KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

OBJEKTINIS PROGRAMAVIMAS I (P175B118)

Darbų aplankas

Atliko:
IF-8/1 gr. studentai
Martynas Kemežys
Rytis Ališauskas

2018 m. gruodžio 19 d.

Priėmė:
Lekt. Vytautas Bukšnaitis

KAUNAS 2018

TURINYS

1. Objektų rinkinys 3

1.1. Darbo užduotis 3

1.2. Programos tekstas 3

1.3. Pradiniai duomenys ir rezultatai 7

1.4. Dėstytojo pastabos 9

2. Konteineris 10

2.1. Darbo užduotis 10

2.2. Programos tekstas 10

2.3. Pradiniai duomenys ir rezultatai 10

2.4. Dėstytojo pastabos 12

3. Paveldėjimas 13

3.1. Darbo užduotis 13

3.2. Programos tekstas 13

3.3. Pradiniai duomenys ir rezultatai 28

3.4. Dėstytojo pastabos 34

4. Teksto analizė ir redagavimas 35

4.1. Darbo užduotis 35

4.2. Programos tekstas 35

4.3. Pradiniai duomenys ir rezultatai 39

4.4. Dėstytojo pastabos 40

5. Polimorfizmas 41

5.1. Darbo užduotis 41

5.2. Programos tekstas 41

5.3. Pradiniai duomenys ir rezultatai 52

5.4. Dėstytojo pastabos 56

1. Objektų rinkinys

1.1. Darbo užduotis

Nekilnojamo turto agentūra. Turite duomenis apie šiuo metu Kaune parduodamus namus. Duomenų faile pateikta ši informacija: mikrorajonas, gatvė, namo numeris, tipas, pastatymo metai, plotas, kambarių skaičius.

- Raskite seniausią namą, ekrane atspausdinkite jo amžių, adresą, tipą ir plotą.
- Raskite, kurioje gatvėje daugiausiai parduodamų namų, ekrane atspausdinkite gatvės pavadinimą ir parduodamų namų kiekį.
- Sudarykite visų mūrinių namų, kurių plotas didesnis nei 100 kv.m., sąrašą, į failą „M100.csv“ įrašykite visus duomenis apie šiuos namus.
- Sudarykite visų blokinių namų, kurių plotas didesnis nei 150 kv.m., sąrašą, į failą „B150.csv“ įrašykite visus duomenis apie šiuos namus

1.2. Programos tekstas

Namas.cs

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

namespace U1._6
{
    /// <summary>
    /// Sukuriama namų klasė
    /// </summary>
    class Namas
    {

        public string Rajonas { get; set; }
        public string Gatve { get; set; }
        public double Namonr { get; set; }
        public string Tipas { get; set; }
        public double Metai { get; set; }
        public double Plotas { get; set; }
        public double Kambariai { get; set; }
        public Namas(string rajonas, string gatve, double namonr, string tipas,
                     double metai, double plotas, double kambariai)
        {

            Rajonas = rajonas;
            Gatve = gatve;
            Namonr = namonr;
            Tipas = tipas;
            Metai = metai;
            Plotas = plotas;
            Kambariai = kambariai;
        }
    }
}
```

DaugausiaParduodamu.cs

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

```
namespace U1._6
{
    /// <summary>
    /// Sukuriama atskira klasė būtent gatvėmis ir jose namų kiekiui išgauti
    /// </summary>
    class DaugiausiaParduodamu
    {
        public string Gatvespav { get; set; }
        public int Kiekis { get; set; }

