KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

OBJEKTINIO PROGRAMAVIMO PAGRINDAI I (P175B117)

Laboratorinio darbo ataskaita

Atliko:

IFIN5/3 gr. studentas

Karolis Staniukynas

2015 m. gruodžio 14 d.

Priėmė:

Lektorius, Mindaugas Jančiukas

KAUNAS 2015

TURINYS

1. Pažintis su klase 3

1.1. Darbo užduotis 3

1.2. Programos tekstas 3

1.3. Pradiniai duomenys ir rezultatai 6

2. Objektų rinkinys 7

2.1. Darbo užduotis 7

2.2. Programos tekstas 7

2.3. Pradiniai duomenys ir rezultatai 10

3. Konteinerinė klasė 11

3.1. Darbo užduotis 11

3.2. Programos tekstas 11

3.3. Pradiniai duomenys ir rezultatai 15

4. Teksto analizė ir redagavimas 16

4.1. Darbo užduotis 16

4.2. Programos tekstas 16

4.3. Pradiniai duomenys ir rezultatai 17

5. Susieti rinkiniai 18

5.1. Darbo užduotis 18

5.2. Programos tekstas 18

5.3. Pradiniai duomenys ir rezultatai 22

# Pažintis su klase

## Darbo užduotis

U2-10. Malūnas.

Sukurkite klasę Grūdai, kuri turėtų kintamuosius grūdų rūšiai, rupumui ir malimo nuostoliams (procentais, %) saugoti. Malūnas mala 3 skirtingų rūšių grūdus. Raskite, kurie malami grūdai rupiausi ir kurių grūdų malimo nuostoliai mažiausi.

Papildykite klasę Grūdai kintamuoju, skirtu miltų tankiui saugoti. Sukurkite klasę Malūnas, kuri turėtų kintamąjį malūno pavadinimui saugoti ir 3 kintamuosius (kiekvienai grūdų rūšiai, tonomis), skirtus saugoti per mėnesį reikalingam gauti miltų kiekiui. Suskaičiuokite, kiek tonų kiekvienos rūšies grūdų turi būti pristatyta kas mėnesį į malūną.

Papildykite klasę Malūnas kintamuoju, skirtu saugoti bendram miltų talpyklų tūriui (litrais). Ar malūno per mėnesį primaltas miltų kiekis telpa talpyklose?

## Programos tekstas

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace U2\_10.Malunas

{

class Grudai

{

private string rusis;

private double rupumas,

malim\_nuostoliai,

tankis;

public Grudai(string rusis, double rupumass,double malim\_nuost,double tank)

{

this.rusis = rusis;

rupumas = rupumass;

malim\_nuostoliai = malim\_nuost;

tankis = tank;

}

public string imtrusi() { return rusis; }

public double imtrupumas() { return rupumas; }

public double imtnuostolis() { return malim\_nuostoliai; }

public double imttankis() { return tankis; }

}

class Malunas

{

private string pavadinimas;

private double pirm,

antr,

tret,

turis;

public Malunas(string pavadinimass ,double pirm ,double antr ,double tret,double turis)

{

this.pavadinimas = pavadinimass;

this.pirm=pirm;

this.antr=antr;

this.tret=tret;

this.turis = turis;

}

public string imtpav() { return pavadinimas; }

public double imtpirm() { return pirm; }

public double imtantr() { return antr; }

public double imttret() { return tret; }

public double imtturi() { return turis; }

}

class Program

{

static void Main(string[] args)

{

string rus,pavad;

double rupum,

nuost,

rupiaus = 0,

tankis,

turis,

maz=100,

bendras\_miltu\_kiekis;

// reikiamas miltu kiekis:

double pirm,

antr,

tret;

Console.Write("Įveskite grudų rūšį: ");

rus = Console.ReadLine();

Console.Write("Rupumas: ");

rupum = double.Parse(Console.ReadLine());

Console.Write("Nuostoliai %: ");

nuost = double.Parse(Console.ReadLine());

Console.Write("miltu tantis kg/m3: ");

tankis = double.Parse(Console.ReadLine());

Grudai rusis1;

rusis1 = new Grudai(rus,rupum,nuost,tankis);

if(maz>rusis1.imtnuostolis())

{

maz = rusis1.imtnuostolis();

}

if(rupiaus<rusis1.imtrupumas())

{

rupiaus = rusis1.imtrupumas();

}

//--------------------------------------------------

Console.Write("Įveskite grudų rūšį: ");

rus = Console.ReadLine();

Console.Write("Rupumas: ");

rupum = double.Parse(Console.ReadLine());

Console.Write("Nuostoliai %: ");

nuost = double.Parse(Console.ReadLine());

Console.Write("miltu tantis kg/m3: ");

tankis = double.Parse(Console.ReadLine());

Grudai rusis2;

rusis2 = new Grudai(rus, rupum, nuost,tankis);

if (maz > rusis2.imtnuostolis())

{

maz = rusis2.imtnuostolis();

}

if (rupiaus < rusis2.imtrupumas())

{

rupiaus = rusis2.imtrupumas();

}

//---------------------------------------------------

Console.Write("Įveskite grudų rūšį: ");

rus = Console.ReadLine();

Console.Write("Rupumas: ");

rupum = double.Parse(Console.ReadLine());

