ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

2η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ ΣΤΟ

ΜΑΘΗΜΑ:

ΥΠΟΛΟΓΙΣΤΙΚΗ ΝΟΗΜΟΣΥΝΗ

ΦΟΙΤΗΤΕΣ:

Σκούρα Κωνσταντίνα ΑΜ :4168
Μπουραζάνης Νικόλαος ΑΜ:2311
Ζήγος Αυγουστίνος ΑΜ : 2973

ΠΕΡΙΕΧΟΜΕΝΑ:

Άσκηση 2

Στον υποφάκελο «createExamples» έχουμε το αρχείο Main.java ,με το οποίο δημιουργούμε το αρχείο examples.txt που περιέχει τα 1200 παραδείγματα. Έχουμε προσθέσει και ένα ενδεικτικό αρχείο με παραδείγματα.

Πιο συγκεκριμένα, για να δημιουργηθούν 150 τυχαία σημεία στο τετράγωνο [0.8,1.2]x[0.8,1.2] έχουμε ορίσει ένα άνω όριο upperbound= 119 και ένα κάτω όριο lowerbound = 80 για το x1 και x2 αντίστοιχα και κάθε φορά υπολογίζεται ένα τυχαίο ζεύγος ανάμεσα στο 80 και στο 120 ,το οποίο διαιρούμε με 100 ώστε να είναι στο επιθυμητό εύρος τιμών. Έπειτα γράφουμε το x1,x2 στο αρχείο μας και αλλάζουμε γραμμή.

```java
public class Main {
    public static void main(String[] args) {

        float x1= 0.0F;
        float x2= 0.0F;
        int upperbound1 = 0;
        int lowerbound1 = 0;
        int upperbound2 = 0;
        int lowerbound2 = 0;


        try {
            FileWriter myObj = new FileWriter("examples.txt");

            Random rand = new Random();
            upperbound1 = 119;
            lowerbound1 = 80;
            upperbound2 = 119;
            lowerbound2 = 80;

            for (int i=0;i<150;i++) {
                x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
                x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

                String formattedFloat1 = String.format("%.2f", x1);
                String formattedFloat2 = String.format("%.2f", x2);
                myObj.write(formattedFloat1);
                myObj.write("\t");
                myObj.write(formattedFloat2);
                myObj.write("\n");
            }
```

Ίδια διαδικασία ακολουθείται και για τα 150 σημεία στο τετράγωνο [0,0.5]x[0,0.5],με αλλαγή των ορίων upperbound = 49 και lowerbound = 0

```
upperbound1 = 49;
lowerbound1 = 0;
upperbound2 = 49;
lowerbound2 = 0;

for (int i=0;i<150;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 150 σημεία στο τετράγωνο [1.5,2]x[0,0.5] όπου τώρα το x1 θα έχει lowebound= 150 και upperbound = 199 ενώ το x2 θα έχει lowerbound = 0 και upperbound = 49

```
upperbound1 = 199;
lowerbound1 = 150;
upperbound2 = 49;
lowerbound2 = 0;
for (int i=0;i<150;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 150 σημεία στο τετράγωνο [0,0.5]x[1.5,2] όπου τώρα το x1 θα έχει lowebound= 0 και upperbound = 49 ενώ το x2 θα έχει lowerbound = 150 και upperbound = 199

```
upperbound1 = 49;
lowerbound1 = 0;
upperbound2 = 199;
lowerbound2 = 150;

for (int i=0;i<150;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 150 σημεία στο τετράγωνο [1.5,2] x [1.5,2], το x1 και το x2 θα έχουν lowebound= 150 και upperbound = 199

```
upperbound1 = 199;
lowerbound1 = 150;
upperbound2 = 199;
lowerbound2 = 150;

for (int i=0;i<150;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 75 σημεία στο τετράγωνο [0,0.4]x[0.8, 1.2] όπου τώρα το x1 θα έχει lowebound= 0 και upperbound = 39 ενώ το x2 θα έχει lowerbound = 80 και upperbound = 119

```
upperbound1 = 39;
lowerbound1 = 0;
upperbound2 = 119;
lowerbound2 = 80;

for (int i=0;i<75;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 75 σημεία στο τετράγωνο [1.6,2]x[0.8, 1.2] όπου τώρα το x1 θα έχει lowebound= 160 και upperbound = 199 ενώ το x2 θα έχει lowerbound = 80 και upperbound = 119