        public DaugiausiaParduodamu(string gatvesPav, int kiekis)
        {
            Gatvespav = gatvesPav;
            Kiekis = kiekis;
        }
    }
}
```

Program.cs

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

namespace U1._6
{
    class Program
    {
        /// <summary>
        /// Įvesties ir išvesties failai priskirti konstantoms
        /// </summary>
        const string CFd = @"duomenys.txt";//pastatų duomenys
        const string CFr = @"M100.csv";//mūrinių virš 100m2 rez
        const string CFg = @"B150.csv";//blokinių virš 150m2 rez
        static void Main(string[] args)
        {
            int senoi = 0;
            double murinium = 100;
            double blokinium = 150;
            string muriniai = "murinis";
            string blokiniai = "blokinis";
            Program programa = new Program();
            List<Namas> namai = programa.FailoSkaitymas();
            List<string> gatves = programa.FiltruotiGatves(namai);
              List<DaugiausiaParduodamu> daugiausiaParduodu = programa.Daugiausia(namai,
              gatves);
            programa.SeniausiasNamas(namai, senoi);
            programa.SeniausioNamoIrasymas(namai, senoi);
            programa.MuriniuNamuIrasymas(namai, muriniai, murinium);
            programa.BlokiniuNamuIrasymas(namai, blokiniai, blokinium);
            programa.Maximumas(daugiausiaParduodu);
        }
        /// <summary>
        /// Nuskaitome ir įvedame failo duomenis
        /// </summary>
        /// <returns>gražina įvestus kintamuosius</returns>
        List<Namas> FailoSkaitymas()//sudaromas sąrašas
        {
            List<Namas> namai = new List<Namas>();

            string[] eilutes = File.ReadAllLines(CFd);
            foreach (string eilute in eilutes)
            {
                //įvedami duomenys
```

```
                string[] skyriklis = eilute.Split(';');
                string rajonas = skyriklis[0];
                string gatve = skyriklis[1];
                double namonr = double.Parse(skyriklis[2]);
                string tipas = skyriklis[3];
                double metai = double.Parse(skyriklis[4]);
                double plotas = double.Parse(skyriklis[5]);
                double kambariai = double.Parse(skyriklis[6]);
                //
              Namas Namas = new Namas(rajonas, gatve, namonr, tipas, metai, plotas,
              kambariai);
                namai.Add(Namas);
            }
            return namai;
        }
        /// <summary>
        /// Suskaičiuojama namų kiekis gatvėje
        /// </summary>
        /// <param name="namai"></param>
        /// <param name="gatves"></param>
        /// <returns>gražina kiekį</returns>
        int Kiek(List<Namas> namai, string gatves)
        {

            int kiekis = 0;
            for (int i = 0; i < namai.Count; i++)
            {
                Namas namas = namai[i];
                if (namas.Gatve == gatves)
                {
                    kiekis++;
                }
            }
            return kiekis;
        }
        /// <summary>
        /// Sufiltruojamos gatvės
        /// </summary>
        /// <param name="namai"></param>
        /// <returns>gražina sufiltruotas gatves</returns>
        List<string> FiltruotiGatves(List<Namas> namai)
        {
            List<string> gatves = new List<string>();
            foreach (Namas namas in namai)
            {
                if (!gatves.Contains(namas.Gatve))
                {
                    gatves.Add(namas.Gatve);
                }
            }
            return gatves;
        }
        /// <summary>
        /// sukuriamas sąrašas parduodamų namų pagal gatves
        /// </summary>
        /// <param name="namai"></param>
        /// <param name="gatves"></param>
        /// <returns>gražina gatves ir ir jų priskirta kiekį</returns>
        List<DaugiausiaParduodamu> Daugiausia(List<Namas> namai, List<string> gatves)
        {
            List<DaugiausiaParduodamu> daugiausiaParduodamu = new
List<DaugiausiaParduodamu>();

            foreach (var gatve in gatves)
            {
                int kiekis = Kiek(namai, gatve);
                DaugiausiaParduodamu rez = new DaugiausiaParduodamu(gatve, kiekis);
                daugiausiaParduodamu.Add(rez);
```

```
            }

            return daugiausiaParduodamu;
        }
        /// <summary>
        /// Randama gatvė kurioje yra daugiausiai namų
        /// </summary>
        /// <param name="daugiausiaParduodamu"></param>
        void Maximumas(List<DaugiausiaParduodamu> daugiausiaParduodamu)
        {
            int maxi = 0;
            string gatvespav = "";
            for (int i = 0; i < daugiausiaParduodamu.Count; i++)
            {
                DaugiausiaParduodamu gatviu = daugiausiaParduodamu[i];
                if (gatviu.Kiekis > maxi)
                {
                    maxi = daugiausiaParduodamu[i].Kiekis;
                    gatvespav = daugiausiaParduodamu[i].Gatvespav;
                }
            }
         Console.WriteLine("Daugiausia parduodamų namų yra: {0} gatvėje: {1}", gatvespav,
       maxi);
        }
        /// <summary>
        /// Randamas seniausias namas
        /// </summary>
        /// <param name="namai"></param>
        /// <param name="senoi"></param>
        void SeniausiasNamas(List<Namas> namai, int senoi)//sukuriamas void'as kuriame
randamas seniausias namas
        {
            double Senas = 2000000;
            for (int i = 0; i < namai.Count; i++)
            {
                if (namai[i].Metai < Senas)
                {
                    Senas = namai[i].Metai;
                    senoi = i;
                }
            }
        }
        /// <summary>
        /// Įrašome į faila mūrinių namų duomenis
        /// </summary>
        /// <param name="namai"></param>
        /// <param name="muriniai"></param>
        /// <param name="murinium"></param>
        void MuriniuNamuIrasymas(List<Namas> namai, string muriniai, double murinium)
        {
            if (File.Exists(CFr))
            {
                File.Delete(CFr);
            }
            for (int i = 0; i < namai.Count; i++)
            {
                if (namai[i].Tipas == muriniai && namai[i].Plotas > murinium)//funkcija jei
yra mūrinis ir turi virš 100m2
                {

                    String simtotextas = String.Format("[{0}] - {1} {2}g, tipas - {3} {4}m
{5}m2 {6} kambariai", namai[i].Rajonas, namai[i].Gatve,
                    namai[i].Namonr, namai[i].Tipas, namai[i].Metai, namai[i].Plotas,
namai[i].Kambariai) + Environment.NewLine;
                    File.AppendAllText(CFr, simtotextas);
                }
            }
        }
```

```
        /// <summary>
        /// Konsolėje parašome seniausio namo informaciją
        /// </summary>
        /// <param name="namai"></param>
        /// <param name="senoi"></param>
       void SeniausioNamoIrasymas(List<Namas> namai, int senoi)//įrašoma į dokumentus bei
       console tekstą
        {
            double amzius = DateTime.Now.Year - namai[senoi].Metai;
              Console.WriteLine("Seniausias namas yra: {0} metų, adresu: {1} {2}g tai yra {3}
              pastatas, kurio plotas {4} m2",
            amzius, namai[senoi].Gatve, namai[senoi].Namonr, namai[senoi].Tipas,
namai[senoi].Plotas);
        }
        /// <summary>
        /// Faile įrašome blokinių namų duomenis
        /// </summary>
        /// <param name="namai"></param>
        /// <param name="blokiniai"></param>
        /// <param name="blokinium"></param>
        void BlokiniuNamuIrasymas(List<Namas> namai, string blokiniai, double blokinium)
        {
            if(File.Exists(CFg))
            {
                File.Delete(CFg);
            }
            for (int i = 0; i < namai.Count; i++)
            {
                if (namai[i].Tipas == blokiniai && namai[i].Plotas > blokinium)//funkcija jei
yra blokinis ir turi virš 150m2
                {

                    String pustextas = String.Format("[{0}] - {1} {2}g, tipas - {3} {4}m
{5}m2 {6} kambariai", namai[i].Rajonas, namai[i].Gatve,
                    namai[i].Namonr, namai[i].Tipas, namai[i].Metai, namai[i].Plotas,
namai[i].Kambariai) + Environment.NewLine;
                    File.AppendAllText(CFg, pustextas);
                }
            }
        }
    }
}
```