Console.Write("Nuostoliai %: ");

nuost = double.Parse(Console.ReadLine());

Console.Write("miltu tantis kg/m3: ");

tankis = double.Parse(Console.ReadLine());

Grudai rusis3;

rusis3 = new Grudai(rus, rupum, nuost,tankis);

if (maz > rusis3.imtnuostolis())

{

maz = rusis3.imtnuostolis();

}

if (rupiaus < rusis3.imtrupumas())

{

rupiaus = rusis3.imtrupumas();

}

Console.Write("Iveskite maluno pavadinima: ");

pavad = Console.ReadLine();

Console.Write("{0} reikiamas miltu kiekis t: ", rusis1.imtrusi());

pirm = double.Parse(Console.ReadLine());

Console.Write("{0} reikiamas miltu kiekis t: ", rusis2.imtrusi());

antr = double.Parse(Console.ReadLine());

Console.Write("{0} reikiamas miltu kiekis t: ", rusis3.imtrusi());

tret = double.Parse(Console.ReadLine());

Console.Write("bendras talpyklu turis L: ");

turis = double.Parse(Console.ReadLine());

turis = turis / 1000;

Console.Clear();

Malunas pav;

pav = new Malunas(pavad, pirm, antr, tret,turis);

//rupiausi miltai

Console.WriteLine("{0} malunas", pav.imtpav());

if (rusis1.imtrupumas() == rusis2.imtrupumas() && rusis1.imtrupumas() == rusis3.imtrupumas())

Console.WriteLine("visu miltu rupumas vienodas");

else

{

if (rupiaus == rusis1.imtrupumas())

Console.WriteLine(" rupiausi miltai: {0}", rusis1.imtrusi());

if (rupiaus == rusis2.imtrupumas())

Console.WriteLine("rupiausi miltai: {0}", rusis2.imtrusi());

if (rupiaus == rusis3.imtrupumas())

Console.WriteLine("rupiausi miltai: {0}", rusis3.imtrusi());

}

if (rusis1.imtnuostolis() == rusis2.imtnuostolis() && rusis1.imtnuostolis() == rusis3.imtnuostolis())

Console.WriteLine("visu grudu nuostolis vienodas");

else

{

if (maz == rusis1.imtnuostolis())

Console.WriteLine("maziausias malimo nuostolis {0} grudu", rusis1.imtrusi());

if (maz == rusis2.imtnuostolis())

Console.WriteLine("maziausias malimo nuostolis {0} grudu", rusis2.imtrusi());

if (maz == rusis3.imtnuostolis())

Console.WriteLine("maziausias malimo nuostolis {0} grudu", rusis3.imtrusi());

}

//reikiamo grudu kiekio skaiciavimas:

Console.WriteLine("{0} grudu reikia {1} t", rusis1.imtrusi(),Math.Round (pav.imtpirm() / (1 - rusis1.imtnuostolis() / 100.00),2));

Console.WriteLine("{0} grudu reikia {1} t", rusis2.imtrusi(),Math.Round( pav.imtantr() / (1 - rusis2.imtnuostolis() / 100.00),2));

Console.WriteLine("{0} grudu reikia {1} t", rusis3.imtrusi(),Math.Round( pav.imttret() / (1 - rusis3.imtnuostolis() / 100.00),2));

//bendras miltu kiekis kubiniais metrais;

bendras\_miltu\_kiekis = (pav.imtpirm()\*1000/rusis1.imttankis()) + (pav.imtantr()\*1000/rusis2.imttankis()) + (pav.imttret()\*1000/rusis3.imttankis());

if (pav.imtturi() >= bendras\_miltu\_kiekis)

Console.WriteLine("Miltai telpa ");

else

Console.WriteLine("Talpyklose nepakankamai vietos ");

}

}

}

## Pradiniai duomenys ir rezultatai

Pradiniai duomenys:

Rūšis: Ruginiai.

Rūpumas: 20.

Nuostolis(%):20.

Tankis kg/m3: 1.

Rūšis: Kvietiniai.

Rūpumas: 30.

Nuostolis(%):10.

Tankis kg/m3: 2.

Rūšis: Grikių.

Rūpumas: 100.

Nuostolis(%):5.

Tankis kg/m3: 0,5.

Maluno pavadinimas: Kauno.

Reikiamas miltų kiekis(t):1

Reikiamas miltų kiekis(t):20

Reikiamas miltų kiekis(t):30

Bendras talpyklų tūris(l):30000000

Rezultatai:

Rupiausi miltai:Grikiu.

Maziausias malimo nuostolis- Grikiu grudų.

Rūginių grudų reikia 1,25 t.

Kvietiniai grudų reikia:22,22 t.

Grikiu grudų reikia:31,58 t.

Talpyklose nepakankamai vietos.

# Objektų rinkinys

## Darbo užduotis

**U3–10. Siuvykla**

Siuvyklos siuvamų kostiumų duomenys saugomi faile: modelio pavadinimas, medžiagos pavadinimas, pasiuvimui reikalingos medžiagos ilgis bei plotis ir atraižų kiekis (procentais, %). Pirmoje eilutėje yra siuvyklos, kuri siuva kostiumus, pavadinimas. Sukurkite klasę Kostiumas, kuri turėtų kintamuosius kostiumų duomenims saugoti. Kurio modelio kostiumui pasiūti reikia daugiausiai medžiagos ir kurio modelio kostiumui atraižų lieka mažiausiai?

