/\*

\* @Student : Kanimozhi Kalaichelvan

\* @Professor : Cong Pu

\* @Marshall ID : 901881363

\* @Date : 17 Nov 2018

\* @Course : CS620 Applied Algorithms

\* @Program : Clustering Algorithm

\*/

**import** java.util.ArrayList;

// Class to implement K-Means Clustering Algorithm

**public** **class** ClusteringKpoints {

// Give Number of Clusters required to be 2

**private** **static** **int** *NumofClusters* = 2, *randomNum1*, *randomNum2*,*status*;

**private** **static** **double**[][] *mean*= **new** **double**[*NumofClusters*][*NumofClusters*],*prevmean* = **new** **double**[*NumofClusters*][*NumofClusters*];

**private** **static** **double**[][] *inputsequence* = {{0,1}, {1,0}, {0.5,0.5}, {5,6}, {6,5}, {5.5,5.5}};//{3,15,2,5,17,1,18,4,20,16}; //{4,10,1,7,3,8,9,2}; //{3,15,2,5,17,1,18,4,20,16,11,9,13,14};

**private** **static** ArrayList<Integer> *Array1* = **new** ArrayList<Integer>();

**private** **static** ArrayList<Integer> *Array2* = **new** ArrayList<Integer>();

// Function to generate Random Number

**public** **int** randomGen()

{

**int** randomNum = (**int**) (Math.*random*()\*5);

System.***out***.println("Randomnly chosen Number is : "+randomNum);

**return** randomNum;

}

// Function to Calculate Mean of Each Cluster

**public** **static** **double**[][] calculateMeanDist()//(float old\_mean1,float old\_mean2)

{

**double** sum1 = 0, sum2 = 0, sum3 = 0, sum4 =0, avg1, avg2, avg3, avg4;

**int** size1, size2;

size1 = *Array1*.size();

size2 = *Array2*.size();

**for** (**int** i=0; i< *Array1*.size(); i++)

{

**int** elementinseq = *Array1*.get(i);

sum1 += *inputsequence*[elementinseq][0];

sum2 += *inputsequence*[elementinseq][1];

}

avg1 = sum1 / size1;

avg2 = sum2 / size1;

System.***out***.println("Avg 1 : "+avg1);

System.***out***.println("Avg 2 : "+avg2);

*mean*[0][0] = avg1;

*mean*[0][1] = avg2;

**for** (**int** j=0; j< *Array2*.size(); j++)

{

**int** elementinseq2 = *Array2*.get(j);

sum3 += *inputsequence*[elementinseq2][0];

sum4 += *inputsequence*[elementinseq2][1];

}

avg3 = sum3 / size2;

avg4 = sum4 / size2;

System.***out***.println("Avg 3 : "+avg3);

System.***out***.println("Avg 4 : "+avg4);

*mean*[1][0] = avg3;

*mean*[1][1] = avg4;

System.***out***.println("Mean 0 is : ");

**for**(**int** q=0;q<*mean*[0].length;q++)

{

System.***out***.print(*mean*[0][q] + " ");

}

System.***out***.println();

System.***out***.println("Mean 1 is : ");

**for**(**int** q=0;q<*mean*[1].length;q++)

{

System.***out***.print(*mean*[1][q] + " ");

}

**return** *mean*;

}

// Function to calculate Distance between two points

**public** **double** calculatedistance2pts(**double**[] meantemp,**double**[] inputsequence2)

{

**double** distance;

**double** x1, y1, x2, y2;

x1 = meantemp[0];

y1 = meantemp[1];

x2 = inputsequence2[0];

y2 = inputsequence2[1];

System.***out***.println("x1 is : "+ x1 +"y1 is : "+ y1 +"x2 is : "+ x2 +"y2 is : "+ y2 );

distance = Math.*sqrt*(Math.*pow*(y2 - y1, 2) + Math.*pow*(x2 - x1,2));

**return** distance;

}

// Function to Calculate the Point with Nearest mean for Each element of the InputSequence Array

**public** **int** calculateNearestMean(**double**[] inputsequence2)

{

*calculateMeanDist*();