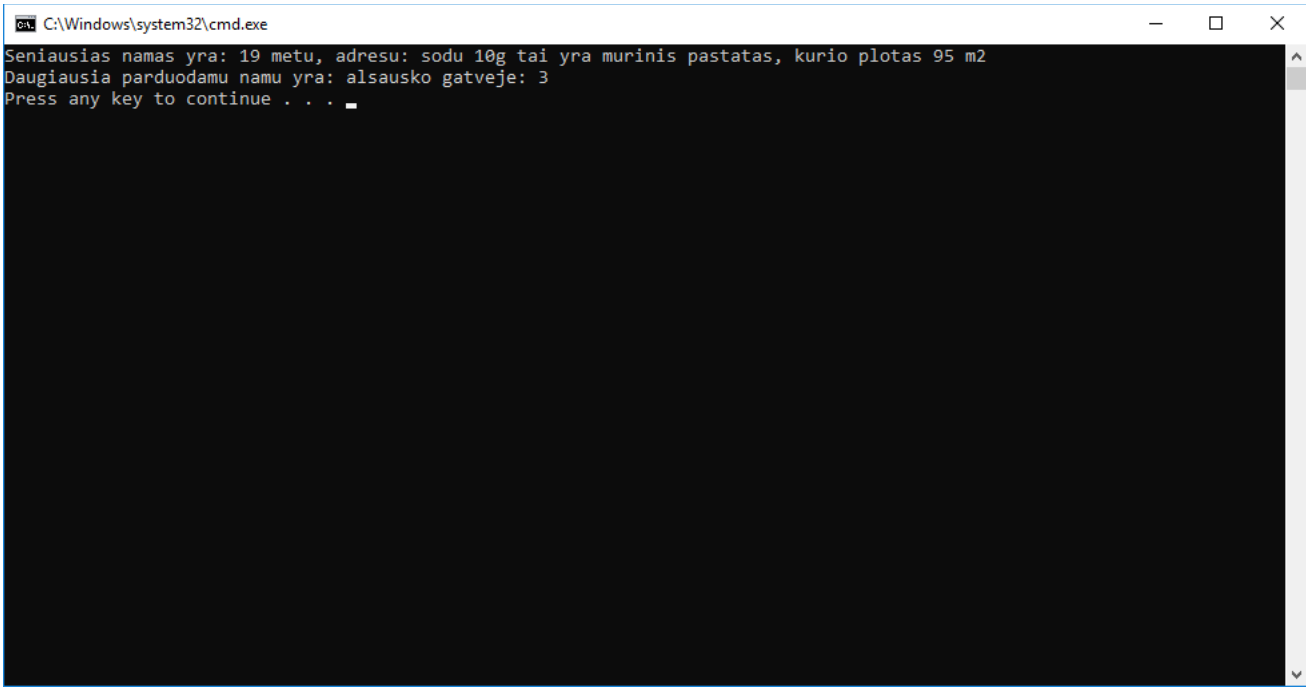
1.3. Pradiniai duomenys ir rezultatai

I - asis testavimo pavyzdys

- duomenys.txt

Dainava;sodu;10;murinis;1999;95;10;
Petrasiunai;vetrunges;18;medinis;2004;180;8;
Silainiai;barono;20;blokinis;2001;174;6;
Dainava;aklasto;5;medinis;2000;115;12;
Silainiai;alsausko;26;blokinis;2011;170;8;
Silainiai;alsausko;2;murinis;2003;180;6;
Silainiai;alsausko;20;blokinis;2005;105;8;
Petrasiunai;vetrunges;19;murinis;2003;160;12;

- Ekrane išvedami duomenys



- M100.csv

[Silainiai] - alsausko 2g, tipas - murinis 2003m 180m2 6 kambariai
[Petrasiunai] - vetrunges 19g, tipas - murinis 2003m 160m2 12 kambariai

- B150.csv

[Silainiai] - barono 20g, tipas - blokinis 2001m 174m2 6 kambariai
[Silainiai] - alsausko 26g, tipas - blokinis 2011m 170m2 8 kambariai

- DuomenysLenteleje.csv

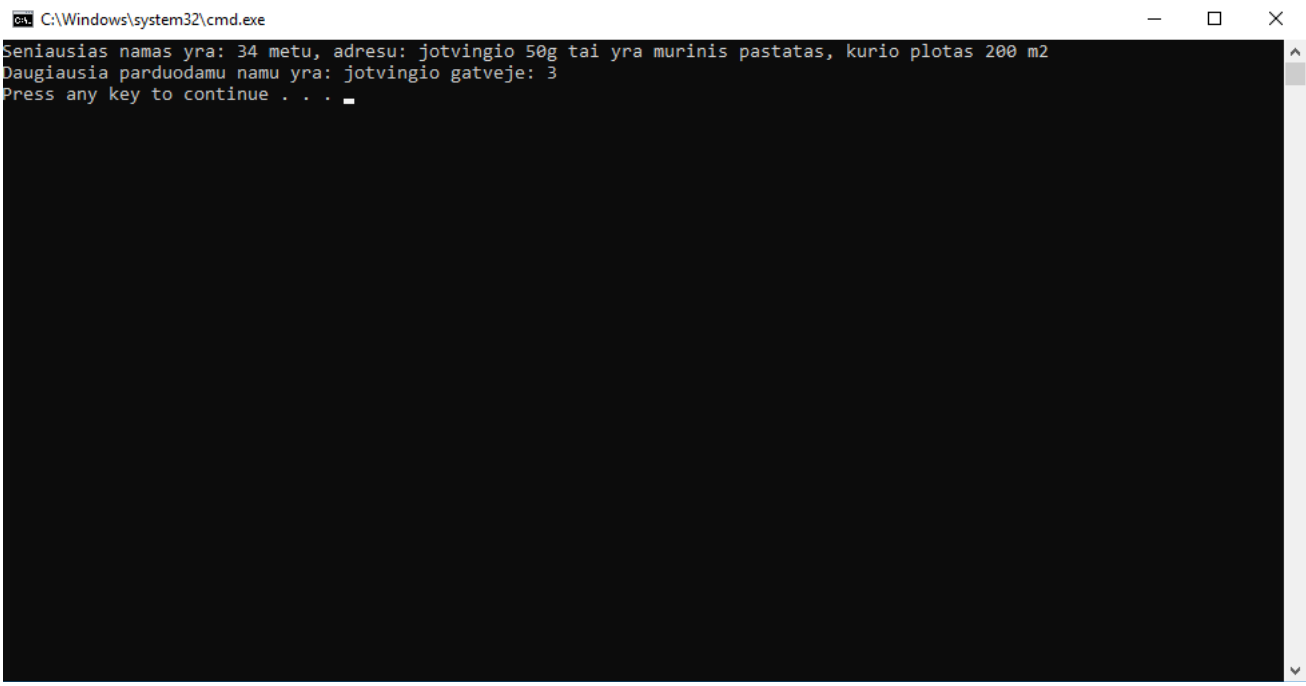
Mikrorajonas	Gatvė	Namo numeris	Tipas	Pastatymo metai	Plotas	Kambarių skaičius
Dainava	sodu	10	murinis	1999	95	10
Petrasiunai	vetrunges	18	medinis	2004	180	8
Silainiai	barono	20	blokinis	2001	174	6
Dainava	aklasto	5	medinis	2000	115	12
Silainiai	alsausko	26	blokinis	2011	170	8
Silainiai	alsausko	2	murinis	2003	180	6
Silainiai	alsausko	20	blokinis	2005	105	8
Petrasiunai	vetrunges	19	murinis	2003	160	12

II – asis testavimo pavyzdys

- duomenys.txt

Dainava;jotvingio;50;murinis;1984;200;10;
Jiesia;kuzmos;85;blokinis;2012;30;1;
Freda;aliejaus;44;medinis;2003;2630;25;
Dainava;jotvingio;35;blokinis;1999;350;10;
Petrasiunai;vetrunges;47;murinis;2003;160;12;
Rokai;jureivio;45;blokinis;1995;130;20;
Freda;aliejaus;65;murinis;2003;2630;25;
Petrasiunai;vetrunges;5;blokinis;2003;160;12;
Dainava;jotvingio;2;blokinis;1999;218;10;

- Ekrane išvedami duomenys



- M100.csv

[Dainava] - jotvingio 50g, tipas - murinis 1984m 200m2 10 kambariai
[Petrasiunai] - vetrunges 47g, tipas - murinis 2003m 160m2 12 kambariai
[Freda] - aliejaus 65g, tipas - murinis 2003m 2630m2 25 kambariai