Papildykite programą veiksmais su dviejų siuvyklų siūlomų kostiumų rinkiniais. Kiekvienos siuvyklos duomenys saugomi atskiruose failuose. Kuriame sąraše yra kostiumas, kuriam reikia daugiausia medžiagos? Surašykite į atskirą rinkinį visus abiejų sąrašų kostiumų modelius, kurie yra vidutinio dydžio, duomenis. Vidutinis kostiumas yra tas, kuriam pasiūti reikalingos medžiagos kiekis skiriasi nuo visų modelių reikalingos medžiagos aritmetinio vidurkio ne daugiau, kaip p proc.

## Programos tekstas

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace U3\_10.Siuvykla

{

class Kostiumai

{

private string pavadinimas;

private double ilgis;

private double plotis;

private double atraizos;

public Kostiumai(string pavadinimas,double ilgis,double plotis,double atraizos)

{

this.pavadinimas = pavadinimas;

this.ilgis = ilgis;

this.plotis = plotis;

this.atraizos = atraizos;

}

public string imtpav() { return pavadinimas; }

public double imtilg() { return ilgis; }

public double imtplot() { return plotis; }

public double imtatr() { return atraizos; }

}

class Program

{

const string skait1 = "...\\...\\siuv1.txt";

const string skait2 = "...\\...\\siuv2.txt";

const string ras = "...\\...\\Rez.txt";

static void Main(string[] args)

{

int kiek1;

int kiek2;

string siuv1;

string siuv2;

string daug1; // 1os siuvyklos daugiausiai medziagos reikalaujantis modelis

string maz1;//1os siuvyklos maziausiai atraizu;

double s1;// plotas didz

double m1; //atraizu skaic

string daug2; // 2os siuvyklos daugiausiai medziagos reikalaujantis modelis

string maz2;

double s2;//plotas didz

double m2;

double visas\_s1;

double visas\_s2;

double p;

Console.WriteLine("iveskite skirtumas p %");

p =int.Parse( Console.ReadLine());

if (File.Exists(ras))

File.Delete(ras);

Kostiumai[] k1 = new Kostiumai[100];

Kostiumai[] k2 = new Kostiumai[100];

skaitymas(skait1, k1, out kiek1,out siuv1); //perskaitomi duomenys i pirmo failo

skaitymas(skait2, k2, out kiek2, out siuv2); //skaitoma is antro failo

skaiciavimas(k1,kiek1, out daug1,out s1,out maz1,out m1,out visas\_s1,p);

skaiciavimas(k2, kiek2, out daug2, out s2,out maz2,out m2,out visas\_s2,p);

//rasoma siuvyklu daugiausiai medziagos reikalaujantys kostiumai ir maziausiai atraizu paliekantys kostiumai

rasymas(ras, k1, k2,siuv1,siuv2, kiek1, kiek2, s1, s2, daug1, daug2, maz1, maz2);

rasym(ras, k1, kiek1, visas\_s1,siuv1); //rasomas 1 siuvyklos vidutiniu kostiumu sarasas

rasym(ras, k2, kiek2, visas\_s2, siuv2); //rasomas 1 siuvyklos vidutiniu kostiumu sarasas

}

static void skaitymas(string skait,Kostiumai[] k,out int kiek,out string siuv)

{

kiek = 0;

string pav;

siuv="nera";

double ilgis, plotis, atraizos;

using (StreamReader reader = new StreamReader(skait))

{

string line;

line = reader.ReadLine();

siuv = line;

string[] parts;

while ((line = reader.ReadLine()) != null)

{

parts = line.Split(' ');

pav = parts[0];

ilgis = double.Parse(parts[1]);

plotis = double.Parse(parts[2]);

atraizos = double.Parse(parts[3]);

k[kiek] = new Kostiumai(pav, ilgis, plotis, atraizos);

kiek++;

}

}

}

static void skaiciavimas(Kostiumai[] k,int kiek,out string daug,out double s,out string maz,out double m,out double visas\_s,double p)

{

visas\_s = 0;

daug = k[0].imtpav();

s = k[0].imtilg() \* k[0].imtplot();

maz = k[0].imtpav();

m = k[0].imtilg() \* k[0].imtplot() \* (k[0].imtatr() / 100);

for (int i=0;i< kiek;i++)

{

visas\_s = visas\_s + k[i].imtilg() \* k[i].imtplot();

// randamas dougiausiai medziagos reikalaujantis modelis

if (s< k[i].imtilg() \* k[i].imtplot())

{

daug = k[i].imtpav();

s = k[i].imtilg() \* k[i].imtplot();

}

//randama kuris modelis su maziausiai atraizu

if(m> k[i].imtilg() \* k[i].imtplot() \* (k[i].imtatr() / 100))

{

maz = k[i].imtpav();

m = k[i].imtilg() \* k[i].imtplot() \* (k[i].imtatr() / 100);

}

}

visas\_s = visas\_s \*(p/100);

}

static void rasymas(string ras,Kostiumai[] k1,Kostiumai[] k2,string siuv1,string siuv2,int kiek1,int kiek2,double s1,double s2,string daug1,string daug2,string maz1,string maz2)