```
upperbound1 = 199;
lowerbound1 = 160;
upperbound2 = 119;
lowerbound2 = 80;

for (int i=0;i<75;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 75 σημεία στο τετράγωνο [0.8,1.2]x[0.3,0.7], όπου τωρα το x1 θα έχει lowebound= 80 και upperbound = 119 ενώ το x2 θα έχει lowerbound = 30 και upperbound = 69

```
upperbound1 = 119;
lowerbound1 = 80;
upperbound2 = 69;
lowerbound2 = 30;

for (int i=0;i<75;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Για τα 75 σημεία στο τετράγωνο [0.8,1.2]x[1.3,1.7],όπου το x1 θα έχει lowebound= 80 και upperbound = 119 ενώ το x2 θα έχει lowerbound = 130 και upperbound = 169

```
upperbound1 = 119;
lowerbound1 = 80;
upperbound2 = 169;
lowerbound2 = 130;

for (int i=0;i<75;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Τέλος για τα 150 σημεία στο τετράγωνο [0,2]x[0,2],το x1 και το x2 θα έχουν lowebound= 0 και upperbound = 119 .

```
upperbound1 = 119;
lowerbound1 = 0;
upperbound2 = 119;
lowerbound2 = 0;

for (int i=0;i<150;i++) {
    x1 = (rand.nextFloat() * (upperbound1 - lowerbound1 + 1) + lowerbound1)/100;
    x2 = (rand.nextFloat() * (upperbound2 - lowerbound2 + 1) + lowerbound2)/100;

    String formattedFloat1 = String.format("%.2f", x1);
    String formattedFloat2 = String.format("%.2f", x2);
    myObj.write(formattedFloat1);
    myObj.write("\t");
    myObj.write(formattedFloat2);
    myObj.write("\n");
}
```

Στον ergasia2 έχουμε τον κώδικα για τον αλγόριθμο μας. Αρχικά στο αρχείο Point.java φτιάχνουμε μια κλάση Point ,όπου έχει πεδία τα x1,x2 και την ομάδα του κάθε σημείου.

```
public class Point {
    float x1;
    float x2;
    int previousTeam;
    Point(float x1, float x2) {
        this.x1 = x1;
        this.x2 = x2;
    }
}
```

Στη συνέχεια στο αρχείο Load.java ορίζουμε έναν πίνακα με Points,έναν πίνακα με τα κέντρα(Centers) ,τον αριθμό των points και το M δηλαδή σε πόσες ομάδες είναι χωρισμένα. Έχουμε μια μέθοδο readfile,με την οποία διαβάζουμε το αρχείο examples.txt χρησιμοποιώντας την κλάση Scanner. Αρχικά διαβάζουμε γραμμή-γραμμή και αποθηκεύουμε στον πίνακα currentpoint τα x1 και x2 ,δημιουργούμε νέα αντικείμενα τύπου Point και μετά τα αντιγράφουμε στον πίνακα points.

```
public class Load {
    private Point[] points;
    private Point[] centers;
    private int numberofPoints;
    private int M;
    public Load(Point[] points, Point[] centers, int numberofPoints, int M) {
        this.points = points;
        this.centers = centers;
        this.numberofPoints = numberofPoints;
        this.M = M;
    }

    public void readFile(String filename) {
        Scanner reader = null;
        int pointIndex = 0;

        try {
            reader = new Scanner(new FileInputStream(filename));
        } catch (FileNotFoundException e) {
            System.out.println("Could not find file, please verify the path location!!");
            System.exit( status: 0);
        }
        while (reader.hasNextLine()) {
            String line = reader.nextLine();
            String currentPoint[] = line.split( regex: "\t");
            Point point = new Point(Float.parseFloat(currentPoint[0]), Float.parseFloat(currentPoint[1]));
            points[pointIndex++] = point;
        }
        reader.close();
    }
```