- B150.csv

[Dainava] - jotvingio 35g, tipas - blokinis 1999m 350m2 10 kambariai
[Petrasiunai] - vetrunges 5g, tipas - blokinis 2003m 160m2 12 kambariai
[Dainava] - jotvingio 2g, tipas - blokinis 1999m 218m2 10 kambariai

- DuomenysLenteleje.csv

Mikrorajonas	Gatvė	Namo numeris	Tipas	Pastatymo metai	Plotas	Kambarių skaičius
Dainava	jotvingio	50	murinis	1984	200	10
Jiesia	kuzmos	85	blokinis	2012	30	1
Freda	aliejaus	44	medinis	2003	2630	25
Dainava	jotvingio	35	blokinis	1999	350	10
Petrasiunai	vetrunges	47	mūrinis	2003	160	12
Rokai	jureivio	45	blokinis	1995	130	20
Freda	aliejaus	65	mūrinis	2003	2200	25
Petrasiunai	vetrunges	5	blokinis	2003	160	12
Dainava	jotvingio	2	blokinis	1999	218	10

1.4. Dėstytojo pastabos

- Blogas ataskaitos failo pavadinimas (ne pdf failas)
- Nesutvarkytas (nespalvotas) programos tekstas
- Nekomentuoti (arba nepilnai pakomentuoti) metodai
- Ne tas šrifto tipas

2. Konteineris

2.1. Darbo užduotis

U2-8. Turistų informacijos centras.

Turizmo informacijos centre perorganizuoti ir atskirai surašyti duomenys apie kiekviename mieste veikiančius muziejus. Keičiasi duomenų formatas. Pirmoje eilutėje – miestas, antroje – atsakingo asmens vardas ir pavardė. Toliau informacija apie muziejus pateikta tokiu pačiu formatu kaip L1 užduotyje, tik nebėra miesto stulpelio.

- Suskaičiuokite, kiek muziejų kiekviename turi gidus, rezultatą atspausdinkite ekrane.
- Raskite, kokio tipo muziejus galima aplankyti kiekviename mieste trečiadieniais, ir atspausdinkite muziejų tipus ekrane.
- Sudarykite Vilniaus ir Kauno muziejų, kurių pavadinimai sutampa, sąrašą ir įrašykite jų duomenis į failą „Sutampa.csv“.
- Sudarykite kiekvieno miesto muziejų, kuriuos galima aplankyti nemokamai, sąrašą, į failus „Nemokami_miestas.csv“ įrašykite muziejaus tipą ir pavadinimą. Jei muziejus dirba tik šeštadieniais ir sekmadieniais, atitinkamoje eilutėje įrašykite „TIK SAVAITGALIAIS“.

2.2. Programos tekstas

2.3. Pradiniai duomenys ir rezultatai

I - asis testavimo pavyzdys

Duomenys.txt

```
Vilnius; Marius Tumosius; Dailės muziejus; Menas; 0; 0; 1; 0; 0; 1; 1; 0,00; taip
Vilnius; Giedrius Levuckas; Gedimino pilies bokšto muziejus; Istorija; 0; 1; 0; 1; 1; 1; 1; 4,00; taip
Vilnius; Martynas Kartiskis; Kraštotyros muziejus; Istorija; 0; 0; 0; 0; 0; 1; 1; 0,00; ne
Vilnius; Jolinta Metelyte; M. K. Čiurlionio dailės galerija; Zoologija; 1; 1; 1; 1; 1; 1; 1; 0,00; ne
Kaunas; Jonas Lumonavicius; Baltų muziejus; Istorija; 0; 1; 1; 1; 1; 0; 0; 3,00; taip
Kaunas; Justina Tijosiene; Vytauto Didžiojo karo muziejus; Istorija; 0; 0; 0; 0; 0; 1; 1; 7,50; taip
Vilnius; Justina Tijosiene; Vytauto Didžiojo karo muziejus; Istorija; 0; 0; 0; 0; 0; 1; 1; 7,50; taip
Vilnius; Kristina Viliutiene; Energetikos ir technikos muziejus; Istorija; 1; 1; 0; 1; 1; 0; 0; 0,00; ne
Kaunas; Jolinta Metelyte; M. K. Čiurlionio dailės galerija; Menas; 1; 1; 1; 1; 1; 1; 1; 0,00; ne
Kaunas; Arbetas Tauravicius; T. Ivanausko zoologijos muziejus; Zoologija; 1; 1; 0; 1; 1; 1; 1; 0,00; taip
Vilnius; Kauras Miriontas; Lietuvos teatro, muzikos ir kino muziejus; Istorija; 0; 1; 1; 1; 1; 0; 0; 2,50;taip
Vilnius; Jaunaras Tauravicius; Jotvario kino muziejus; Zoologija; 0; 0; 0; 0; 0; 1; 1; 0,00; taip
Kaunas; Linas Viturionis; Dailės muziejus; Istorija; 1; 1; 0; 1; 1; 0; 0; 5,00; taip
```

Ekrane išvedami duomenys:

```
Vilnius turi 5 gidus(-a) muziejuose.
Kaunas turi 4 gidus(-a) muziejuose.

Treciadieni galima aplankyti siuose miestuose, siu tipu muziejus:
> Vilnius - Menas
> Vilnius - Zoologija
> Kaunas - Istorija
> Kaunas - Menas
> Vilnius - Istorija
```

Nemokami_Kaunas.csv

```
Menas;M. K. Čiurlionio dailės galerija
Zoologija;T. Ivanausko zoologijos muziejus
```

Nemokami_Vilnius.csv

```
Menas;Dailės muziejus
Istorija;Kraštotyros muziejus;TIK SAVAITGALIAIS
Zoologija;M. K. Čiurlionio dailės galerija
Istorija;Energetikos ir technikos muziejus
Zoologija;Jotvario kino muziejus;TIK SAVAITGALIAIS
```

Sutampa.csv

```
[ Vytauto Didžiojo karo muziejus ]
[ M. K. Čiurlionio dailės galerija ]
[ Dailės muziejus ]
```