{

using (var fr = File.AppendText(ras))

{

fr.WriteLine("{0} siuvykloje ",siuv1);

fr.WriteLine("daugiausiai medziagos reika: {0}", daug1);

fr.WriteLine("maziausiai atraizu lieka nuo: {0}", maz1);

fr.WriteLine(" ");

fr.WriteLine("{0} siuvykloje",siuv2);

fr.WriteLine("daugiausiai medziagos reika: {0}", daug2);

fr.WriteLine("maziausiai atraizu lieka nuo: {0}", maz2);

fr.WriteLine(" ");

if(s1>s2)

{

fr.WriteLine("{0} siuvykloje daugiausiai medziagos reikalaujantis modelis", siuv1);

fr.WriteLine("{0} modelis", daug1);

}

if (s1 < s2)

{

fr.WriteLine("{0} siuvykloje daugiausiai medziagos reikalaujantis modelis", siuv2);

fr.WriteLine("{0} modelis", daug2);

}

}

}

static void rasym(string ras,Kostiumai[] k,int kiek,double visas\_s,string siuv)

{

int j = 0;

using (var fr = File.AppendText(ras))

{

fr.WriteLine("{0} siuvyklos vidutiniai kostiumai ",siuv);

fr.WriteLine(" ");

for(int i=0;i< kiek;i++)

{

if (k[i].imtilg() \* k[i].imtplot() <= visas\_s)

j++;

}

if (j > 0)

{

for (int i = 0; i < kiek; i++)

{

if (k[i].imtilg() \* k[i].imtplot() <= visas\_s)

fr.WriteLine("{0,-15}", k[i].imtpav());

}

fr.WriteLine(" ");

}

else

fr.WriteLine("nera");

}

}

}

}

## Pradiniai duomenys ir rezultatai

Pradiniai duomenys:

moteriski kostiumai

didelis 2 3 80

vidutinis 3 5 10

mazas 5 6 20

vyriski

didelis 5 8 10

vidutinis 4 8 20

mazas 3 5 30

Įvedama p reikšmė: 20

Rezultatai:

moteriski kostiumai siuvykloje

daugiausiai medziagos reika: mazas

maziausiai atraizu lieka nuo: vidutinis

vyriski siuvykloje

daugiausiai medziagos reika: didelis

maziausiai atraizu lieka nuo: didelis

vyriski siuvykloje daugiausiai medziagos reikalaujantis modelis

didelis modelis

moteriski kostiumai siuvyklos vidutiniai kostiumai

didelis

vyriski siuvyklos vidutiniai kostiumai

mazas

# Konteinerinė klasė

## Darbo užduotis

U4-10. Kopijavimo darbai Tekstiniame faile surašyti kopijavimo darbai: popieriaus formatas, lapų skaičius, kopijų skaičius, vienpusis ar dvipusis kopijavimas, kopijų grupavimas, papildoma kopijos apdorojimo funkcija. Papildoma kopijų apdorojimo funkcija, jeigu jos reikia, gali būti susegimas arba skylių išmušimas. Parašykite programą, kuri spausdintų darbų sąrašą lentele, suskaičiuotų, kiek darbų turi papildomą susegimo funkciją ir kiek darbų turi skylių išmušimo funkciją, surastų daugiausiai kopijavimo aparato ciklų reikalaujantį darbą. Papildykite programą veiksmais, kurie leistų atrinkti vienpusio kopijavimo darbus ir surikiuotų juos pagal formatą ir bendrą reikalingo popieriaus kiekį mažėjimo tvarka.

## Programos tekstas

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Collections;

namespace \_10\_Kopijavimas

{

class Kopija

{

private string pav, formatas,Vp\_Dp,funkcija,grupavimas;

private int lapu\_sk, kop\_sk;

public Kopija()

{

pav = " ";

lapu\_sk = 0;

formatas = " ";

Vp\_Dp = " ";

funkcija = " ";

grupavimas = " ";

kop\_sk = 0;

}

public Kopija(string pav,string formatas,int lapu\_sk,int kop\_sk,string Vp\_Dp,string grupavimas,string funkcija)

{

this.pav = pav;

this.lapu\_sk = lapu\_sk;

this.formatas = formatas;

this.Vp\_Dp = Vp\_Dp;

this.funkcija = funkcija;

this.grupavimas = grupavimas;

this.kop\_sk = kop\_sk;

}

public string imtpav() { return pav; }

public string imtfr() { return formatas; }

public int imtlap() { return lapu\_sk; }

public int imtkop() { return kop\_sk; }

public string imtpuse() { return Vp\_Dp; }

public string imtgrup() { return grupavimas; }

public string imtfu() { return funkcija; }

public override string ToString()

{

string eilute;

eilute = string.Format("|{0,10} |{1,4} |{2,4} |{3,10} |{4,10} |{5,10} |{6,10}|",pav, formatas, lapu\_sk, kop\_sk, Vp\_Dp, grupavimas,funkcija);

return eilute;

}

public static bool operator >=(Kopija kp1, Kopija kp2)

{

int p = String.Compare(kp1.imtfr(), kp2.imtfr(), StringComparison.CurrentCulture);

bool v = (kp1.imtkop() \* kp1.imtlap() >= kp2.imtkop() \* kp2.imtlap());

