In the first task I created a function that takes a dataset and an integer value, *k*, which is the number of clusters, as the parameters. In addition there is a third parameter, *g*, which is optional just in case the user wants to display a graph of the centers changing with each iteration. The function chooses *k* random center points in the range of the given dataset, calculates the distances from the centers to the points in the dataset, finds the center which has the lowest distance to the data point and assigns it to that specific cluster. Then a mean of the data points in each cluster is calculated and the new center is placed at that value. This is repeated until the centers do not change position. If the user decides to output the graph, a little movie is created and saved in the source folder. The function returns the array of which cluster the data points belong to and the centers that have been calculated.

For the second task the function that was created the parameters that it takes is an array that shows which points belong to which cluster; the dataset; the test data; and the number of nearest neighbours, *n*. The function first outputs a graph of the dataset with the clusters; finds the distance from each test point to each data point; it then sorts the distance array and finds the *n* number of nearest points to the test point; after that a majority vote is performed on the identification of the cluster type and the new test point is allocated to that cluster. The function then outputs the array of the cluster types the test points belong to,