Duomenu_Isvedimas.txt

Pavadinimas	Atsakingas asmuo	Miestas	Tipas	Pr	A	T	K	Pn	Š	S	Kaina	Gidas
Dailės muziejus	Marius Tumosius	Vilnius	Menas	Nedirba	Nedirba	Dirba	Nedirba	Nedirba	Dirba	Dirba	0.00	Yra
Gedimino pilies bokšto muziejus	Giedrius Levuckas	Vilnius	Istorija	Nedirba	Dirba	Nedirba	Dirba	Dirba	Dirba	Dirba	4.00	Yra
Kraštotyros muziejus	Martynas Kartiskis	Vilnius	Istorija	Nedirba	Nedirba	Nedirba	Nedirba	Nedirba	Dirba	Dirba	0.00	Nėra
M. K. Čiurlionio dailės galerija	Jolinta Metelyte	Vilnius	Zoologija	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	0.00	Nėra
Baltų muziejus	Jonas Lumonavicius	Kaunas	Istorija	Nedirba	Dirba	Dirba	Dirba	Dirba	Nedirba	Nedirba	3.00	Yra
Vytauto Didžiojo karo muziejus	Justina Tijosiene	Kaunas	Istorija	Nedirba	Nedirba	Nedirba	Nedirba	Nedirba	Dirba	Dirba	7.50	Yra
Vytauto Didžiojo karo muziejus	Justina Tijosiene	Vilnius	Istorija	Nedirba	Nedirba	Nedirba	Nedirba	Nedirba	Dirba	Dirba	7.50	Yra
Energetikos ir technikos muziejus	Kristina Viliutiene	Vilnius	Istorija	Dirba	Dirba	Nedirba	Dirba	Dirba	Nedirba	Nedirba	0.00	Nėra
M. K. Čiurlionio dailės galerija	Jolinta Metelyte	Kaunas	Menas	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	0.00	Nėra
T. Ivanausko zoologijos muziejus	Arbetas Tauravicius	Kaunas	Zoologija	Dirba	Dirba	Nedirba	Dirba	Dirba	Dirba	Dirba	0.00	Yra
Lietuvos teatro, muzikos ir kino muziejus	Kauras Miriontas	Vilnius	Istorija	Nedirba	Dirba	Dirba	Dirba	Dirba	Nedirba	Nedirba	2.50	Yra
Jotvario kino muziejus	Jaunaras Tauravicius	Vilnius	Zoologija	Nedirba	Nedirba	Nedirba	Nedirba	Nedirba	Dirba	Dirba	0.00	Yra
Dailės muziejus	Linas Viturionis	Kaunas	Istorija	Dirba	Dirba	Nedirba	Dirba	Dirba	Nedirba	Nedirba	5.00	Yra

II – asis testavimo pavyzdys

Duomenys2.txt

```
Kaunas; Linas Viturionis; Dailės muziejus; Istorija; 1; 0; 0; 1; 1; 1; 1; 5,00; taip
Kaunas; Jolinta Metelyte; M. K. Čiurlionio dailės galerija; Menas; 1; 1; 1; 1; 1; 0; 0; 1,00; ne
Kaunas; Arbetas Tauravicius; T. Ivanausko zoologijos muziejus; Zoologija; 1; 1; 0; 1; 1; 1; 1; 0,00; taip
Panevezys; Kauras Miriontas; Lietuvos teatro muziejus; Istorija; 0; 1; 1; 1; 1; 1; 1; 0,00; taip
Vilnius; Jaunaras Tauravicius; Jotvario kino muziejus; Zoologija; 0; 1; 0; 0; 0; 0; 0; 0,00; taip
Vilnius; Martynas Kartiskis; Kraštotyros muziejus; Istorija; 0; 0; 1; 0; 0; 1; 1; 3,00; ne
Vilnius; Jolinta Metelyte; M. K. Čiurlionio dailės galerija; Zoologija; 1; 1; 1; 1; 1; 0; 1; 0,00; ne
Kaunas; Jonas Lumonavicius; Baltų muziejus; Istorija; 0; 1; 1; 1; 1; 1; 1; 3,00; taip
Kaunas; Justina Tijosiene; Vytauto Didžiojo karo muziejus; Istorija; 1; 0; 1; 0; 1; 1; 1; 0,00; taip
Vilnius; Justina Tijosiene; Vytauto Didžiojo karo muziejus; Istorija; 1; 0; 0; 0; 1; 0; 1; 7,50; taip
```

Ekrane išvedami duomenys:

```
Kaunas turi 4 gidus(-a) muziejuose.
Panevezys turi 1 gidus(-a) muziejuose.
Vilnius turi 2 gidus(-a) muziejuose.

Treciadieni galima aplankyti siuose miestuose, siu tipu muziejus:
> Kaunas - Menas
> Panevezys - Istorija
> Vilnius - Istorija
> Vilnius - Zoologija
> Kaunas - Istorija
> Kaunas - Istorija
```

Nemokami_Vilnius.csv

```
Zoologija;Jotvario kino muziejus
Zoologija;M. K. Čiurlionio dailės galerija
```

Nemokami_Panevezys.csv

```
Istorija;Lietuvos teatro, muzikos ir kino muziejus
```

Nemokami_Kaunas.csv

```
Zoologija;T. Ivanausko zoologijos muziejus
Istorija;Vytauto Didžiojo karo muziejus
```

Sutampa.csv

```
[ M. K. Čiurlionio dailės galerija ]
[ Vytauto Didžiojo karo muziejus ]
```

Duomenu_Isvedimas.txt

Pavadinimas	Atsakingas asmuo	Miestas	Tipas	Pr	A	T	K	Pn	Š	S	Kaina	Gidas
Dailės muziejus	Linas Viturionis	Kaunas	Istorija	Dirba	Nedirba	Nedirba	Dirba	Dirba	Dirba	Dirba	5.00	Yra
M. K. Čiurlionio dailės galerija	Jolinta Metelyte	Kaunas	Menas	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	Nedirba	1.00	Nėra
T. Ivanausko zoologijos muziejus	Arbetas Tauravicius	Kaunas	Zoologija	Dirba	Dirba	Nedirba	Dirba	Dirba	Dirba	Dirba	0.00	Yra
Lietuvos teatro, muzikos ir kino muziejus	Kauras Miriontas	Panevezys	Istorija	Nedirba	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	0.00	Yra
Jotvario kino muziejus	Jaunaras Tauravicius	Vilnius	Zoologija	Nedirba	Dirba	Nedirba	Nedirba	Nedirba	Nedirba	Nedirba	0.00	Yra
Krašt otyros muziejus	Martynas Kartiskis	Vilnius	Istorija	Nedirba	Nedirba	Dirba	Nedirba	Nedirba	Dirba	Dirba	3.00	Nėra
M. K. Čiurlionio dailės galerija	Jolinta Metelyte	Vilnius	Zoologija	Dirba	Dirba	Dirba	Dirba	Dirba	Nedirba	Dirba	0.00	Nėra
Baltų muziejus	Jonas Lumonavicius	Kaunas	Istorija	Nedirba	Dirba	Dirba	Dirba	Dirba	Dirba	Dirba	3.00	Yra
Vytauto Didžiojo karo muziejus	Justina Tijosiene	Kaunas	Istorija	Dirba	Nedirba	Dirba	Nedirba	Dirba	Dirba	Dirba	0.00	Yra
Vytauto Didžiojo karo muziejus	Justina Tijosiene	Vilnius	Istorija	Dirba	Nedirba	Nedirba	Nedirba	Dirba	Nedirba	Dirba	7.50	Yra

2.4. Dėstytojo pastabos

3. Paveldėjimas

3.1. Darbo užduotis

U3_6. Nekilnojamojo turto agentūra. Turite ir kitų nekilnojamojo turto agentūrų duomenis. Keičiasi duomenų formatas. Pirmoje eilutėje pavadinimas, antroje –adresas, trečioje – telefonas. Nekilnojamojo turto agentūra parduoda butus ir nuosavus namus. Sukurkite klasę „NTObjektas“ (laukai mikrorajonas, gatvė, namo numeris, tipas, pastatymo metai, plotas, kambarių skaičius), kurią paveldės klasės “Butas” (papildomas laukas - aukštas) ir “Namas” (papildomas laukas –šildymo būdas).