// return (p < 0 || (p == 0 && v==true));

return (p < 0 || (p == 0 && v));

}

public static bool operator <=(Kopija kp1, Kopija kp2)

{

int p = String.Compare(kp1.imtfr(), kp2.imtfr(), StringComparison.CurrentCulture);

bool v = (kp1.imtkop()\*kp1.imtlap() <= kp2.imtkop()\*kp2.imtlap());

return (p > 0 || (p == 0 && v ));

}

}

class Kopijos

{

private Kopija[] K;

private int kiek;

public Kopijos()

{

kiek = 0;

K = new Kopija[100];

}

public Kopija imtko(int i) { return K[i]; }

public int imtkiek() { return kiek; }

public void detikopija(Kopija kop)

{ K[kiek++] = kop; }

//public static bool operator <=(Kopija kp1, Kopija kp2)

//{

//int p = String.Compare(kp1.imtfr, kp2.imtfr, StringComparison.CurrentCulture);

//int v = String.Compare(kp1.imtkop, kp2.imtkop, StringComparison.CurrentCulture);

// return (p < 0 || (p == 0 && v < 0));

//}

//public static bool operator >=(Kopija kp1, Kopija kp2)

//{

// int p = String.Compare(kp1.imtfr, kp2.imtfr, StringComparison.CurrentCulture);

// int v = String.Compare(kp1.imtkop, kp2.imtkop, StringComparison.CurrentCulture);

// return (p > 0 || (p == 0 && v > 0));

//}

//public void Rikiuoti()

//{

// for (int i = 0; i <kiek-1;i++)

// {

// Kopijos min=

// }

// }

public void Rikiuoti()

{

for (int i = 0; i <kiek -1; i++)

{

Kopija min = K[i];

int im = i;

for (int j = i + 1; j < kiek; j++)

if (K[j] >= min)

{ // naudojamas užklotas operatorius <=

min = K[j];

im = j;

}

K[im] = K[i];

K[i] = min;

}

}

}

class Program

{

const string Failas = "...//...//Duom.txt";

static void Main(string[] args)

{

Kopijos K = new Kopijos();

int kiek,sus\_sk,ism\_sk;

skaitymas(Failas, ref K, out kiek);

Kopija[] mas = new Kopija[K.imtkiek()];

Kopija[] mas1 = new Kopija[K.imtkiek()];

Kopija[] mas2 = new Kopija[K.imtkiek()];

Kopija[] mas3 = new Kopija[K.imtkiek()];

veiksmai(ref K, mas,mas2, out kiek,out sus\_sk,out ism\_sk);

K.Rikiuoti();

// Array.Sort(mas2, mas2);

int kiek1 = 0;

for (int i = 0; i < kiek; i++)

{

if (K.imtko(i).imtpuse() == "vienpusis")

{

mas2[kiek1] = K.imtko(i);

kiek1++;

}

}

// rasymas(ref K, mas, mas1,mas2, ref kiek,ref kiek1);

Console.WriteLine("ismusimo funkcija");

Console.WriteLine(ism\_sk);

Console.WriteLine("susegimo funkcija");

Console.WriteLine(sus\_sk);

// Console.WriteLine(mas2[0].ToString());

// int min = K.imtkop(0).imtkop() \* K.imtkop(0).imtlap();

int end = kiek1;

//----------------------------------------------------------

// for (int i=0;i<kiek1-1;i++)

// {

// for(int j=i+1;j<kiek1;j++)

// {

// if (mas2[i].imtkop()\*mas2[i].imtlap()< mas2[j].imtkop() \* mas2[j].imtlap())

// {

// mas3[0] = mas2[i];

// mas2[i] = mas2[j];

// mas2[j] = mas3[0];

// }

// }

//}

rasymas(ref K, mas, mas1, mas2, ref kiek, ref kiek1);

}

static void skaitymas(string Failas,ref Kopijos K,out int kiek)

{

string pav, formatas, Vp\_Dp, funkcija, grupavimas,line;

int lapu\_sk,kop\_sk ;

kiek = 0;

using (StreamReader reader = new StreamReader(Failas))

{

string[] parts;

while((line=reader.ReadLine())!=null)

{

parts = line.Split(' ');

pav = parts[0];

formatas = parts[1];

lapu\_sk = int.Parse(parts[2]);

kop\_sk = int.Parse(parts[3]);

Vp\_Dp = parts[4];

grupavimas = parts[5];

funkcija = parts[6];

Kopija kop = new Kopija(pav,formatas, lapu\_sk, kop\_sk, Vp\_Dp, grupavimas, funkcija);

K.detikopija(kop);

kiek++;

}

}

}

static void veiksmai(ref Kopijos K,Kopija[] mas,Kopija[] mas2,out int kiek,out int sus\_sk,out int ism\_sk )

{

sus\_sk = 0;

kiek = 0;

ism\_sk = 0;

for(int i=0;i<K.imtkiek();i++)

{

mas[i] = K.imtko(i);

kiek++;

if (K.imtko(i).imtfu() == "susegimas")

sus\_sk++;

if (K.imtko(i).imtfu() == "ismusimas")

ism\_sk++;

}

}

static void rasymas(ref Kopijos K,Kopija[] mas,Kopija[] mas1,Kopija[] mas2, ref int kiek,ref int kiek1)

