```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Linq;
using System.Drawing;
using System.IO;

namespace kMeans
{
    class Program
    {
        static void Main(string[] args)
        {
            List<kVector> Vectors = new List<kVector>();
            List<kVector> ClassCenters = new List<kVector>();
            int dimensions = 3;
            int vectorCount = 0;
            int imageWidth;
            int imageHeight;
            int classCount;


            #region Input

            Bitmap myBitmap = new Bitmap(@"C:/temp/winter.jpg");
            imageWidth = myBitmap.Width;
            imageHeight = myBitmap.Height;
            for (int x = 0; x < myBitmap.Width; x++)
            {
                for (int y = 0; y < myBitmap.Height; y++)
                {
                    Color pixelColor = myBitmap.GetPixel(x, y);

                    kVector kV = new kVector(3); // RGB
                    kV.dimensions[0] = pixelColor.R;
                    kV.dimensions[1] = pixelColor.G;
                    kV.dimensions[2] = pixelColor.B;

                    Vectors.Add(kV);
                    vectorCount++;
                }
            }

            Console.Write("Class count: ");
            classCount = int.Parse(Console.ReadLine());

            #endregion

            #region Klassenzentren
            List<kVector> tempVectors = Vectors;
            int tempCount = vectorCount - 1;

            for (int i = 0; i < classCount; i++)
            {
                Random r = new Random();
                int random = r.Next(0, tempCount);
```

```
                Console.WriteLine("Zentrum " + i + ": ");
                for (int d = 0; d < dimensions; d++)
                {
                    Console.Write(tempVectors[random].dimensions[d] + ", ");
                }
                ClassCenters.Add(tempVectors[random]);

                tempVectors.RemoveAt(random);

                tempCount -= 1;
            }
            #endregion

            #region Klassenzentrumzuordnung

            foreach (kVector vector in Vectors)
            {
                double distance = 1000000000000000000;
                int vectorCounter = 0;
                int vectorNumber = 0;

                foreach (kVector clVector in ClassCenters)
                {
                    double sqrtSum = 0;
                    for (int d = 0; d < dimensions; d++)
                    {
                        sqrtSum += Math.Pow(vector.dimensions[d] -
clVector.dimensions[d], 2);
                    }
                    double sum = Math.Sqrt(sqrtSum);

                    if (sum < distance)
                    {
                        distance = sum;
                        vectorNumber = vectorCounter;
                    }
                    vectorCounter++;
                }

                vector.classCentrum = vectorNumber;
            }

            #endregion

            #region Iteration
            for (int i = 0; i < 20; i++)
            {
                int c = 0;
                foreach(kVector clVector in ClassCenters)
                {
                    double[] sum = new double[dimensions];
                    int count = 0;

                    foreach (kVector vector in Vectors)
                    {
                        if (vector.classCentrum == c)
                        {
```

```
                        for (int d = 0; d < dimensions; d++)
                        {
                            sum[d] += vector.dimensions[d];
                        }
                        count++;
                    }
                }

                for (int a = 0; a < dimensions; a++)
                {
                    clVector.dimensions[a] = sum[a] / count;
                }

                c++;
            }



            // Neu zuordnen
            foreach (kVector vector in Vectors)
            {
                double distance = 100000000000000000;
                int vectorCounter = 0;
                int vectorNumber = 0;

                foreach (kVector clVector in ClassCenters)
                {
                    double sqrtSum = 0;
                    for (int d = 0; d < dimensions; d++)
                    {
                        sqrtSum += Math.Pow(vector.dimensions[d] -
clVector.dimensions[d], 2);
                    }
                    double sum = Math.Sqrt(sqrtSum);

                    if (sum < distance)
                    {
                        distance = sum;
                        vectorNumber = vectorCounter;
                    }
                    vectorCounter++;
                }

                vector.classCentrum = vectorNumber;
            }

        }
        #endregion


        for (int i = 0; i < classCount; i++)
        {
            Console.WriteLine("Zentrum " + i + ": ");
            for (int d = 0; d < dimensions; d++)
            {
                Console.Write(ClassCenters[i].dimensions[d] + ", ");
            }
            Console.WriteLine("");
```

```
            }


            // Save Image File

            kVector[] ClassCentersArray = ClassCenters.ToArray();

            for (int x = 0; x < myBitmap.Width; x++)
            {
                for (int y = 0; y < myBitmap.Height; y++)
                {
                    Color pixelColor = myBitmap.GetPixel(x, y);

                    kVector kV = new kVector(3); // RGB
                    kV.dimensions[0] = pixelColor.R;
                    kV.dimensions[1] = pixelColor.G;
                    kV.dimensions[2] = pixelColor.B;

                    double distance = 1000000000000000000;
                    int vectorCounter = 0;
                    int vectorNumber = 0;

                    foreach (kVector clVector in ClassCenters)
                    {
                        double sqrtSum = 0;
                        for (int d = 0; d < dimensions; d++)
                        {
                            sqrtSum += Math.Pow(kV.dimensions[d] -
clVector.dimensions[d], 2);
                        }
                        double sum = Math.Sqrt(sqrtSum);

                        if (sum < distance)
                        {
                            distance = sum;
                            vectorNumber = vectorCounter;
                        }
                        vectorCounter++;
                    }

                    kVector kVv = ClassCentersArray[vectorNumber];
                    Color newColor = Color.FromArgb(Convert.ToInt32(kVv.dimensions[0]),
Convert.ToInt32(kVv.dimensions[1]), Convert.ToInt32(kVv.dimensions[2]));
                    myBitmap.SetPixel(x, y, newColor);
                }
            }

            myBitmap.Save(@"C:/temp/winter-" + classCount + ".jpg",
System.Drawing.Imaging.ImageFormat.Jpeg);

            Console.ReadLine();
        }
    }


    class kVector
    {
```

```
        public double[] dimensions;
        public int classCentrum;

        public kVector(int dimensions)
        {
            this.dimensions = new double[dimensions];
        }
    }
}
```