As a delivery company, one of the biggest challenges is to keep the routes for the drivers as tightly clustered as possible and making sure that each driver ends up finishing their routes in their preferred regions. Which is why it is important to break the service area (GTA) into sub-regions or zones (North York, Markham, Vaughan etc).

Now, we don't just want to break the entire service area into zones just based on postal codes that are in that city...although we could…and it would be much easier to do! But that would be too easy!

The real reason is that we don't get uniform density of packages in each city for it to be worthwhile for the drivers. For example based on the number of delivery orders received each day, it might be better to combine areas such as Oakville and Milton just because of the population there. This results in more balanced routes for the drivers and makes it worthwhile for the drivers to deliver enough packages so that they can make enough money that day.

Another reason is that if we optimize routes in one shot, it takes exponentially longer to reach a solution as the number of stops increases (Vehicle Routing Problem being an NP-Hard problem). So we want to break the large optimization problem for the large service area into multiple smaller ones for each zone.

Now the challenge is to write a program that partitions the given number of delivery orders into clusters that minimizes the within cluster variance (distance). These partitions would dictate the zones for a given dataset of delivery orders on a given day. You can pick any number of partitions to break the region into. Any combination of regions described below would work although you don't necessarily have to use the image below!

Notes:

- Using Euclidean distance would be good enough. Although in the real world, the distances between the two stops are usually asymmetric based on our road networks.
- Use any language you want with any library
- Just send the code, the solution and any supporting document explaining the decisions you made while solving this problem in a zipped folder to jamal@envoinow.com and sergey@envoinow.com
- Hint: This is an NP-Hard problem!
- Also find attached dataset of addresses and their corresponding coordinates