{

int cik = 0;

// Console.WriteLine(kiek1);

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.WriteLine("| Kopijos |");

Console.WriteLine("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|");

for(int i=0;i<kiek; i++)

{

Console.WriteLine("{0}",mas[i].ToString());

}

for (int i = 0; i < kiek; i++)

{

if (cik < K.imtko(i).imtkop() \* K.imtko(i).imtlap())

{

cik = K.imtko(i).imtkop() \* K.imtko(i).imtlap();

mas1[0] = K.imtko(i);

}

//Console.WriteLine("{0}", mas1[0].ToString());

}

Console.WriteLine("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|");

Console.WriteLine("| vienpuses kopijos pagal kopiju skaiciu maz tvarka |");

Console.WriteLine("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|");

for (int i=0;i<kiek1;i++)

{

Console.WriteLine("{0}", mas2[i].ToString());

}

Console.WriteLine("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|");

Console.WriteLine("daugiausiai ciklu");

Console.WriteLine("{0}", mas1[0].ToString());

}

}

}

## Pradiniai duomenys ir rezultatai

d1 A4 10 8 vienpusis psl susegimas

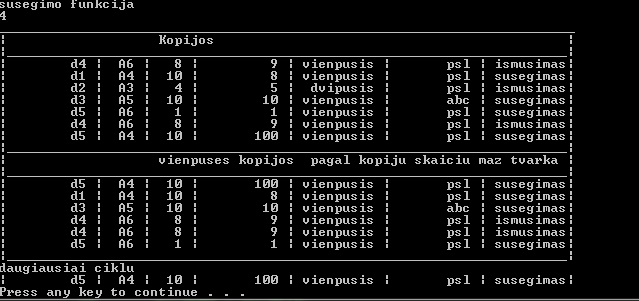
d2 A3 4 5 dvipusis psl ismusimas

d3 A4 10 10 vienpusis abc susegimas

d5 A6 1 1 vienpusis psl susegimas

d4 A6 8 9 vienpusis psl ismusimas

d5 A4 10 100 vienpusis psl susegimas



# Teksto analizė ir redagavimas

## Darbo užduotis

U5-10. Skaitmenys Tekstiniame faile pateiktas tekstas. Žodžiai iš eilutės į kitą eilutę nekeliami. Skyrikliai žinomi. Kiekvienoje eilutėje po vieną teksto žodį, sudarytą tik iš skaitmenų, jei toks yra, kartu su už jo esančiais skyrikliais perkelkite į eilutės pradžią.

## Programos tekstas

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace \_u10lab

{

class Program

{

const string CFd = "..\\..\\Duomenys.txt";

const string CFr = "..\\..\\Rezultatai.txt";

static void Main(string[] args)

{

Apdoroti(CFd, CFr);

Console.WriteLine("Programa baigė darbą!");

}

/// <summary>

/// Skaitymas ir apdorojimas

/// </summary>

/// <param name="fv"></Duomenu failas>

/// <param name="fvr"></ Rezultatu failas>

static void Apdoroti(string fv, string fvr)

{

string[] lines = File.ReadAllLines(fv, Encoding.GetEncoding(1257));

using (var fr = File.CreateText(fvr))

{

foreach (string line in lines)

{

if (line.Length > 0)

{

string nauja = line;

Redagavimas(line, out nauja);

if (nauja.Length > 0)

fr.WriteLine(nauja);

}

}

}

}

/// <summary>

/// Redagavimas

/// </summary>

/// <param name="line"></perskaityta eilute>

/// <param name="nauja"></naujai kuriama eilute kuria graziname>

/// <returns></returns>

static bool Redagavimas(string line, out string nauja)

{

string eil;

eil = line;

nauja = eil;

int a = 0;

int yra = 0;

char[] mass= { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };

char[] mass2 = { ',', '.', '-', ';', ':', '?', '!' };

for (int i = 0; i < line.Length ; i++)

{

for(int j=0;j<=9; j++)

{

if(line[i]==mass[j])

{

eil = nauja.Insert(a, line[i].ToString());

nauja = eil.Remove(i+1,1);

a++;

yra++;

}

}

if(yra>0)

{

for(int b=0;b<=5;b++)

{

if (line[i] == mass2[b])

{

eil = nauja.Insert(a, line[i].ToString());

nauja = eil.Remove(i+1 ,1) ;

a++;

}

}

}

}

return false;

}

}

}

## Pradiniai duomenys ir rezultatai

Kuciu ryta ..2...,,1,,1,...

Anksti Kuciu ryta šeimininke budina savo vyra:

- Eik greiciau,1260 saulei netekejus, kur 1 dalges kabo, ištverk dalges. Dalges

padek po stogu, o dalgiakocius sudek svirnan.

Šeimininke ieško kubilo lanko, kad but visai apskritas, nepertrukes niekur.

Ta lanka neša vištu tvartan, vidury tvarto paguldo. O tada šeimininke skuba

tvartant prie 9kodžio, kur buna žirniai supilti. Šeimininke tuos žirnius semia

negailedama didžiuli gorciu, kad visos vištos prilestu lig soties. Šeimininke

pila tuos žirnius tan kubilo lankan, kad nei vienas žirnis nebut už kubilo

lanko - kad vištos visos detu kiaušinius vienan daiktan, nemetytu kiaušiniu.

