1 Problem Statement

The input instance of Graph Coloring is a undirected graph $G = \langle V, E \rangle$. The decision problem is to find an labeling function $d : V \rightarrow Z$ such that $\forall (u, v) \in E, d(u) \neq d(v)$. The optimization problem is to find a $d$ such that $|Z|$ is minimized. In English, the Graph Coloring Problem is to assign a color to each vertex such that every adjacent pair of vertices does not share the same color and the minimize the number of colors used.

The graphs we will be testing on are governed by two parameters $n$ and $p$, where $|V| = n$ and $p$ is the probability of having an edge between any two vertices.

2 Assignment

Write a algorithm to solve the Graph Coloring problem, you can apply any technique you prefer, including but not limited to, LS, CP, IP, LP, DP, brute force etc. Your algorithm should be able to perform within the following parameter values $n \in \{20, 50, 70, 100, 1000\}$ and $p \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. The assignment should be performed by teams of at most 2 people. We always expect

- both source files and binary programs, if any, of the working algorithm;
- a specification on how to run the program;
- a brief report in plain text containing, the names of each team member, a brief discussion of your solution strategy, implementation techniques and experimental observations. The report should be concise. An example report can be found here,

/course/cs258/data/README_example

3 I/O Specification

**Format**  The input consists of $|E| + 1$ lines. The first line contains two number $|V|$ and $|E|$. It follows by $|E|$ lines, each line represents an edge $u_i, v_i$ where $0 \leq u_i, v_i < |V|$.

```
[V] [E]
[Input Format]
[|V|] [|E|]
u_1 v_1
u_2 v_2
... u_]|E| v_]|E|
```
The output has two lines. The first line contains two numbers: the first number is the best objective value found by the algorithm; the second number is a flag, 1 if the algorithm is able to prove the optimality, otherwise 0. The second line is the coloring.

[Output Format]
|Z| opt_flag
d(0) d(1) ... d(|V|-1)

Example

[Sample Input]
5 7
0 1
0 4
1 2
1 3
2 3
2 4
3 4

If the algorithm proves the optimal value 3, the output is:

[Sample Output 1]
3 1
0 1 0 2 1

If the best objective value found by the algorithm is 3 but cannot prove optimality, the output is:

[Sample Output 2]
3 0
0 1 0 2 1

Instructions  We will run your submission using the command: ./gc <timelimit> <filename>

For example: ./gc 300 /course/cs258/data/coloring/gc_50_7
means the program will use gc_50_7 as input and will run at most 300 seconds.

We use stdout for output. Output to other stream will be ignored. Your submission will be tested on department’s linux machine. If your algorithm is a standalone program, please name it gc, otherwise, please specify the compilation procedure, it is appreciated if you also provide a script that follows the above format to run the program.

Resources  Some sample input files are available in: /course/cs258/data/coloring

Script to generate input instances: /course/cs258/bin/gen_gc <n> <p>

4 Remarks

Handin  Command: /course/cs258/bin/cs258_handin hw1

All of the files in the current directory and sub-directories will be submitted. Only the last submission will be marked.
Questions  Please contact the class GTA Carleton (cjc@cs.brown.edu).

Time Limit  You can assume that the time limit will be something between 60 and 300 seconds.

Hint  Optimal value for $p = 0.5$ and $n = 1000$ should be about 85.