Η άλλη μέθοδο που έχουμε στο αρχείο Load.java είναι η findRandomCenters ,η οποία βρίσκει τυχαία αρχικά κέντρα .Χρησιμοποιούμε τη κλάση Random και έναν ακέραιο τον randomIndex.Για κάθε ομάδα βρίσκουμε ένα randomIndex ανάλογα τον αριθμό των points και παίρνουμε σαν κέντρο ένα τυχαίο point από τον πίνακα των points βάση αυτού του randomIndex.Προσθέτουμε το κέντρο που πήραμε στον πίνακα centers και αρχικοποιούμε την ομάδα του αντίστοιχου κέντρου.

```
    public void findRandomCenters() {
        Random r = new Random();
        Point center;
        int randomIndex;
        for (int i = 0; i < M; i++) {
            randomIndex = r.nextInt( bound: numberofPoints - 1);
            //random point
            center = points[randomIndex];
            centers[i] = center;
            centers[i].previousTeam = i + 1; //Arxikopoiw to previousTeam gia ola ta kentra
        }
    }
}
```

Στο αρχείο KMeans.java έχουμε τη μέθοδο findNewCenters με την οποία βρίσκουμε τα νέα κέντρα ,υπολογίζοντας τον μέσο ορό των x1 και x2 από όλα τα points.Αρχικά έχουμε έναν counter =0 με τον οποίο μετράμε ποσά points είναι σε κάθε ομάδα ώστε όταν βρούμε το άθροισμα των x1(sumX1) και το άθροισμα των x2(sumX2) όλων των σημείων στη αντίστοιχη ομάδα να τα διαιρέσουμε με αυτόν για να υπολογίσουμε τις συντεταγμένες των νέων κέντρων.

```
public class KMeans {
    private Point[] points;
    private Point[] centers;
    private int numberofPoints;
    private int M;
    private int flag;
    public KMeans(Point[] points, Point[] centers, int numberofPoints, int flag, int M) {
        this.points = points;
        this.centers = centers;
        this.numberofPoints = numberofPoints;
        this.flag = flag;
        this.M = M;
    }

    //Ypologizw ta nea kentra me meso oro twn syntetagmenwn apo ola ta points:
    public void findNewCenters() {
        for (int i = 0; i < M; i++) {
            int counter = 0;
            float sumX1 = 0, sumX2 = 0;
            for (Point point : points) {
                if(point.previousTeam == i + 1){
                    sumX1 += point.x1;
                    sumX2 += point.x2;
                    counter++;
                }
            }
            centers[i].x1 = sumX1 / counter;
            centers[i].x2 = sumX2 / counter;
        }
    }
```

Με τη μέθοδο assignLabels διατρέχουμε τα points μας και τα κέντρα και καλούμε τη μέθοδο euclideanDistance που επιστρέφει την ευκλείδεια απόσταση μεταξύ τους. Έχοντας ορίσει μια τιμή αρχικά min =100 αν η απόσταση μεταξύ κέντρου και σημείου είναι μικρότερη του min τότε το min παίρνει ως τιμή τη μικρότερη απόσταση από το αντίστοιχο κέντρο και αυτό αποθηκεύεται στο minCenter .Όταν θα έχουμε διατρέξει όλα τα κέντρα για ένα σημείο θέτουμε την ομάδα του σημείου .Η ίδια διαδικασία θα γίνει για όλα τα σημεία. Επιπλέον, καλούμε τη συνάρτηση terminate,η οποία αν έχει αλλάξει η ομάδα ενός point θέτει ένα flag= 1.

```java
public void assignLabels() {
    float min = 100.0f,eDistance = 0.0f;
    int minCenter = -1;

    for (int i = 0; i < numberofPoints; i++){
        for (int j = 0; j < M; j++){
            eDistance = euclideanDistance(points[i], centers[j]);
            if(eDistance < min){
                min = eDistance;
                minCenter = j + 1;
            }
        }
        terminate(points[i], minCenter);
        points[i].previousTeam = minCenter;
        min=100.0f;
    }
}

public float euclideanDistance(Point p, Point c) {
    return (float) Math.sqrt(Math.pow(p.x1 - c.x1, 2) + Math.pow(p.x2 - c.x2, 2));
}
```

```java
    public void terminate(Point p, int currentTeam) {
        if (p.previousTeam != currentTeam) {
            flag = 1;
        }
    }
}
```