Berdama žirnius tan lankan, 9šeimininke garsiai sako vištom:

- Žiurekite, kad nei vieno kiaušinio, nei vieno niekur nepamestut, visus

vienon vieton dekite!

Na ir visos vištos šeimininkes 5.40 isakyma vykdo.

2...,,1,,1,...Kuciu ryta ..

Anksti Kuciu ryta eimininke budina savo vyra:

1260,1,. - Eik greiciau, saulei netekejus kur dalges kabo itverk dalges Dalges

padek po stogu, o dalgiakocius sudek svirnan.

eimininke ieko kubilo lanko, kad but visai apskritas, nepertrukes niekur.

Ta lanka nea vitu tvartan, vidury tvarto paguldo. O tada eimininke skuba

9,.tvartant prie kod˛io kur buna ˛irniai supilti eimininke tuos ˛irnius semia

negailedama did˛iuli gorciu, kad visos vitos prilestu lig soties. eimininke

pila tuos ˛irnius tan kubilo lankan, kad nei vienas ˛irnis nebut u˛ kubilo

lanko - kad vitos visos detu kiauinius vienan daiktan, nemetytu kiauiniu.

9:Berdama ˛irnius tan lankan, eimininke garsiai sako vitom

- ˇiurekite, kad nei vieno kiauinio, nei vieno niekur nepamestut, visus

vienon vieton dekite!

5.40.Na ir visos vitos eimininkes isakyma vykdo

# Susieti rinkiniai

## Darbo užduotis

U6–10. Leidiniai Pirmoje failo eilutėje nurodytas prenumeruojamų leidinių kiekis ir mėnesio dienų skaičius. Tolesnėse eilutėse pateikta informacija apie leidinius: pavadinimas, mėnesio prenumeratos kaina, banko pavadinimas, sąskaitos numeris, procentai, atiduodami prenumeratos rinkėjui. Žemiau pateikta informacija apie kiekvieno leidinio prenumeratos eigą: leidiniai (eilutės), kiek jų užsakyta (stulpeliai). Suskaičiuokite, kiek per mėnesį kurio leidinio užsakyta. Nustatykite, kuriam leidiniui blogiausiai sekasi. Kiekvienam bankui sudarykite pavedimų sąrašą, kad pervesti leidiniui prenumeratos pinigus. Nustatykite, kuris leidinys daugiausiai uždirbs.

## Programos tekstas

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Leidiniai

{

/// <summary>

/// Saugomi leidiniu failai

/// </summary>

class Leidinys

{

private string pavad;

private int kaina;

private string bankas;

private int kodas, procentai, Kiekis = 0;

public void Deti(string pav, int kaina, string bankas, int kodas, int procentai)

{

this.pavad = pav;

this.kaina = kaina;

this.bankas = bankas;

this.kodas = kodas;

this.procentai = procentai;

}

public void DetiKiek(int Kiek) { Kiekis = Kiek; }

public string imtpav() { return pavad; }

public int imtkaina() { return kaina; }

public string imtbank() { return bankas; }

public int imtkoda() { return kodas; }

public int imtproc() { return procentai; }

public int imtKiek() { return Kiekis; }

}

class leidykla

{

private Leidinys[] Leidiniai;

public int n { get; set; }

private int[,] A;

public int m { get; set; }

public leidykla()

{

n = 0;

Leidiniai = new Leidinys[100];

m = 0;

A = new int[100, 30];

}

public Leidinys imti(int nr) { return Leidiniai[nr]; }

public void Deti(Leidinys ob) { Leidiniai[n++] = ob; }

public void DetiA(int i,int j,int r) {A[i, j] = r; }

public int imtiA(int i,int j) { return A[i, j]; }

public void pakeistleid(int nr,Leidinys lei) { Leidiniai[nr] = lei; }

public void papild()

{

int suma;

Leidinys lei;

for (int i=0;i< n;i++)

{

suma = 0;

for (int j = 0; j < m; j++)

suma = suma + A[i, j];

lei = imti(i);

lei.DetiKiek(suma);

pakeistleid(i, lei);

}

}

}

class Program

{

const string duom = "...\\...\\duom.txt";

// const string duom2 = "...\\...\\duom1.txt";

const string rez = "...\\...\\rez.txt";

static void Main(string[] args)

{

if (File.Exists(rez))

File.Delete(rez);

int n, m;

// string[] bankai= { "SEB", "DNB", "Swedbank", "Medicinos bankas", "Danske Bank" };

leidykla leid1 = new leidykla();

Skaityti(duom, ref leid1, out n, out m);

//Spaustinti(rez, leid1);

Skaitytkiek(duom, ref leid1, ref n, ref m);

//Spausdintikiek(rez, leid1);

//Console.WriteLine(leid1.m);

int[] kiek;

kiek = new int[n];

kuris(leid1,kiek);

Spaustinti(rez, leid1,kiek);

//Spausdintikiek(rez, leid1);

}

/// <summary>

/// perskaitoma leidiniu informacija

/// </summary>

/// <param name="fd"></param>

/// <param name="leid1"></param>

/// <param name="n"></param>

/// <param name="m"></param>

static void Skaityti(string fd, ref leidykla leid1, out int n, out int m)

