

Documentation

The expectation of all coding assignments is that they are well-documented. This means that logic is documented with line comments and method pre- and post- conditions are properly documented immediately after the method's parameter list.

Pre-conditions and post-conditions are used to specify precisely what a method does. However, a pre-condition/post-condition specification does not indicate how that method accomplishes its task (if such commenting is necessary it should be done through line level comments). Instead, pre-conditions indicate what must be true before the method is called while the post-condition indicates what will be true when the method is finished.

Proofs

Use LaTeXto complete a formal proof of the following in your `proof.pdf` file:

> An undirected graph is bipartite if and only if it contains no cycles of odd length.

Rubric

Note that any coding projects that do not compile with the provided `test_bipartite_example.cpp` file will be given a 0. All projects that are able to be successfully compiled will be graded using the following rubric.

Coloring	**15 Total Points**		
	correct coloring provided in `coloring.txt`		3 pts each
	correct conclusion in 2(b)		3 pts
C++ Implementation	does not compile: 0/25		
	25 Total Points		
	Code	**Correctness** passes unit testing	20 pts
		Validation average-case linear time complexity	5 pts
	Documentation	**Documentation**	3 pts
		extremely sparse documentation	0/3
		missing comments or pre- and post-conditions	1/3
		documentation lacks detail in areas	2/3
		detailed comments & pre- and post-conditions	3/3

LaTeX Proof PDF	**3 Total Proof Points**	
	Correctness	3 pts
	incomplete, barebones, or missing	0/3
	demonstrates significant error in understanding either proof logic or underlying concept	1/3
	errors in writing or small error in logic	2/3
	well-written, complete proof	3/3