Criterion E - Evaluation

= Evaluation of Solution =

- Success Criteria -

- color any map with 4 colors
  - Flexible input of map borders enables any map to be described relatively easily. The random search produced solutions for all tested maps.

- easily enter border data
  - The use of a "map-diagram" minimizes input work. It took 3 minutes to type data for the US FLAG image. In a beta test (typing Canada), the client had a few questions but agreed it was easy after brief practice. [Additional information regarding testing is very useful. In this case, the benefit of the additional text outweighs the fact that this text is used in the word count, especially as total word count for the solution is approximately 1400]

- automatically find a set of 4-colors, neighbouring regions having different colors
  - worked correctly in all test cases.

- use fewer than 4 colors if possible
  - worked correctly in all test cases, although requiring many tries with the US FLAG.

- print solution as a list of regions and corresponding colors
  - worked correctly in all cases.

This would work better if the success criteria is replicated from criterion A verbatim. Perhaps a table format would allow for easier moderation.

- Effectiveness -

The Canada test usually found a 4-color solution in under 1000 tries. It required several thousand tries to find a 3-color solution. The total running time was under 1 minute. The US FLAG required more iterations, but still finished in under 1 minute.

The most time consuming part is typing the border data. For a map under 15 regions, this takes about 5 minutes. It is unclear how long it would take for a larger map, but hopefully a large map like the USA could be done in less than an hour.

The client had a bit of trouble typing the data for Canada, but succeeded after a couple tries. He did not like the Python interface at first (wasn't pretty enough), but after a couple uses agreed it was adequate. In the end, the client agreed that the product was usable and successful.
[Very clear evidence of feedback obtained from the client]

= Recommendations for Further Development =

- Minor Improvements -

- **More Iterations -** For larger maps, it's useful to increase the iterations from 1000 to 5,000 or more, by inputting this value at program start.
- **Sorted Output -** The color list prints in an unpredictable order. It should be either alphabetical or follow the same order as the input file. This can be solved be using the **state** array to control the printing order and/or adding a sorting method.

- Major Improvements -

- **Data Validation -** The user might type incorrect border data in the text-file, e.g.:
  - misspelling the ID of region, like "NV" instead of "NW"
  - accidentally typing too many neighbors or too few

  Validation checks could include:
  - check that all neighbor entries match one on the region names (first entires)
  - check that any neighbor pairs appear twice, e.g. YU --> BC and also BC --> YU.

  This would detect many errors, preventing pointlessly unsuccessful retries.
- **Nicer Interface -** The program could be rewritten in JavaScript and then run in a web-page. This would provide a "nicer" interface, as well as eliminating the need to install Python.

* **Words = 500 ****

This criterion has been awarded 6 marks.