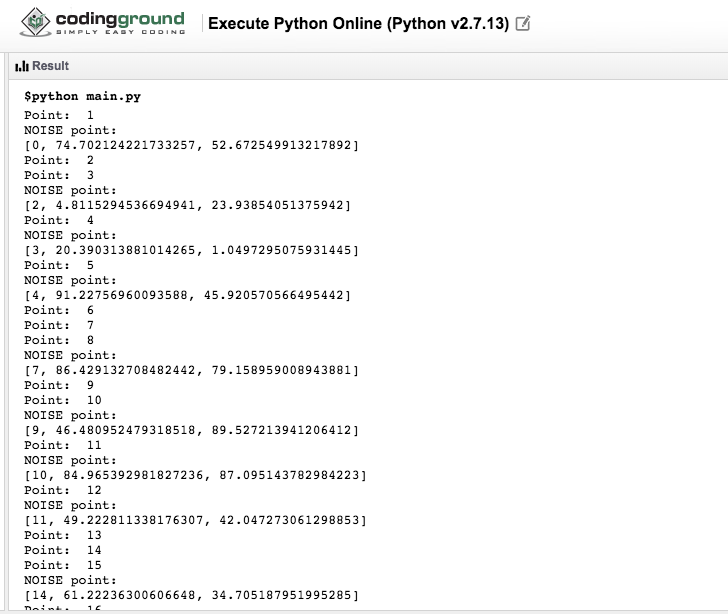


My first python program!!!!!! I am excited to finally get this working. I chose eps and min points to give me some clusters that are not too dense, because I am graphing by hand. I have attached graphs of all three cases, input, and output. Source code is attached above. If you would like me to send actual code so you can run it let me know and I will email it. I was iterating over my random points wrong initially, then had some problems passing values in Python for my first time. I put print statements at various spots to make sure the values I thought were being passed were actually getting passed and followed what was happening then made changes. I output only clusters with points greater than MinPts because for some reason I am selecting each point of the cluster as an individual cluster along with the actual cluster it belongs to. I am pretty sure I can tell where to put an output statement to identify branch points would lines 47 and 48 “if len(neighbourPts\_2) >= MinPts: neighbourPts +=neighbourPts\_2”, but I did not put an output statement there because not required in program requirements. I output all noise points, then “Number of clustered points:” and “Number of noise points:” statements followed by the actual clusters and their points. Noise points are NOT included on the graph. I ran and compiled in online ide so no graphing functions. I graphed by hand on excel. I ran separately for each case, and hardcoded eps and MinPts. I can come into your office and demonstrate code, come to class 15 minutes on day of final and explain my code to class, or anything else that needs to be done. The graphing is unnecessary to learning about the algorithm so skipped for case b and case c.

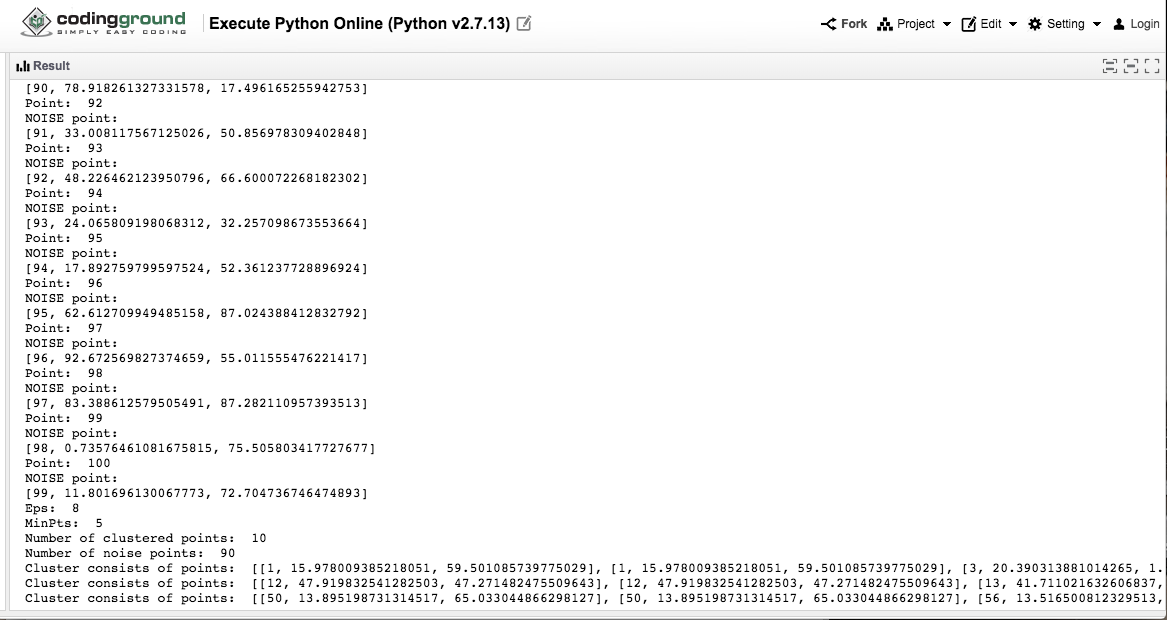
EDIT: I added print ("Eps: ") , (eps) above line 33. Outputting variables.