{

string pav, bankas; int kaina, kodas, procentai;

string line;

using (StreamReader reader = new StreamReader(fd))

{

line = reader.ReadLine();

string[] parts;

parts = line.Split(' ');

n = int.Parse(parts[0]);

m = int.Parse(parts[1]);

leid1.m = m;

for (int i = 0; i < n; i++)

{

line = reader.ReadLine();

parts = line.Split(' ');

pav = parts[0];

kaina = int.Parse(parts[1]);

bankas = parts[2];

kodas = int.Parse(parts[3]);

procentai = int.Parse(parts[4]);

Leidinys lei;

lei = new Leidinys();

lei.Deti(pav, kaina, bankas, kodas, procentai);

leid1.Deti(lei);

}

}

}

/// <summary>

/// skitoma matrica

/// </summary>

/// <param name="duom"></param>

/// <param name="leid1"></param>

/// <param name="n"></param>

/// <param name="m"></param>

static void Skaitytkiek(string duom, ref leidykla leid1, ref int n, ref int m)

{

int kiekis;

string line;

using (StreamReader reader = new StreamReader(duom))

{

line = reader.ReadLine();

for(int i=0;i<leid1.n;i++)

{

line = reader.ReadLine();

}

string[] parts;

for (int i = 0; i < leid1.n; i++)

{

line = reader.ReadLine();

parts = line.Split(' ');

for (int j = 0; j < leid1.m; j++)

{

kiekis = int.Parse(parts[j]);

leid1.DetiA(i, j, kiekis);

}

}

}

}

/// <summary>

/// isvedami atsakymai

/// </summary>

/// <param name="fv"></param>

/// <param name="leid1"></param>

/// <param name="kiek"></param>

static void Spaustinti(string fv, leidykla leid1, int[] kiek)

{

string blog;

string daug;

using (var fr = File.AppendText(fv))

{

blog = leid1.imti(0).imtpav();

daug = leid1.imti(0).imtpav();

fr.WriteLine("Pavadinimas Kiekis ");

fr.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

for (int i = 0; i < leid1.n; i++)

{

fr.WriteLine("{0,-10} {1,7}", leid1.imti(i).imtpav(), kiek[i]);

}

fr.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

for (int i = 0; i < leid1.n; i++)

for (int j = 0; j < leid1.n; j++)

if (kiek[i] > kiek[j])

blog = leid1.imti(j).imtpav();

fr.WriteLine("Blogiausiai sekas '{0}' zurnalui", blog);

for (int i = 0; i < leid1.n-1; i++)

for (int j = 0; j < leid1.n; j++)

if (kiek[i] \* leid1.imti(i).imtkaina() < kiek[j] \* leid1.imti(j).imtkaina())

daug = leid1.imti(j).imtpav();

fr.WriteLine("daugiausiai uzdirbs {0}", daug);

fr.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

fr.WriteLine(bankas(leid1,kiek));

}

}

static void Spausdintikiek(string fv, leidykla leid1)

{

using (var fr = File.AppendText(fv))

{

for (int i = 0; i < leid1.n; i++)

{

for (int j = 0; j < leid1.m; j++)

fr.Write("{0,3:d}", leid1.imtiA(i, j));

fr.WriteLine(" ");

}

}

}

static int kuris(leidykla leid1,int[] kiek)

{

int suma;

for(int i=0;i<leid1.n;i++)

{

suma = 0;

for(int j=0;j<leid1.m;j++)

{

suma = suma + leid1.imtiA(i, j);

}

kiek[i] = suma;

}

return 0;

}

/// <summary>

/// pervedimu info

/// </summary>

/// <param name="leid1"></param>

/// <param name="kiek"></param>

/// <returns></returns>

static string bankas(leidykla leid1,int []kiek)

{

string[] mas = new string[100];

string Q = "";

int n=0;

for(int i=0;i<leid1.n;i++)

{

int z = 0;

for (int j = 0; j < n; j++)

if (leid1.imti(i).imtbank() == mas[j])

z++;

if(z==0)

{

mas[n] = leid1.imti(i).imtbank();

n++;

}

}

for(int i=0;i< n;i++)

{

Q = Q + mas[i]+":\r\n";

for(int j=0;j<leid1.n;j++)

if(leid1.imti(j).imtbank()==mas[i])

{

Q = Q + leid1.imti(j).imtpav()+" "+leid1.imti(j).imtkoda()+" " +kiek[j]\*leid1.imti(j)

.imtkaina()+"Eu" ;

Q = Q + "\r\n";

}

}

return Q;

}

}

}

## Pradiniai duomenys ir rezultatai

4 4

Zmones 12 Seb 2165131 2

Keksas 10 Swed 513135 10

Mew 20 Danske 13512213 5

cookie 100000 Seb 153515 6

4 2 7 100

1 5 7 600

50 6 6 5

1 1 1 1

Rezultatai

Pavadinimas Kiekis

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Zmones 113

Keksas 613

Mew 67

cookie 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Blogiausiai sekas 'cookie' zurnalui

daugiausiai uzdirbs cookie

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Seb:

Zmones 2165131 1356Eu

cookie 153515 400000Eu

Swed:

Keksas 513135 6130Eu

Danske:

Mew 13512213 1340Eu