- Raskite, kurioje gatvėje daugiausiai parduodamų nekilnojamo turto objektų (namų ir butų), ekrane atspausdinkite gatvės pavadinimą ir parduodamų objektų kiekį.
- Raskite seniausią nekilnojamojo turto objektą, ekrane atspausdinkite visą jo informaciją.
- Raskite, kurių namų ar butų skelbimai yra paskelbti daugiau nei vienoje agentūroje (savininkas tikriausiai labai skuba parduoti, ir bus linkęs nuleisti kainą). Išrikiuokite juos pagal gatvės pavadinimą ir namo numerį. Į failą „Kartojasi.cvs“ įrašykite informaciją apie šiuos objektus.
- Sudarykite visų namų, kurių plotas didesnis nei 100 kv.m., sąrašą, išrikiuokite pagal plotą ir kambarių skaičių ir įrašykite visus duomenis apie šiuos namus į failą „Namas100.csv“. Sudarykite visų butų, kurių plotas didesnis nei 50 kv.m., sąrašą, išrikiuokite pagal plotą ir kambarių skaičių ir įrašykite visus duomenis apie šiuos butus į failą „Butas50.csv“.

3.2. Programos tekstas

Agentura.cs

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

namespace U1._6_2
{
    class Agentura
    {
        public string Pavadinimas { get; set; } // Pavadinimas
        public string Adresas { get; set; } // Adresas
        public string Numeris { get; set; } // Telefono numeris
        public ObjektuKonteineris Butai; // Butų konteineris
        public ObjektuKonteineris Namai; // Namų konteineris

        public Agentura()
        {

        }

        public Agentura(string pavadinimas, string adresas, string numeris)
        {
            Pavadinimas = pavadinimas;
            Adresas = adresas;
            Numeris = numeris;
            Butai = new ObjektuKonteineris();
            Namai = new ObjektuKonteineris();
        }

        public void PridetiButa(Butas butas)
        {
            Butai.PridetiObjekta(butas);
        }

        public void PridetiNama(Namas namas)
        {
            Namai.PridetiObjekta(namas);
```

```
        }
    }
}
```

Butas.cs

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

namespace U1._6_2
{
    class Butas : NTObjektas
    {
              public int Aukstas { get; set; } // buto aukštas

              public Butas(string mikrorajonas, string gatve, string namoNumeris, string
              tipas,
              int pastatymoMetai, int plotas, int kambariuSkaicius, int aukstas) :
              base(mikrorajonas, gatve, namoNumeris, tipas,
              pastatymoMetai, plotas, kambariuSkaicius)
        {
            Aukstas = aukstas;
        }

        /// <summary>
        /// Perrašomas Equals metodas, kuris leidžia patikrinti ar vienodi du pateikti butai
        /// </summary>
        /// <param name="obj"> Lyginamasis objektas </param>
        /// <returns></returns>
        public override bool Equals(object obj)
        {
            Butas s = obj as Butas;
            if (s.Mikrorajonas == Mikrorajonas && s.Gatve == Gatve &&
                s.NamoNumeris == NamoNumeris && s.Tipas == Tipas &&
                s.PastatymoMetai == PastatymoMetai && s.Plotas == Plotas
                && s.KambariuSkaicius == KambariuSkaicius && s.Aukstas == Aukstas)
            {
                return true;
            }
            else
            {
                return false;
            }
        }

        /// <summary>
        /// Dėl pakeisto Equals metodo, pakeičiamas GetHashCode metodas
        /// </summary>
        /// <returns></returns>
        public override int GetHashCode()
        {
              return Gatve.GetHashCode() ^ Mikrorajonas.GetHashCode() ^
              NamoNumeris.GetHashCode() ^
              Tipas.GetHashCode() ^ PastatymoMetai.GetHashCode() ^
              KambariuSkaicius.GetHashCode() ^ Plotas.GetHashCode() ^ Aukstas.GetHashCode();
        }

        public override string ToString()
        {
              return String.Format("Mikrorajonas: {0,-15} Gatvė: {1,-15} Namo numeris: {2,10}
              Tipas: {3,-15} " +
              "Pastatymo metai: {4,10} Plotas: {5,10} Kambarių skaičius: {6,5} Aukštas:
              {7,5}", Mikrorajonas,
              Gatve, NamoNumeris, Tipas, PastatymoMetai, Plotas, KambariuSkaicius, Aukstas);
```

```
        }
    }
}
```

PopuliariausiuGatviuKonteineris.cs

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

namespace U1._6_2
{
    class PopuliariausiuGatviuKonteineris
    {
        public const int DidziausiasGatviuSkaicius = 100; //Didžiausias gatvių skaičius
        public int Kiekis { get; private set; } //Gatvių kiekis
        public string[] PopGatves { get; set; } //Gatvių masyvas

        public PopuliariausiuGatviuKonteineris()
        {
            PopGatves = new string[DidziausiasGatviuSkaicius];
            Kiekis = 0;
        }

        /// <summary>
        /// Prideda gatvę prie masyvo
        /// </summary>
        /// <param name="gatve"> Gatvės pavadinimas </param>
        public void PridetiGatve(string gatve)
        {
            PopGatves[Kiekis++] = gatve;
        }