Στο αρχείο Main.java έχουμε ένα flag = 1 αρχικοποιήσει από την αρχή του προγράμματος ,με το οποίο ελέγχουμε αν τα σημεία αλλάζουν ομάδες. Αν έστω και ένα σημείο αλλάξει ομάδα ο αλγόριθμος συνεχίζει ενώ αν σταματήσουν να αλλάζουν ομάδες σημαίνει ότι έχουμε βρει τα τελικά μας κέντρα ,τα εκτυπώνουμε και υπολογίζουμε το σφάλμα ομαδοποίησης. Επομένως ,μέσα στο while κάνουμε το flag =0 καλούμε την assignlabels ,η οποία αντίστοιχα καλεί την terminate που κάνει τον έλεγχο για τυχόν αλλαγή ομάδας και αλλαγή flag.Επιπλέον στη main αρχικοποιούμε το M=3,6,9,12 ανάλογα τον αριθμό των ομάδων που θέλουμε να εκτελέσουμε τον αλγόριθμο, θέτουμε τον αριθμό των points ,δημιουργούμε τα αντίστοιχα αντικείμενα να καλέσουμε τις μεθόδους των κλάσεων Load,Kmeans,Error
.

```java
public class Main {
    public static void main(String[] args) {

        int numberofPoints = 1200;
        int M = 12; // Arithmos omadwn
        int flag = 1; //Flag gia ton termatismo tou algorithmou
        Point points[] = new Point[numberofPoints];
        Point centers[] = new Point[M];
        String filepath = args[0];

        KMeans kmeans = new KMeans(points, centers, numberofPoints, flag, M);
        Error error = new Error(points, centers, numberofPoints);
        Load load = new Load(points, centers, numberofPoints, M);
        load.readFile(filepath);
        load.findRandomCenters();

        while (flag == 1) {
            flag = 0;
            kmeans.assignLabels();
            kmeans.findNewCenters();
        }
        //Ektypose ta telika kentra:
        System.out.println("Final centers:");
        for (int i = 0; i < M; i++) {
            System.out.println("Cluster " + (i + 1) + ": " + centers[i].x1 + ", " + centers[i].x2);
        }
        error.computeError();
    }
}
```

Τέλος στο αρχείο Error.java υπολογίζουμε σφάλμα ομαδοποίησης για κάθε σημείο.

```java
public class Error {
    private Point[] points;
    private Point[] centers;
    private int numberofPoints;
    public Error(Point[] points, Point[] centers, int numberofPoints) {
        this.points = points;
        this.centers = centers;
        this.numberofPoints = numberofPoints;
    }
    public void computeError(){
        float  sfalma = 0;
        for (int i=0;i<numberofPoints;i++){
            sfalma += Math.sqrt(Math.pow(centers[points[i].previousTeam - 1].x1-points[i].x1,2)+Math.pow(centers[points[i].previousTeam - 1].x2-points[i].x2,2));
        }
        System.out.println();
        System.out.println("Sfalma Omadopoihshs:" + sfalma);
    }
}
```

