CSE2046/CSE2246 Spring 2022

Assignment 2: Graph Coloring Problem

Due date: June 5th, 2021 (for the test outputs), June 7th, 2021 (for the report and codes)

In this assignment, you are asked to design and implement an algorithm for Graph Coloring Problem that is described below.

Graph Coloring (or Vertex Coloring) problem is defined as assignment of smallest number of colors to the vertices of an undirected graph, such that no adjacent vertices are of the same color. A coloring using at most k colors is called a (proper) k-coloring. The smallest number of colors needed to color a graph G is called its chromatic number and is often denoted $\chi(G)$. The goal is to color all vertices of the graph, such that number of colors used (k) is as close as possible to the optimal result, i.e. $\chi(G)$. Since this problem is NP-Hard, it is very difficult to find the optimal solution especially for large instances.

Graph Coloring can be used to model many practical problems, such as event scheduling.

Project Specification:

Your team (up to 3 students) is asked to design and implement an algorithm for Graph Coloring problem described above. (Please do not ask for the possibility of 4-student group).

Your goal is not to design an algorithm for the optimal solution, but you are requested to do your best. This is an open-ended assignment.

You may do the following:

- Read as much as you want to learn about how to solve the Graph Coloring problem. But **you have to cite** any resources you use. You may want to start with some approximation algorithms (such as local search heuristics) described in chapter 12.
- You may use whichever programming language you want.

You may **not use** the following:

- Existing implementations or subroutines
- Extensive libraries (if you are not sure, check with the instructor)
- Other people's code (including the existing codes in github or elsewhere).

Input format: Inputs will always be given as a text file. Input file format should be as follows:

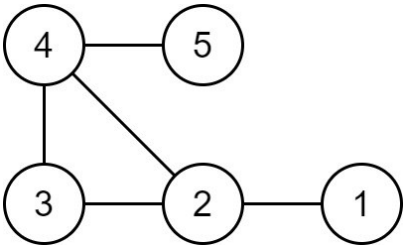
The first line is the problem line which indicates the number of vertices and the number of edges. It starts with "p", and there is only one such line.

```
p <NumVertices> <NumEdges>
```

The rest of the lines are edge lines. Each line starts with “e” and indicates an edge between two vertices.

```
e <VertexNumber1> <VertexNumber2>
```

The above line means that there is an edge between VertexNumber1 and VertexNumber2. Here is a sample graph:



This graph can be represented as the following input:

```
p 5 5
e 1 2
e 2 3
e 2 4
e 3 4
e 4 5
```

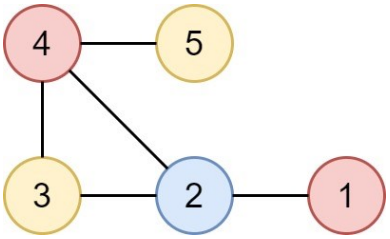
Vertices should always be numbered from 1 to n, where n is the number of vertices.

No other input format will be accepted!

Output format: The output should be a text file that consists of two lines:

- The first line should include a single integer value, the number of colors used (k).
- The second line should include the labels of the colors used for all vertices, from vertex 1 to vertex n. Label values are integers between zero and k-1 and they are separated by a space character.

A possible coloring for the above graph is as follows:



If the labels of the colors are assigned as: Red: 0; Blue: 1; Yellow: 2

Then the output would be as following:

3
0 1 2 0 2

No other output format will be accepted!

Example instances: You may find example instances in the google classroom. There are three example inputs (sample1.txt, sample2.txt, and sample3.txt). The optimal values are given in the following table:

Input File	Optimal Solution
Sample1.txt (87 vertices, 812 edges)	11
Sample2.txt (1000 vertices, 245000 edges)	50
Sample3.txt (500 vertices, 58862 edges)	122

Output files for optimal solutions are not available.

Test instances: On June 4th, we will announce 4 test instances. By June 5th, 23:59, you will be required to submit 4 separate output text files (with the correct output format) corresponding to each of these test instances. These files should be called output1.txt, output2.txt, output3.txt, output4.txt and should be submitted via google classroom. The deadline is strict.

Project report: The project report should describe the ideas behind your algorithms as completely as possible. **Also it should clearly indicate the division of labor among the group members.** It should not exceed 3 pages in length in no less than 10pt. Also please include a README file to briefly describe how to run your code. You should submit a zip file (with name **studentID1_studentID2_studentID3.zip**) including your project report, README file and commented source codes by June 7th, 2022, 23:59.

Grading policy:

60% of your grade will be determined by your project report. Clarity and creativity of your work will significantly affect your grade.

40% of your grade will be determined by your solutions to the test instances. Studies that find solutions closer to the best possible solution will get higher grades.

Best projects will be awarded with bonus points.

Important note: We will announce a verifier code for the graph coloring problem, which verifies whether an output file is a correct solution for a given input file. Your outputs should pass the verification, otherwise they will not be graded.