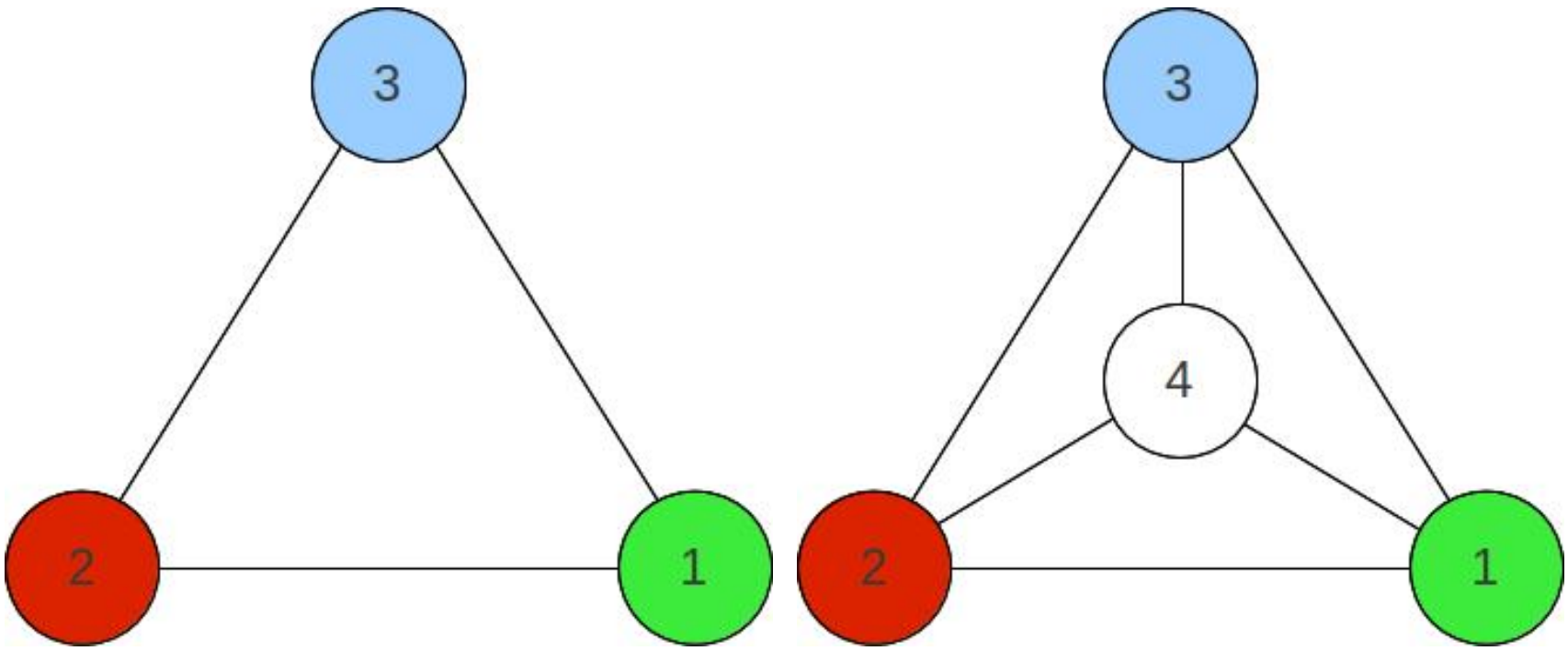
The "Fundamentals of Computing" Specialization Learn More ×

Graph coloring

Help Center

Graph coloring is the process of assigning "colors" (or any other labels, really) to nodes in graph in such a way that no two neighboring nodes have the same color. You can read more about graph coloring here. As an example, consider the graphs below:



The graph on left can be colored with three colors while the graph on right requires four colors.

Graph coloring can be used to solve many problems. Perhaps the easiest to visualize is coloring a map. The map can be represented as a graph where nodes are states (or countries) and edges indicate that the states (countries) are adjacent to each other. A successful coloring would then be one that ensure that no adjacent states (countries) have the same color, which enables you to easily visually distinguish boundaries on the map.

Graph coloring can also be used to solve scheduling problems (such as how a shipping company should schedule trucks to use its loading docks), register allocation in compilers, solving Sudoku puzzles, and many others.

For this activity, we will consider the 3 coloring problem. You will have three tasks:

- Develop pseudo-code for a brute-force algorithm to 3 color a graph. Once you are happy with your solution, you are welcome to examine our pseudo-code.
- Implement your pseudo-code in Python and test it on the following data:

```
GRAPH1 = {0: set([1, 2]),
      1: set([0, 2]),
      2: set([0, 1])}

GRAPH2 = {0: set([1, 2, 3]),
      1: set([0, 2, 3]),
      2: set([0, 1, 3]),
      3: set([0, 1, 2])}
```

```
GRAPH3 = {0: set([1, 2, 4, 5]),
      1: set([0, 2, 3, 5]),
      2: set([0, 1, 3, 4]),
      3: set([1, 2, 4, 5]),
      4: set([0, 2, 3, 5]),
      5: set([0, 1, 3, 4])}

GRAPH4 = {1: set([2, 8]),
      2: set([1, 3, 4, 6, 8]),
      3: set([2, 4]),
      4: set([2, 3, 5, 6, 8]),
      5: set([4, 6]),
      6: set([2, 4, 5, 7, 8]),
      7: set([6, 8]),
      8: set([1, 2, 4, 6, 7])}
```

Remember that this is a decision problem in which we must decide whether the nodes of a particular graph can be colored with only three colors (call them red, green, and blue) in such a way that no adjacent nodes have the same color.

As hint, `GRAPH1` is 3 colorable. Remember that a link our Python code is available at the top of the page.

Created Sun 24 Aug 2014 12:41 AM BST

Last Modified Sun 24 Aug 2014 6:00 AM BST