// System.out.println(mean[0]);

**double** minmean = calculatedistance2pts(*mean*[0],inputsequence2);

**int** array=0;

// System.out.println();

// System.out.println("Number is "+inputseqnum+"\nMinmean is "+minmean);

**for**(**int** d=0;d<*mean*[0].length;d++)

{

**double** temp = calculatedistance2pts(*mean*[d],inputsequence2);

**if** (temp<minmean)

{

minmean = temp;

// System.out.println("Minmean is "+minmean);

array = d;

// System.out.println("Recommended Array is: "+ array);

}

}

**return** array;

}

// Function to add Elements Initially

**public** **void** addElements()

{

**for** (**int** i=0; i<*inputsequence*.length ; i++)

{

// Add Elements to Array1

**int** arr = calculateNearestMean(*inputsequence*[i]);

**if**(arr==0 && i!=*randomNum1* && i!=*randomNum2*)

{

*Array1*.add(i);

}

// Add Elements to Array2

**else** **if**(arr==1 && i!=*randomNum1* && i!=*randomNum2*)

{

*Array2*.add(i);

}

}

}

/\*

// Function that adds elements to array in each iteration until mean remains same

public void continueLoopAddElements()

{

for (int i=0; i< inputsequence.length; i++)

{

// Add Elements to Array1

int arr = calculateNearestMean(inputsequence[i]);

if(arr==0)

{

if(Array1.contains(inputsequence[i]))

{

Array1.remove(Array1.indexOf(inputsequence[i]));

}

if(Array2.contains(inputsequence[i]))

{

Array2.remove(Array2.indexOf(inputsequence[i]));

}

Array1.add(inputsequence[i]);

}

// Add Elements to Array2

else if(arr==1)

{

if(Array1.contains(inputsequence[i]))

{

Array1.remove(Array1.indexOf(inputsequence[i]));

}

if(Array2.contains(inputsequence[i]))

{

Array2.remove(Array2.indexOf(inputsequence[i]));

}

Array2.add(inputsequence[i]);

}

}

}

/\* Function to check if mean remains SAME

\* If mean remains same stop the loop

\* If mean is different for all elements of sequence, calculate minimum mean distance and add element to that array

\*

public static int checkstatus(double[] prevmean)

{

mean=calculateMean();

for(int l=0;l<mean.length;l++)

{

if(mean[l]!=prevmean[l])

{

return status=0;

}

}

status=1;

return status;

}

\*/

// MAIN Function where program starts Execution

**public** **static** **void** main(String[] args)

{

// **TODO** Auto-generated method stub

// Instantiate the Clustering Algorithm class

ClusteringKpoints ca = **new** ClusteringKpoints();

*randomNum1* = ca.randomGen();

*Array1*.add(*randomNum1*);

System.***out***.print("Array 1 is ");

**for**(**int** q=0;q<*inputsequence*[*randomNum1*].length;q++)

{

System.***out***.print(*inputsequence*[*randomNum1*][q] + " ");

}

System.***out***.println();

*randomNum2* = ca.randomGen();

// Create randomnumber2 which is not equal to randomnumber1

**while**(*randomNum2*==*randomNum1*)

{

*randomNum2* = ca.randomGen();

}

*Array2*.add(*randomNum2*);

System.***out***.print("Array 2 is ");

**for**(**int** q=0;q<*inputsequence*[*randomNum2*].length;q++)

{

System.***out***.print(*inputsequence*[*randomNum2*][q] + " ");

}

System.***out***.println();

*calculateMeanDist*();

System.***out***.println();

ca.addElements();

//// System.out.println("Array 1 is "+Array1);

//// System.out.println("Array 2 is "+Array2);

// System.out.println();

// while(status!=1)

// {

// prevmean = calculateMean();

// ca.continueLoopAddElements();

// status = checkstatus(prevmean);

// System.out.println("Array 1 is "+Array1);

// System.out.println("Array 2 is "+Array2);

// System.out.println();

// }

System.***out***.println("Cluster 1 is "+*Array1*);