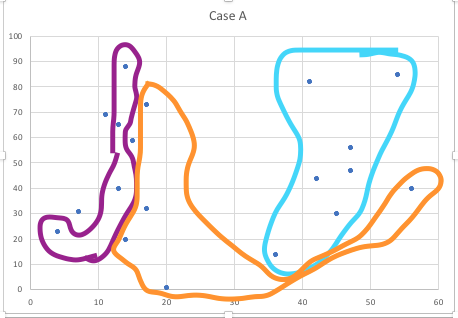
print ("MinPts: ") , (MinPts)

Case A:

Eps: 8

MinPts: 5



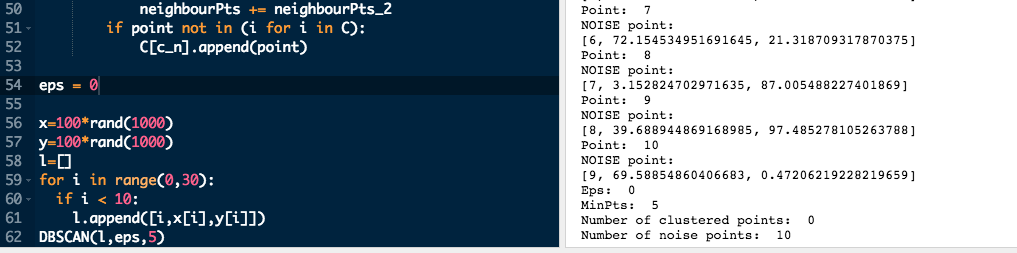


Case B:

Eps: 0

MinPts: 5

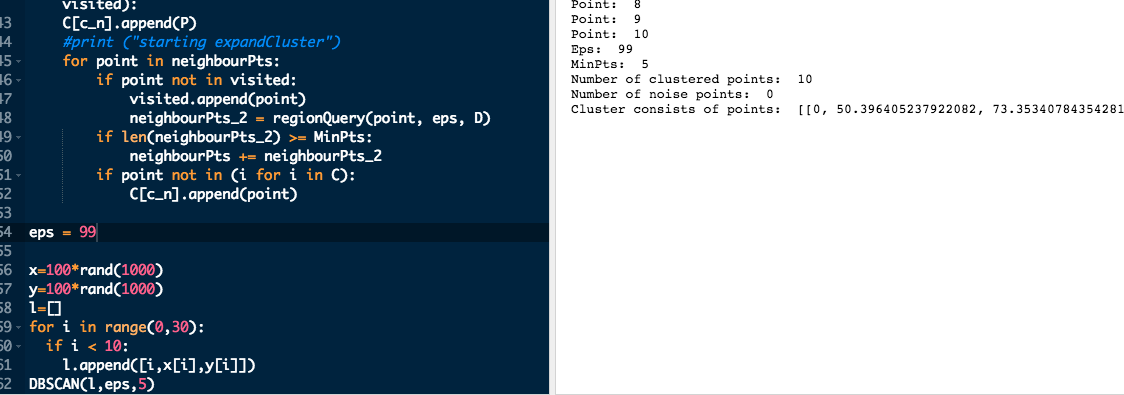
This will show that with eps = 0 gives all points being their own cluster because the distance measure = 0.



Case C:

Eps: 99

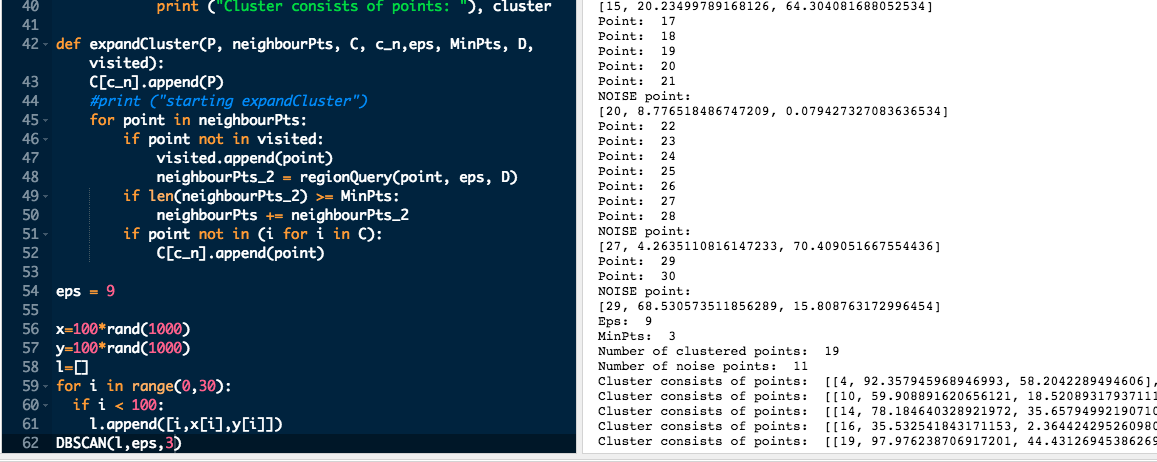
MinPts:2

This will give all the points being in 1 cluster bc the distance measure between two points in the same cluster can be as high as 99 (covers whole area)

Case D:

Eps: 9

MinPts: 3

This configuration shows 19 points in 5 clusters. I included this to show how my program outputs different clusters.