Week 1 programming assignment

Task: It is necessary to develop and implement in C++ a greedy heuristics for solving the vertex coloring problem.

Input: text file in DIMACS format containing an undirected graph to be colored

Output: the number of colors *k* in the 1-st line and then *k* color classes.

Example:

Input file:

```
c c – comments, p – problem, edge – “edge” format, e - edge
c first line: p edge <number of vertices> <number of edges>
c next lines – graph edges: e <vertex 1> <vertex 2>
c WARNING: edges can be duplicated in these lines: (1,2), (2,1)
p edge 5 6
e 1 2
e 2 1
e 2 3
e 3 4
e 4 5
e 5 1
```

Output:

```
3
{1,3}, {2,4}, {5}
```

Here vertices {1, 3} are colored in color 1, vertices {2, 4} – in color 2, and vertex {5} – in color 3.

Input files to test the program should be taken from DIMACS challenge graphs:

https://mat.tepper.cmu.edu/COLOR/instances.html or

https://mat.gsia.cmu.edu/COLOR/instances.html or

https://hse.ru/data/2021/09/16/1471879280/vertex_coloring.7z

The program should be tested on the following files:

- myciel3.col (you can also use it for testing since it has only 11 vertices), myciel7.col – Mycielski graphs – triangle-free graphs with big chromatic numbers (minimum possible number of colors)
- latin_square_10.col – Latin square problem
- school1.col, school1_nsh.col – school timetable problems
- mulsol.i.1.col, inithx.i.1.col – CPU register allocation problems from real codes

- anna.col, huck.col, jean.col – Lev Tolstoy “Anna Karenina” graph with 138 vertices (characters), Mark Twain “Huckleberry Finn” graph with 74 characters, and Victor Hugo “Les Miserables” graph with 80 characters.

Computational results

Please compile a release version of your program with maximal optimizations turned on to obtain good computational results. Together with the program code it is necessary to provide the following table with your computational results including the computational time and the found solution:

Instance	Time, sec	Colors	Color classes
myciel3.col	0.000	4	{1, 2, 3, 4}, {5, 6, 7}, {8, 9}, {10, 11}
myciel7.col	0.001	8	{1, …
school1.col	1.234	100	{1, …
school1_nsh.col	0.123	50	{1, …
anna.col	0.456	40	{1, …
miles1000.col	0.345	55	{1, …
miles1500.col	1.002	60	{1, …
le450_5a.col	0.004	13	{1, …
le450_15b.col	0.003	12	{1, …
queen11_11.col	0.002	11	{1, …

Operating systems: it is possible to provide a C++ program code for Windows, Linux or MacOs. Other operating systems and programming languages are not allowed.

Evaluation:

First evaluation:

1. Maximum number of points is 10.
2. Incorrectly working programs get no more than 3 points. Incorrect programs are those which do not pass Check() test or can fail with “core dumped” or other errors shown by an operating system. Programs with hard-coded solutions are also considered incorrect.
3. A program should construct only one coloring.

Two things are the most important:

1. A quality of solutions: the smaller is the number of colors in your coloring – the better
2. An efficiency of your program: the faster is your program the better

Here are the computational times (for release version – with O3 optimizations) and numbers of colors, which get the maximal mark of 10 points:

Instance	Time, sec	Colors
myciel3.col	≤ 0.01	≤ 4
myciel7.col	≤ 0.08	≤ 8
school1.col	≤ 0.40	≤ 15
school1_nsh.col	≤ 0.30	≤ 21
anna.col	≤ 0.04	≤ 11
miles1000.col	≤ 0.04	≤ 42
miles1500.col	≤ 0.25	≤ 73
le450_5a.col	≤ 0.20	≤ 11
le450_15b.col	≤ 0.20	≤ 18
queen11_11.col	≤ 0.15	≤ 16

Here are approximate results, which get 6 points:

Instance	Time, sec	Colors
myciel3.col	~ 0.05	~ 4
myciel7.col	~ 0.40	~ 8
school1.col	~ 2.00	~ 42
school1_nsh.col	~ 1.50	~ 39
anna.col	~ 0.20	~ 12
miles1000.col	~ 0.20	~ 44
miles1500.col	~ 1.25	~ 76
le450_5a.col	~ 1.00	~ 14
le450_15b.col	~ 1.00	~ 22
queen11_11.col	~ 0.75	~ 17

Correct but slower programs or programs returning worse colorings get 4-5 points. Better and faster programs get 7-10 points.