System.***out***.println("Cluster 2 is "+*Array2*);

}

}

Randomnly chosen Number is : 2

Array 1 is 0.5 0.5

Randomnly chosen Number is : 4

Array 2 is 6.0 5.0

Avg 1 : 0.5

Avg 2 : 0.5

Avg 3 : 6.0

Avg 4 : 5.0

Mean 0 is :

0.5 0.5

Mean 1 is :

6.0 5.0

Cluster 1 is [2]

Cluster 2 is [4]

Randomnly chosen Number is : 4

Array 1 is 6.0 5.0

Randomnly chosen Number is : 2

Array 2 is 0.5 0.5

Avg 1 : 6.0

Avg 2 : 5.0

Avg 3 : 0.5

Avg 4 : 0.5

Mean 0 is :

6.0 5.0

Mean 1 is :

0.5 0.5

Avg 1 : 6.0

Avg 2 : 5.0

Avg 3 : 0.5

Avg 4 : 0.5

Mean 0 is :

6.0 5.0

Mean 1 is :

0.5 0.5 x1 is : 6.0y1 is : 5.0x2 is : 0.0y2 is : 1.0

x1 is : 6.0y1 is : 5.0x2 is : 0.0y2 is : 1.0

x1 is : 0.5y1 is : 0.5x2 is : 0.0y2 is : 1.0

Avg 1 : 6.0

Avg 2 : 5.0

Avg 3 : 0.25

Avg 4 : 0.75

Mean 0 is :

6.0 5.0

Mean 1 is :

0.25 0.75 x1 is : 6.0y1 is : 5.0x2 is : 1.0y2 is : 0.0

x1 is : 6.0y1 is : 5.0x2 is : 1.0y2 is : 0.0

x1 is : 0.25y1 is : 0.75x2 is : 1.0y2 is : 0.0

Avg 1 : 6.0

Avg 2 : 5.0

Avg 3 : 0.5

Avg 4 : 0.5

Mean 0 is :

6.0 5.0

Mean 1 is :

0.5 0.5 x1 is : 6.0y1 is : 5.0x2 is : 0.5y2 is : 0.5

x1 is : 6.0y1 is : 5.0x2 is : 0.5y2 is : 0.5

x1 is : 0.5y1 is : 0.5x2 is : 0.5y2 is : 0.5

Avg 1 : 6.0

Avg 2 : 5.0

Avg 3 : 0.5

Avg 4 : 0.5

Mean 0 is :

6.0 5.0

Mean 1 is :

0.5 0.5 x1 is : 6.0y1 is : 5.0x2 is : 5.0y2 is : 6.0

x1 is : 6.0y1 is : 5.0x2 is : 5.0y2 is : 6.0

x1 is : 0.5y1 is : 0.5x2 is : 5.0y2 is : 6.0

Avg 1 : 5.5

Avg 2 : 5.5

Avg 3 : 0.5

Avg 4 : 0.5

Mean 0 is :

5.5 5.5

Mean 1 is :

0.5 0.5 x1 is : 5.5y1 is : 5.5x2 is : 6.0y2 is : 5.0

x1 is : 5.5y1 is : 5.5x2 is : 6.0y2 is : 5.0

x1 is : 0.5y1 is : 0.5x2 is : 6.0y2 is : 5.0

Avg 1 : 5.5

Avg 2 : 5.5

Avg 3 : 0.5

Avg 4 : 0.5

Mean 0 is :

5.5 5.5

Mean 1 is :

0.5 0.5 x1 is : 5.5y1 is : 5.5x2 is : 5.5y2 is : 5.5

x1 is : 5.5y1 is : 5.5x2 is : 5.5y2 is : 5.5

x1 is : 0.5y1 is : 0.5x2 is : 5.5y2 is : 5.5

Cluster 1 is [4, 3, 5]

Points in Cluster 1 :

6.0 5.0

5.0 6.0

5.5 5.5

Cluster 2 is [2, 0, 1]

Points in Cluster 2 :

0.5 0.5

0.0 1.0

1.0 0.0