        /// <summary>
        /// Paima gatvę iš masyvo
        /// </summary>
        /// <param name="indeksas"> Atitinkama vieta masyve </param>
        /// <returns></returns>
        public string GautiGatve(int indeksas)
        {
            return PopGatves[indeksas];
        }

        public bool Contains(string gatve)
        {
            return PopGatves.Contains(gatve);
        }
    }
}
```

ObjektuKonteineris.cs

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

namespace U1._6_2
{
    class ObjektuKonteineris
    {
        private NTObjektas[] Objektai { get; set; } //Masyvas pagal klasės NTObjektai šabloną
        public int Kiekis { get; private set; } //Masyve esančių elementų skaičius

        public ObjektuKonteineris()
```

```
{
    Objektai = new NTObjektas[Program.ObjektuKiekis];
    Kiekis = 0;
}

/// <summary>
/// Pridedamas objektas į masyvą
/// </summary>
/// <param name="objektas"> Pagal šabloną apibūdintas objektas </param>
public void PridetiObjekta(NTObjektas objektas)
{
    Objektai[Kiekis] = objektas;
    Kiekis++;
}

/// <summary>
/// Paimamas objektas iš masyvo pagal nurodytą indeksą
/// </summary>
/// <param name="indeksas"> Elemento vieta masyve </param>
/// <returns></returns>
public NTObjektas GautiObjekta(int indeksas)
{
    return Objektai[indeksas];
}

public bool Contains(NTObjektas objektas)
{
    return Objektai.Contains(objektas);
}

/// <summary>
/// Rikiuoja pasikartojančius objektus pagal gatvę ir namo numerį
/// </summary>
public void RikiuotiPasikartojancius()
{
    for (int i = 0; i < Kiekis - 1; i++)
    {
        for (int j = i + 1; j < Kiekis; j++)
        {
            NTObjektas laikinas = Objektai[i];
            if (GautiObjekta(j) < GautiObjekta(i))
            {
                Objektai[i] = Objektai[j];
                Objektai[j] = laikinas;
            }
        }
    }
}

/// <summary>
/// Rikiuoja objektus pagal plotą ir kambarių skaičių
/// </summary>
public void Rikiuoti()
{
    for (int i = 0; i < Kiekis; i++)
    {
        for (int j = 0; j < Kiekis; j++)
        {
            NTObjektas laikinas = Objektai[i];
             if (GautiObjekta(j).Plotas < GautiObjekta(i).Plotas ||
             (GautiObjekta(j).Plotas == GautiObjekta(i).Plotas &&
             GautiObjekta(j).KambariuSkaicius < GautiObjekta(i).KambariuSkaicius))
            {
                Objektai[i] = Objektai[j];
                Objektai[j] = laikinas;
            }
        }
    }
```

```
        }
    }
}
```

Namas.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Globalization;

namespace U1._6_2
{
    /// <summary>
    /// klasė namo duomenims aprasyti
    /// </summary>
    class Namas : NTObjektas
    {
        public string SildymoBudas { get; set; } // namo šildymo būdas

              public Namas(string mikrorajonas, string gatve, string namoNumeris, string
              tipas,
              int pastatymoMetai, int plotas, int kambariuSkaicius, string sildymoBudas) :
              base(mikrorajonas, gatve, namoNumeris, tipas,
               pastatymoMetai, plotas, kambariuSkaicius)
        {
            SildymoBudas = sildymoBudas;
        }

        /// <summary>
        /// Perrašomas Equals metodas, kuris leidžia patikrinti ar vienodi du pateikti namai
        /// </summary>
        /// <param name="obj"> Lyginamasis objektas </param>
        /// <returns></returns>
        public override bool Equals(object obj)
        {
            Namas s = obj as Namas;
            if (s.Mikrorajonas == Mikrorajonas && s.Gatve == Gatve &&
                s.NamoNumeris == NamoNumeris && s.Tipas == Tipas &&
                s.PastatymoMetai == PastatymoMetai && s.Plotas == Plotas
                && s.KambariuSkaicius == KambariuSkaicius && s.SildymoBudas == SildymoBudas)
            {
                return true;
            }
            else
            {
                return false;
            }
        }

        /// <summary>
        /// Dėl pakeisto Equals metodo, pakeičiamas GetHashCode metodas
        /// </summary>
        /// <returns></returns>
        public override int GetHashCode()
        {
              return Gatve.GetHashCode() ^ Mikrorajonas.GetHashCode() ^
              NamoNumeris.GetHashCode() ^ Tipas.GetHashCode() ^ PastatymoMetai.GetHashCode()
              ^
                KambariuSkaicius.GetHashCode() ^ Plotas.GetHashCode() ^
              SildymoBudas.GetHashCode();
        }

        public override string ToString()
        {
```

```
            return String.Format("Mikrorajonas: {0,-15} Gatvė: {1,-15} Namo numeris: {2,10}
            Tipas: {3,-15} " +
              "Pastatymo metai: {4,10} Plotas: {5,10} Kambarių skaičius: {6,5} Šildymo
              būdas: {7,-15}", Mikrorajonas,
              Gatve, NamoNumeris, Tipas, PastatymoMetai, Plotas, KambariuSkaicius,
              SildymoBudas);
        }
    }
}
```

NTObjektas.cs

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

namespace U1._6_2
{
    class NTObjektas
    {
        public string Mikrorajonas { get; set; } //Mikrorajono pavadinimas
        public string Gatve { get; set; }   //Gatvės pavadinimas
        public string NamoNumeris { get; set; } //Namo numeris
        public string Tipas { get; set; }   //Namo tipas
        public int PastatymoMetai { get; set; } // Namo pastatymo metai
        public int Plotas { get; set; } //Namo plotas
        public int KambariuSkaicius { get; set; } //Name esančių kambarių skaičius

        public NTObjektas()
        {

        }

        public NTObjektas(string mikrorajonas, string gatve, string namoNumeris, string
        tipas,
        int pastatymoMetai, int plotas, int kambariuSkaicius)
        {
            Mikrorajonas = mikrorajonas;
            Gatve = gatve;
            NamoNumeris = namoNumeris;
            Tipas = tipas;
            PastatymoMetai = pastatymoMetai;
            Plotas = plotas;
            KambariuSkaicius = kambariuSkaicius;
        }

        /// <summary>
        /// Perrašomas Equals metodas, kuris leidžia patikrinti ar vienodi du pateikti
       objektai
        /// </summary>
        /// <param name="obj"> Lyginamasis objektas </param>
        /// <returns></returns>
        public override bool Equals(object obj)
        {
            NTObjektas s = obj as NTObjektas;
            if (s.Mikrorajonas == Mikrorajonas && s.Gatve == Gatve &&
                s.NamoNumeris == NamoNumeris && s.Tipas == Tipas &&
                s.PastatymoMetai == PastatymoMetai && s.Plotas == Plotas
                && s.KambariuSkaicius == KambariuSkaicius)
            {
                return true;
            }
            else
            {
                return false;
            }
```

```
        }

        /// <summary>
        /// Dėl pakeisto Equals metodo, pakeičiamas GetHashCode metodas
        /// </summary>
        /// <returns></returns>
        public override int GetHashCode()
        {
       return Gatve.GetHashCode() ^ Mikrorajonas.GetHashCode() ^ NamoNumeris.GetHashCode() ^
       Tipas.GetHashCode() ^ PastatymoMetai.GetHashCode() ^ KambariuSkaicius.GetHashCode() ^
       Plotas.GetHashCode();
        }

        // Palyginimo operatorius, naudojamas rikiuojant pagal gatvę ir namo numerį
        public static bool operator <(NTObjektas objektas1, NTObjektas objektas2)
        {
            if (objektas1.Gatve == objektas2.Gatve)
                return objektas1.NamoNumeris.CompareTo(objektas2.NamoNumeris) < 0;
                else if (String.Compare(objektas1.Gatve, objektas2.Gatve,
              StringComparison.CurrentCulture) < 0)
                return true;
            return false;
        }

        public static bool operator >(NTObjektas objektas1, NTObjektas objektas2)
        {
            if (objektas1.Gatve == objektas2.Gatve)
                return objektas1.NamoNumeris.CompareTo(objektas2.NamoNumeris) > 0;
                else if (String.Compare(objektas1.Gatve, objektas2.Gatve,
              StringComparison.CurrentCulture) > 0)
                return true;
            return false;
        }

        /// <summary>
        /// Perrašytas išspausdinimo šablonas
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
              return String.Format("Mikrorajonas: {0,-15} Gatvė: {1,-15} Namo numeris: {2,10}
              Tipas: {3,-15} " +
              "Pastatymo metai: {4,10} Plotas: {5,10} Kambarių skaičius: {6,5}",
              Mikrorajonas, Gatve,
               NamoNumeris, Tipas, PastatymoMetai, Plotas, KambariuSkaicius);
        }
    }
}
```

