Code explanations are added in the comments of the code .
Q1.
RowVectorFloat class was created . _str_ method was added so that class can be printed as we want it . len , add , multiplication method was also added .
Set and get methods were also added.

Q2.
SquareMatrixFloat class was created which contains methods like -
sample symmetric - which will create a random symmetric matrix using radom function on each point of matrix
toRowechelonForm – converts a matrix into row echelon form.
isDRDominant - which checks if a matrix is row dominant or not .
Jsolve - which applies jacobi method for a given B,and m iterations,also simultaneously calculates deviation .
gsSolve - applies gauss seidel method ,also simultaneously calculates deviation .

Since sampling a diagonally row dominant matrix directly is difficult , so i do sampling until i find it .
Q3.
This question involved all classes of previous question just a extra plotting method was needed to be made such that , the err values we return for each iteration of each convergence method can be plotted.
Q4.
A polynomial class was created , _str_ method was added in case someone does print on the class .methods , to add , subtract two polynomial were added . also to multiply a constant to polynomial a method was added for such action .

The get method was added in class that returns value of polynomial at a particular point.
.show method will plot the values of the polynomial in a range.

Fitvia matrix method was added which solves the polynomial , given some points using linear equations , i used numpy.linalg module for that .it also plots the value of this polynomial in a certain range .

fitvialagrangePoly method was added which uses points given to compute the lagrange polynomial , also plots the points in the max,min range of points .

Q5.
As shown in the link we had to do 3 interpolations: Akima, barycentric and cubicsplinter.
And compare them to original values after each iteration we had to increase the number of points from the range .
Library’s i used were scipy.interpolate, matplotlib.animation’s funcanimation.