Activity 1. Answer the following questions

* Briefly explain how your algorithm works

The first that my algorithm does is start iterating through a List if ‘Nodes’ that represent each of the countries on the map, it does this with a while loop, and right after that it initializes the array named ‘occupiedColors’ to a new array with the length of the array that contains all the colors, this array is used to store the colors of the adjacent countries of the actually selected country, in this way we can easily select the remaining color to ‘paint’ our country with,.

Then, for each of the adjacent countries of the selected one, we check if the adjacent has been already painted, if it has, we store its color and then we choose the unoccupied color from the array to paint the country with.

* How many colors did you need to use to solve the problem?

In order to solve the problem, only 5 colors were needed (Red, Blue, Green, Yellow and Black).

* May the number of colors change if you use a different order for the countries to be processed by your algorithm?

The number of colors for solving the problem could change if, for example, we ordered the countries in such a way in which the country that has more adjacent countries would be the last one to be painted, given that all of its adjacent have colors different from the others, then perhaps the algorithm would need to use another color.

* How many colors would be used in an optimal solution?

The optimal solution for this problem would have to use as much colors as the number of adjacent countries of the country with the biggest number or adjacent countries plus one (for itself).

* What is the time complexity of your algorithm? Briefly explain it.

First it has a while loop with complexity O(countries.size()), and after that it has 2 loops inside the first while loop whose complexity are O(countries.get(i).getAdjacents().size()), (so it depends on the number of adjacent countries of the selected one) and another one which is always O(colors.length).

Then the total complexity would be:

Countries.size() \* ( Colors.length + Countries(i).getAdjacents().size() )