Things to implement

- A GUI that provides a canvas(a clickable area) where the user can click to keep a dot on the canvas.
- A Load option to load a pre-existing file into the application.
- A save option to save the existing file to the system. – Only **dots** are required to be saved, not the lines that connect the dots.
- A Random generator that randomly places the dots on the canvas. The number of dots can be fixed by you (the Developer) or it can be left to the user (provide a text input in this case).
- A clear screen button that clears the screen.
- A Run button that runs a brute-force version of the DBSCAN clustering algorithm recursively on all the dots on the canvas. After clicking this button, the user should be provided a way to enter the **distance parameter** as an integer value.

What does the algorithm do?

- Randomly pick a dot from the existing dots on the canvas.
- Using the distance parameter that's taken from the user, **connect** (draw a line) the picked dot with all the dots that are at a distance less than or equal to the value of distance parameter.
- All the dots that are at a distance more than the distance parameter will not be connected this way. Mark the current dot and also the dots **connected** to it as "**visited**".
- Repeat steps 2 and 3 by picking every dot that is connected to the current dot. (While doing this, if you come across a visited dot, skip it).
- Repeat the above steps till all the dots that represent the tree rooted at the first picked dot are visited.

Repeat the entire algorithm by picking another random unvisited dot on the canvas.

Note:

- Distance is computed as the euclidean distance between two points.
- Make sure you add a delay after a tree has been formed and before moving onto picking the next random dot. (Could do a sleep or a poll if multithreading is being used).This will show that the trees are actually being formed.
- At last, you will be left with several trees on the canvas. (Any dot whose distance is more than the distance parameter from any other dot stays alone).

Things to submit:

- Java source code files.
- A Jar file that contains that archives the entire application.
- A document that states the personal contribution of each team member in the project. Every team member is expected to contribute to the code. Statements like “This person handled the documentation” reflects a poor division of work. Please divide the coding tasks accordingly. You might be asked questions related to your personal contribution in this assignment on the final exam. You might lose points in the final exam if you do not concentrate enough on this assignment.

Things to remember:

- All work should be done using the Java programming language (Java 8 would be enough).
- No external library apart from **java swing** is allowed.
- ***Do not use*** the form provided by intellij or any other GUI building tool provided by any IDE. GUI should be coded by you. **This will be strictly checked.**
- Java Doc comments are to be kept wherever necessary.
- Please follow coding conventions for the Java programming language.
- Make sure your code has No leaky IO calls, No swallowing of exceptions and no unattended catch statements.

All the best :)
