Data je tekstualna datoteka koja sadrži koordinate tačaka. Napišite klasu koja će da modeluje sadržaj datoteke. Zatim, ispišite sadržaj datoteke, odredite između kojih tačaka je najmanje rastojanje i ispišite vrednost tog rastojanja.

A number with black numbers

Description automatically generated with medium confidence

Klasa Tačka

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tacke

{

internal class Tacka

{

private int x;

private int y;

public void setX(int x)

{

this.x = x;

}

public int getX()

{

return x;

}

public void setY(int y1)

{

y= y1;

}

public int getY()

{

return y;

}

public double rastojanje(Tacka t)

{

double r = 0;

r = Math.Sqrt( (t.x-x)\*(t.x-x)+(t.y-y)\*(t.y-y) );

return r;

}

}

}

Program.cs

namespace Tacke

{

internal class Program

{

static void Main(string[] args)

{

string path = @"c:\tmp\tacke.txt";

List<string> list = new List<string>();

list=File.ReadAllLines(path).ToList();

List<Tacka> tackas= new List<Tacka>();

for(int i=0; i<list.Count;i++)

{

string[] data = list[i].Split(',');

int x = int.Parse(data[0]);

int y = int.Parse(data[1]);

Tacka t=new Tacka();

t.setX(x);

t.setY(y);

tackas.Add(t);

}

foreach(var item in tackas)

{

Console.WriteLine("x={0}; y={1}",item.getX(), item.getY());

}

Tacka t1 = tackas[0];

Tacka t2= tackas[1];

double minDistance = t1.rastojanje(t2);

for (int i=0; i<tackas.Count-1; i++)

{

for (int j = i + 1; j < tackas.Count; j++)

{

double distance = tackas[i].rastojanje(tackas[j]);

if (minDistance > distance)

{

t1 = tackas[i];

t2 = tackas[j];

minDistance= distance;

}

}

}

Console.WriteLine("Najkrace rastojanje je izmedju tacaka x={0}, y={1} i x={2}, y={3}", t1.getX(), t1.getY(), t2.getX(), t2.getY());

Console.WriteLine("Nakrace rastojanje je "+ minDistance);

}

}

}

Zadatak koji pokazuje upotrebu HashSeta.

namespace hashSet

{

internal class Program

{

static void Main(string[] args)

{

HashSet<string> set = new HashSet<string>();

set.Add("C");

set.Add("C++");

set.Add("C#");

set.Add("Ada");

set.Add("Java");

set.Add("Php");

Console.WriteLine("Sadrzaj hashSeta: ");

foreach(string item in set)

{

Console.WriteLine(item);

}

Console.WriteLine("Ukupan broj elemenata: "+set.Count);

HashSet<string> set1 = new HashSet<string>();

set1.Add("C");

set1.Add("C++");

set1.Add("SQL");

set1.Add("A");

set1.Add("B");

Console.WriteLine("Sadrzaj hashSeta1: ");

foreach (string s in set1)

{

Console.WriteLine(s);

}

Console.WriteLine("Razlika hashSeta 1 i 2: ");

set1.ExceptWith(set);

foreach (string s in set1)

{

Console.WriteLine(s);

}

Console.WriteLine("Unija hashSeta 1 i 2: ");

set.UnionWith(set1);

foreach(string s in set)

{

Console.WriteLine(s);

}

Console.WriteLine("Presek hashSeta 1 i 2: ");

set.IntersectWith(set1);

foreach (string s in set)

{

Console.WriteLine(s);

}

}

}

}