Αποτελέσματα

Για M = 3 η λύση με το μικρότερο σφάλμα ομαδοποίησης είναι

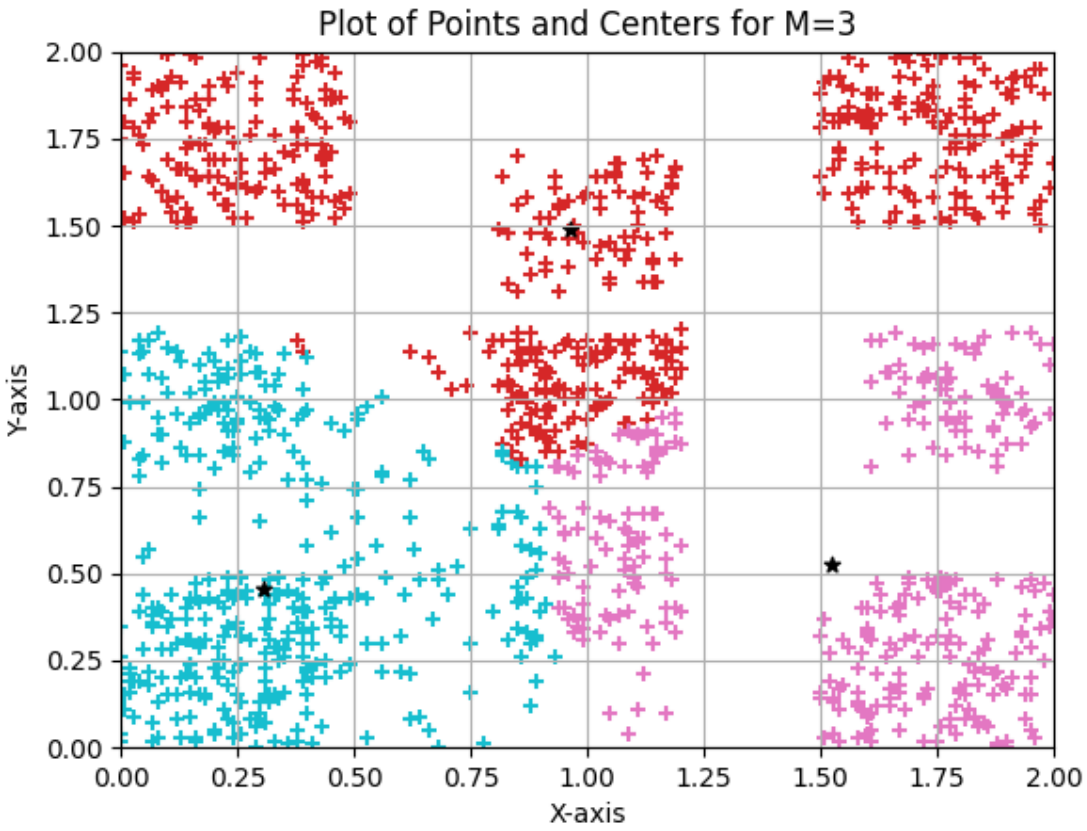
Cluster 1: 0.964575, 1.4872342

Cluster 2: 1.5259637, 0.52445793

Cluster 3: 0.30844447, 0.4552066

Σφάλμα Ομαδοποίησης:613.0433

Και το αντίστοιχο σχήμα με τα παραδείγματα και τις θέσεις κέντρων είναι το εξής:



Για M= 6 η λύση με το μικρότερο σφάλμα ομαδοποίησης είναι

Cluster 1: 1.7727562, 0.46789712

Cluster 2: 1.5021677, 1.6761503

Cluster 3: 0.6277456, 0.41341048

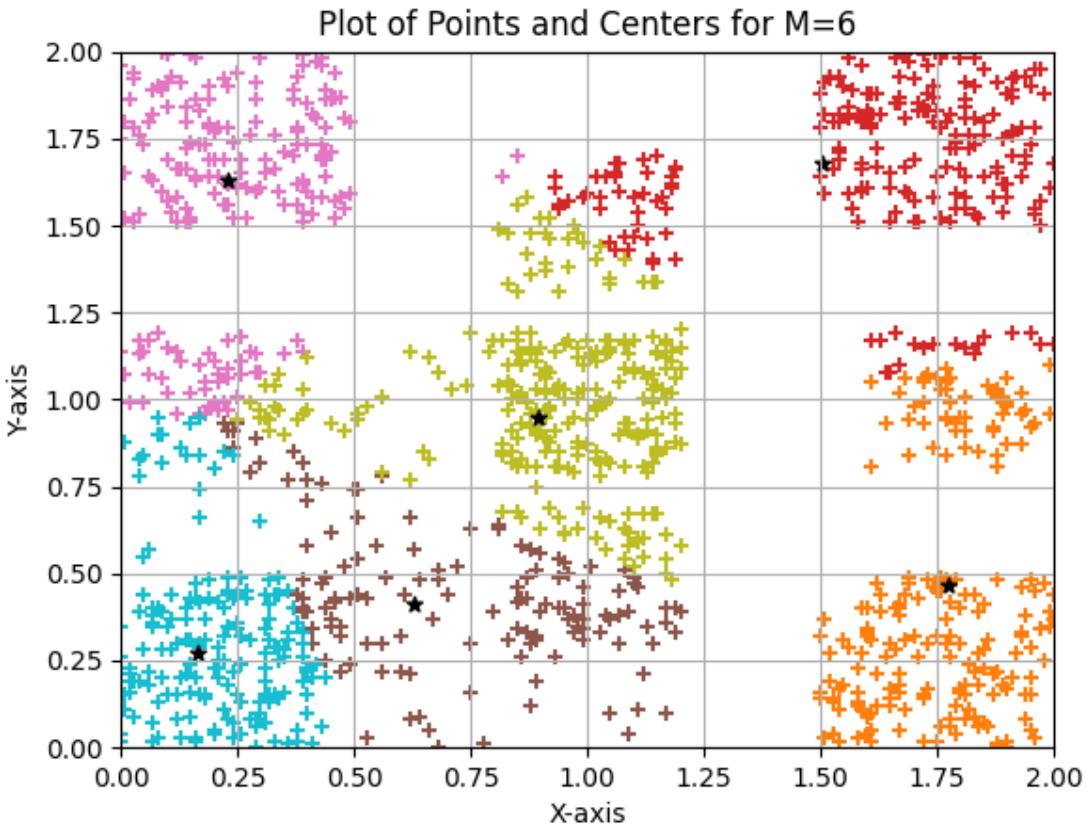
Cluster 4: 0.23083799, 1.6310061

Cluster 5: 0.8932606, 0.94818836

Cluster 6: 0.16568182, 0.27151513

Σφάλμα Ομαδοποίησης:370.80197

Και το αντίστοιχο σχήμα με τα παραδείγματα και τις θέσεις κέντρων είναι το εξής:



Για M= 9 η λύση με το μικρότερο σφάλμα ομαδοποίησης είναι

Cluster 1: 1.8034664, 1.0104003

Cluster 2: 1.7406, 1.7600665

Cluster 3: 0.5254238, 0.17610168

Cluster 4: 1.0044702, 1.2068198

Cluster 5: 0.25113338, 1.732467

Cluster 6: 0.98695314, 0.5697656

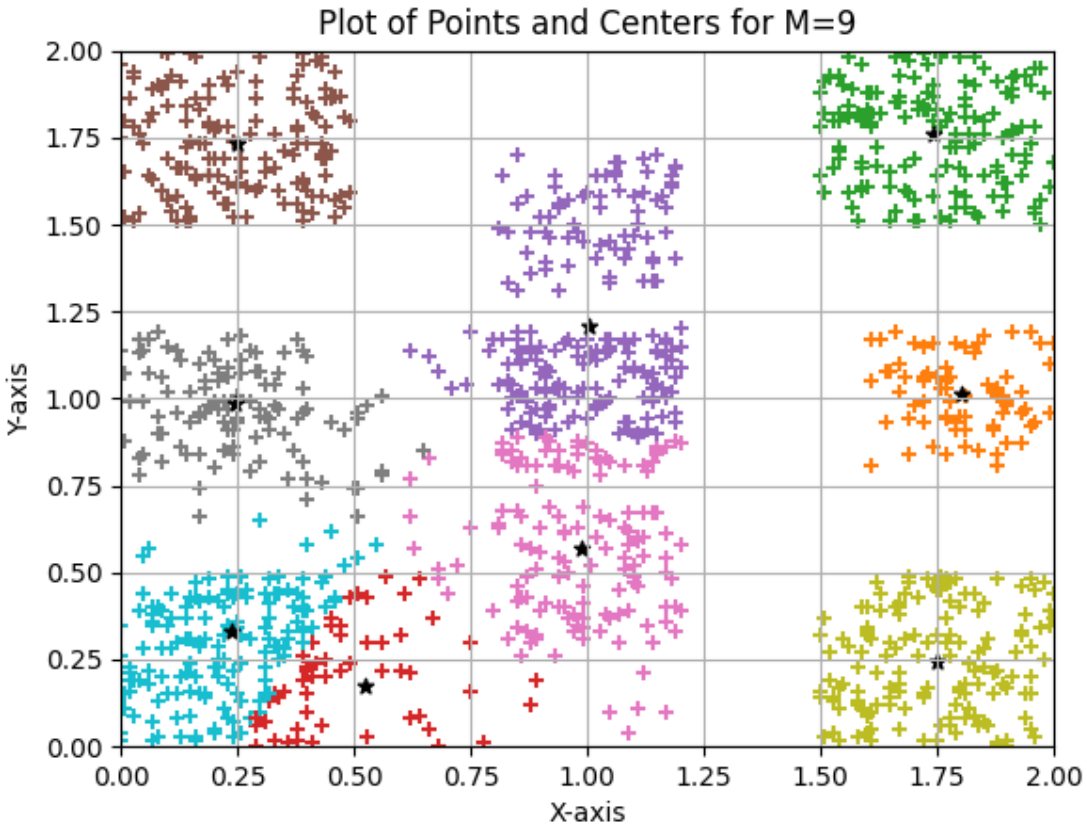
Cluster 7: 0.24428578, 0.9871428

Cluster 8: 1.749466, 0.24479999

Cluster 9: 0.23975901, 0.32963863

Σφάλμα Ομαδοποίησης:238.56721

Και το αντίστοιχο σχήμα με τα παραδείγματα και τις θέσεις κέντρων είναι το εξής:



Για M = 12 η λύση με το μικρότερο σφάλμα ομαδοποίησης είναι

Cluster 1: 1.876452, 1.8670969

Cluster 2: 0.9980004, 1.1822561

Cluster 3: 1.7115619, 0.24412498

Cluster 4: 0.21824567, 1.0914913

Cluster 5: 0.2516923, 1.763308

Cluster 6: 1.7887499, 1.6342187

Cluster 7: 0.38602945, 0.14720589

Cluster 8: 1.8034664, 1.0104003

Cluster 9: 1.5008452, 1.7994365

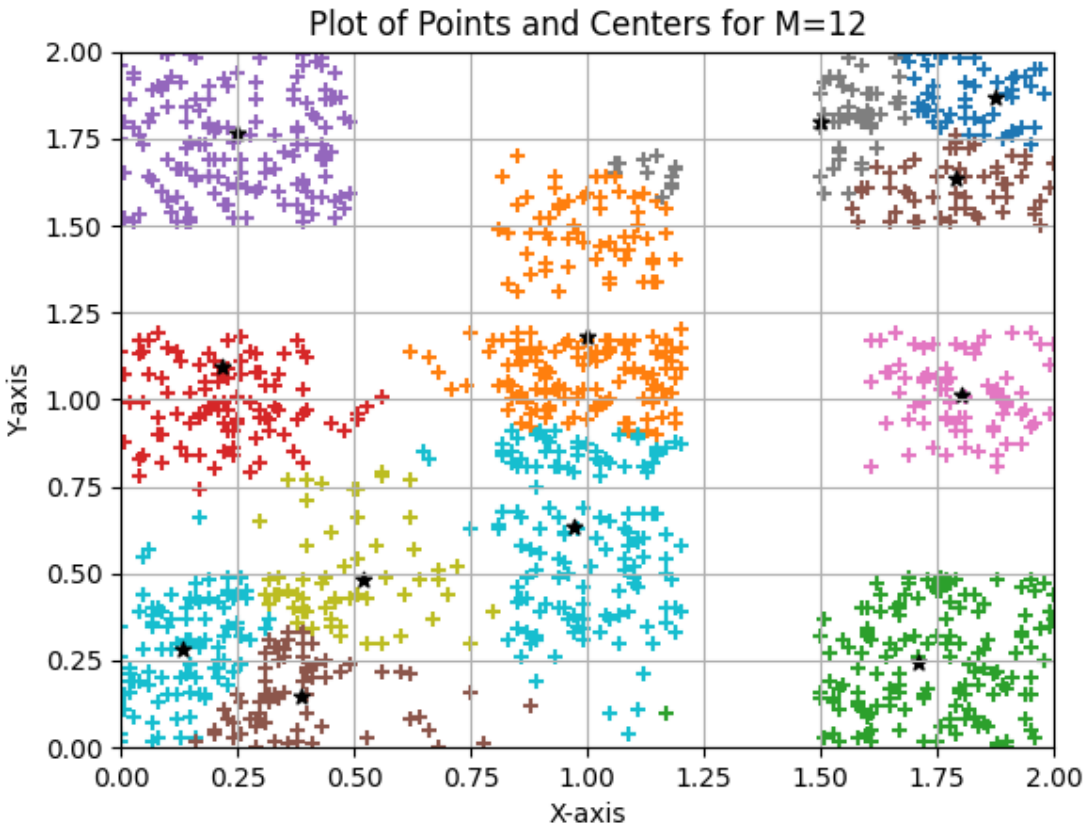
Cluster 10: 0.51975, 0.4838749

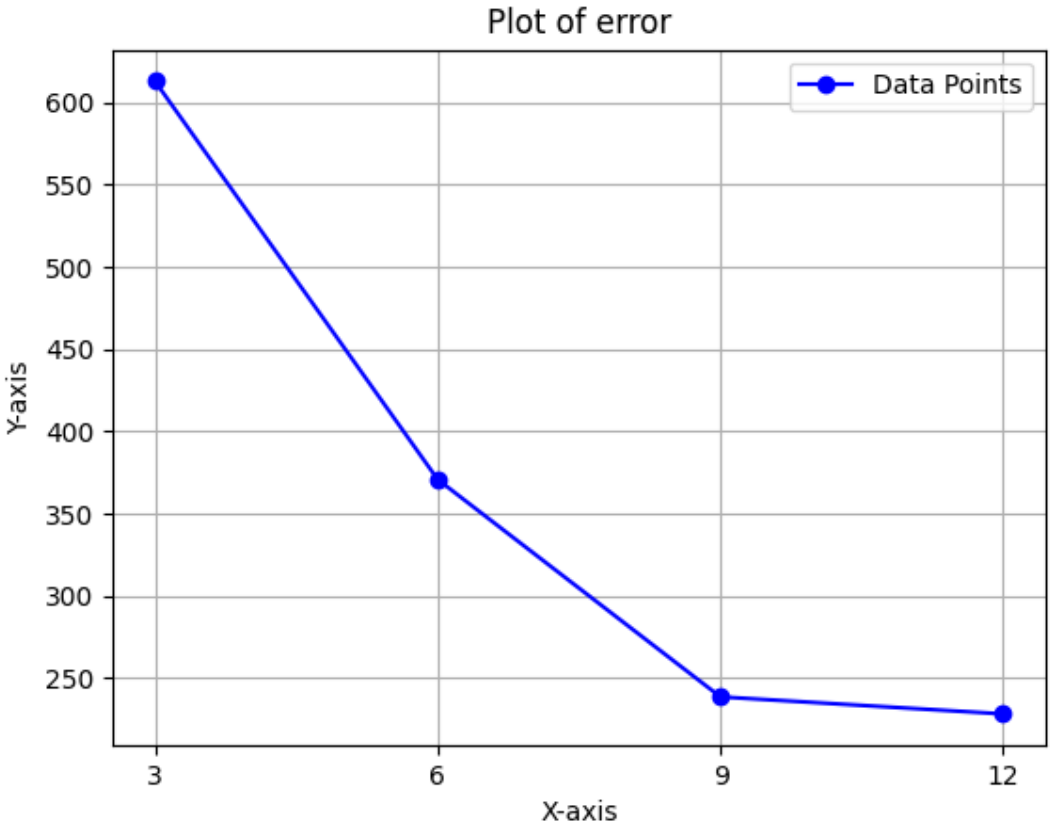
Cluster 11: 0.13445655, 0.28217384

Cluster 12: 0.97049993, 0.63333327

Σφάλμα Ομαδοποίησης:228.19499

Και το αντίστοιχο σχήμα με τα παραδείγματα και τις θέσεις κέντρων είναι το εξής:





Όπως παρατηρούμε στο παραπάνω διάγραμμα του σφάλματος ομαδοποίησης ,το σφάλμα μειώνεται όσο αυξάνεται το M. Το σφάλμα μειώνεται απότομα καθώς πλησιάζει στις 9 ομάδες και έπειτα μέχρι να φτάσει στις 12 ομάδες είναι μικρότερη η μείωση σαν να σταθεροποιείται. Άρα συμπεραίνουμε ότι ο πραγματικός αριθμός των ομάδων είναι 9.

Τα διαγράμματα δημιουργήθηκαν με python scripts.

Τρόπος εκτέλεσης προγράμματος

Στον ergasia2 έχουμε τα αρχεία Load.java,Error.java,Main.java,KMeans.java,Point.java που αποτελούν τον αλγόριθμό μας καθώς και το examples.txt που περιέχει τα 1200 παραδείγματα. Εφόσον όλα τα αρχεία μεταγλωττιστούν με την εντολή javac Load.java κτλ ,γράφουμε την εντολή **java Main examples.txt** ώστε να εκτελεστεί ο αλγόριθμός μας. Είναι απαραίτητο το examples.txt να είναι στο ίδιο φάκελο με τα αρχεία αυτά καθώς δε θα μπορεί να βρεί το αρχείο αλλιώς.

Στον υποφάκελο << createExamples>> έχουμε το αρχείο Main.java με το οποίο αν το τρέξουμε μια φορά θα παραχθεί το αρχείο examples.txt