# PROJECT 8 GRAPH ALGORITHM AND BREADTH FIRST SEARCH TREE

In [graph theory](http://en.wikipedia.org/wiki/Graph_theory), breadth-first search (BFS) is a [strategy for searching in a graph](http://en.wikipedia.org/wiki/Graph_search_algorithm) when search is limited to essentially two operations:

1. Visit and inspect a node (vertex) of a graph;
2. Gain access to visit the nodes that neighbor to the currently visited node.

The BFS begins at a root node and inspects all the neighboring nodes. Then for each of those neighbor nodes in turn, it inspects their neighbor nodes which were unvisited, and so on.

1. Using the graph algorithm discussed in the class, implement a program in any language you desire to do Map Coloring Problem for the attached map of South America:
2. The neighboring countries must use different colors.
3. Convert the map coloring problem to a graph diagram from the attached map of South America.

(**Draw the graph diagram.** You may use pen/pencil to draw this in a very nice drawing if utility software is not available.)

A picture containing person

Description automatically generated

1. **Display** the adjacency list representation of the graph from (a.2)

A picture containing air, table, man, board

Description automatically generated

A close up of text on a black background

Description automatically generated

A screenshot of a cell phone

Description automatically generated

1. **Display** the color used for each country and the colors for all countries.

(Find an algorithm BFS tree to use the minimum number of colors.)

A screenshot of text

Description automatically generated

1. The colors used should be in the following order:

{Blue, Brown, Green, Lavender, Orange, Pink, Red, Yellow, Violet, Gold,

Gray, Indigo, Silver}

**private** String[] **colors** = **new** String[]{**"Blue"**, **"Brown"**, **"Green"**, **"Lavander"**, **"Orange"**, **"Pink"**, **"Red"**, **"Yellow"**, **"Violet"**, **"Gold"**, **"Gray"**, **"Indigo"**, **"Silver"**};



(6) In this project, you also need to **create a BFS tree** for future search purpose.

Use BFS algorithm applying to (a.4) to create a BFS tree and then

**display the tree.**

**See the source code**

1. Do whatever you want if not specified in the project.