Main idea:

We use aggregation algorithm to the people in Eindhoven. We group them into five groups, West EU, South EU, North EU, East EU and NON-EU, each of them is represented with a specific color, West EU = Green, South EU = Blue, North EU = Yellow, East EU = Red and NON-EU = Black.

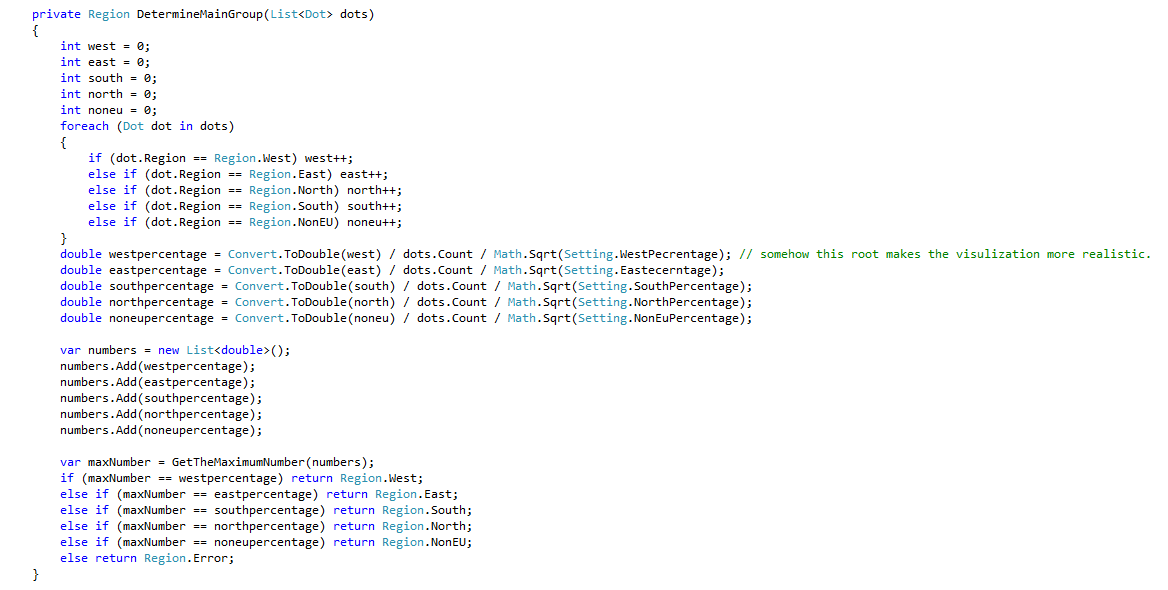
The data used in this program is generated by itself, not actual data.

The data is generated in the following way: It can be basically divided to 2 steps:

1. Firstly, user can define how many people are there in total maximum, start the system and it will calculate all the possible positions where the people can be, because we want that most people are mostly in some specific areas and some areas like forest would contains few people only, hence, the algorithm which is used to generate these positions is currently slow. It might take hours to generate these data if the user want to have millions of people in the city. Note that there is already a file called “data.txt” which contains the data of 1 million people. Note that there will actually be much less than 1 million people, because 1 million means 1,000 \* 1,000 pixels for the whole window, but the map is with certain shape, it does not cover the whole window, hence, it is actually less. Moreover, the map contains some inner polygons (city blocks) which mainly contains many people, the rest of the city are considered as forest or river or somewhere which contains few people. Hence, the total number of person in the actual system would usually be much less than 1 million.
2. Secondly, the system reads the “data.txt” (it will create one based on the user specification if there is no such file) and generate people. The data only contains possible positions, for each position the system will determine what kind of person should be there based on system setting (user can specific the percentage of people from different regions). Moreover, not every position loaded needs to be a person, it is also possible that, at that position, there is nobody.

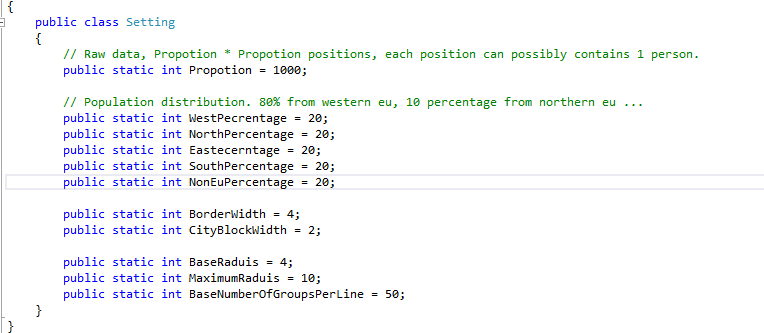
The aggregation mainly works as following:

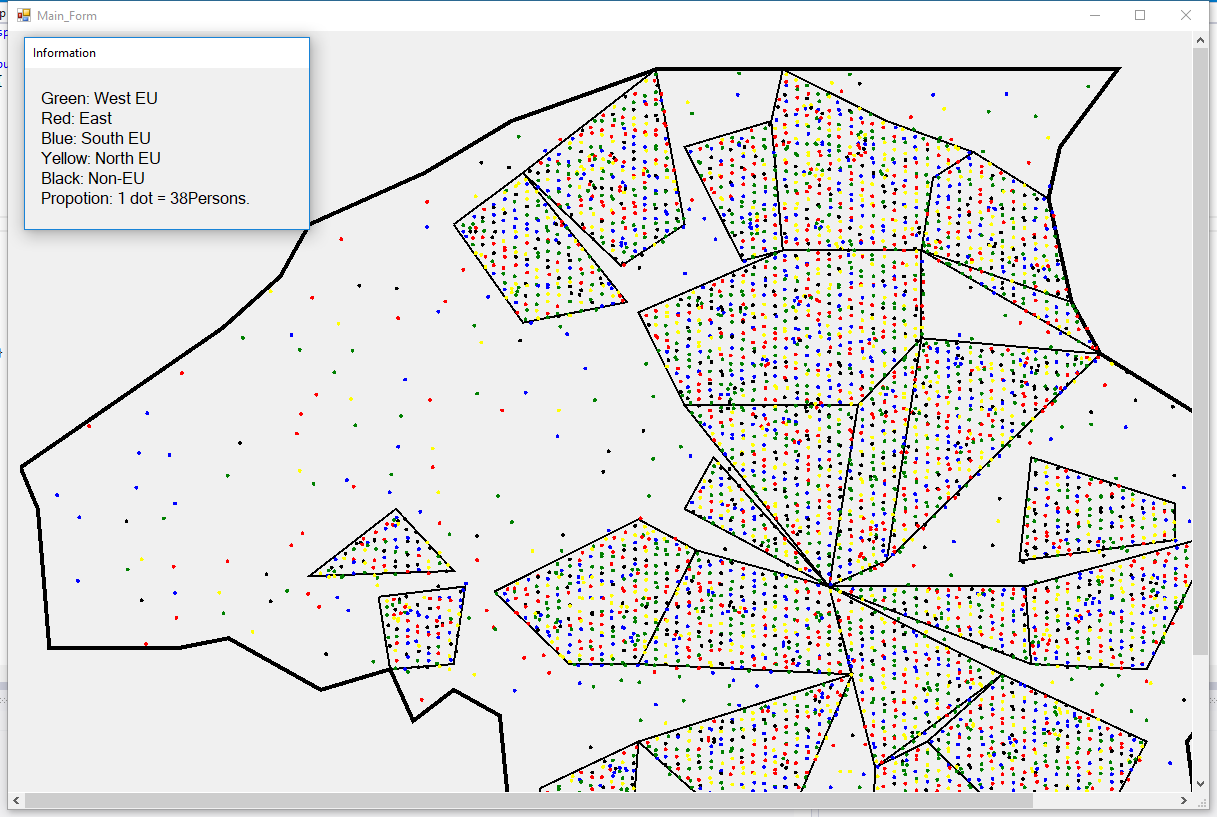
1. User can specific a number, say how much points I want to see in the screen with the initial window size. For example, if the window size is 1000\*600, user may specify that I want to see 50 dots (maximum) for each row and column (each dot can represent certain number of people).
2. We calculate how many points together should be aggregated to 1 dot. Call it “NumberOfPeopleInOneGroup”.
3. It divide the whole map to for parts, left up, right up, left button, right button, it keeps dividing recursively, until the number of dots in the square is less than “NumberOfPeopleInOneGroup”.
4. We aggregate that group of people with the following rule:



Constraints:

Because of the fact that aggregation will lose some details of the original information. So if the percentage of each group are evenly divided, then the information displayed maybe not so correct. For instance, there are in total 100,000 people, 20,000 of them are from West EU, 20,000 of them are from South EU, 20,000 of them are from North EU, 20,000 of them are from East EU and 20,000 of them are from Non-EU, in this case, the visualized result might not be very accurate..





Moreover, the aggregation algorithm runs in O(n) times, where n = number of total people. And for each time, when the “NumberOfPeopleInOneGroup” changes, the aggregation algorithm will be reapplied, hence, when the number of total people becomes very large, the algorithm might not be so efficient.