Program.cs

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

namespace U1._6_2
{
    class Program
    {
        public const int ObjektuKiekis = 100;
        public const int AgenturuSkaicius = 100;

        static void Main(string[] args)
        {
            Console.OutputEncoding = Encoding.UTF8;
```

```
        Program p = new Program();

        Agentura[] agenturos = new Agentura[Program.AgenturuSkaicius];
        int Kiekis = 0; // Kintamasis, kuris nurodo, kiek yra agentūrų masyve

        //Paima visus csv tipo failus, prasidedančius Namai, iš nurodytos direktyvos
          string[] FailoKelias = Directory.GetFiles(Directory.GetCurrentDirectory(),
          @"Namai*.csv");
        foreach (string kelias in FailoKelias)
        {
            //Nuskaito duomenis. Jei faile yra duomenų tada nuskaito
            if (new FileInfo(kelias).Length != 0)
            {
                agenturos[Kiekis++] = p.Skaityti(kelias);
            }
        }

        //Naujas gatvių konteineris, kuriame yra populiariausios gatvės
        PopuliariausiuGatviuKonteineris populiariosGatves =
        p.DaugiausiaiParduodamu(agenturos, Kiekis);
        Console.WriteLine("Daugiausia {0} namai yra parduodami šiose gatvėse:",
        p.Daugiausia(agenturos, Kiekis));
        p.PopuliariausiosGatvesSpausdinimas(populiariosGatves);

        // Naujas namų konteineris, kuris skaičiuoja seniausius namus
        ObjektuKonteineris SenObjektai = new ObjektuKonteineris();
        SenObjektai = p.SeniausiObjektai(agenturos, Kiekis,
        SeniausiPastatymoMetai(agenturos, Kiekis));
        Console.WriteLine("Seniausi (pastatyti {0} metais) namai:",
        SeniausiPastatymoMetai(agenturos, Kiekis));
        p.SeniausioObjektoIsvedimas(SenObjektai);

        // Nauji namų ir butų konteineris, kuriame yra pasikartojantys objektai
        ObjektuKonteineris NamaiKartojasi = new ObjektuKonteineris();
        NamaiKartojasi = p.KartojasiNamai(agenturos, Kiekis);
        ObjektuKonteineris ButaiKartojasi = new ObjektuKonteineris();
        ButaiKartojasi = p.KartojasiButai(agenturos, Kiekis);
        NamaiKartojasi.RikiuotiPasikartojancius();
        ButaiKartojasi.RikiuotiPasikartojancius();
        p.PasikartojanciuNamuSpausdinimas(NamaiKartojasi, ButaiKartojasi);

        // Namų konteineris, kurių plotai didesni už 100
        ObjektuKonteineris namai = new ObjektuKonteineris();
        namai = p.NamuSarasas(agenturos, Kiekis);
        namai.Rikiuoti();
        p.ObjektuPlotuSpausdinimas(namai, @"Namas100.csv");

        // Butų konteineris, kurių plotai didesni už 50
        ObjektuKonteineris butai = new ObjektuKonteineris();
        butai = p.ButuSaras(agenturos, Kiekis);
        butai.Rikiuoti();
        p.ObjektuPlotuSpausdinimas(butai, @"Butas50.csv");

        //Surašo visus duomenis į lenteles
        for (int i = 0; i < Kiekis; i++)
        {
            string failas = "DuomenysTekstiniame" + (i + 1) + ".txt";
            p.DuomenysLenteleje(agenturos[i], failas);
        }
    }

    /// <summary>
    /// Nuskaito duomenis
    /// </summary>
    /// <param name="failas"> Failas, iš kurio skaitomi duomenys </param>
    /// <returns> Agentūra </returns>
    private Agentura Skaityti(string failas)
```

```
{
    Agentura agentura = null;
    using (StreamReader skaitymas = new StreamReader(@failas, Encoding.UTF8))
    {
        string viskas = null;
        string viskas2 = null;
        string viskas3 = null;
        viskas = skaitymas.ReadLine();
        viskas2 = skaitymas.ReadLine();
        viskas3 = skaitymas.ReadLine();
        if (viskas != null && viskas != null && viskas3 != null)
        {
            agentura = new Agentura(viskas, viskas2, viskas3);
        }
        while (null != (viskas = skaitymas.ReadLine()))
        {
            string[] reiksmes = viskas.Split(',');
            char objektas = viskas[0];
            string mikrorajonas = reiksmes[1];
            string gatve = reiksmes[2];
            string namoNumeris = reiksmes[3];
            string tipas = reiksmes[4];
            int pastatymoMetai = int.Parse(reiksmes[5]);
            int plotas = int.Parse(reiksmes[6]);
            int kambariuSkaicius = int.Parse(reiksmes[7]);
            switch (objektas)
            {
                case 'B':
                    int aukstas = int.Parse(reiksmes[8]);
                    Butas butas = new Butas(mikrorajonas, gatve, namoNumeris, tipas,
                    pastatymoMetai, plotas,
                    kambariuSkaicius, aukstas);
                    agentura.PridetiButa(butas);
                    break;
                case 'N':
                    string sildymoBudas = reiksmes[8];
                    Namas namas = new Namas(mikrorajonas, gatve, namoNumeris, tipas,
                    pastatymoMetai, plotas,
                    kambariuSkaicius, sildymoBudas);
                    agentura.PridetiNama(namas);
                    break;
            }
        }
        return agentura;
    }
}

/// <summary>
/// Randamas parduodamų objektų kiekis tam tikroje gatvėje
/// </summary>
/// <param name="agentura"> Agentūrų masyvas </param>
/// <param name="Kiekis"> Agentūrų kiekis </param>
/// <param name="gatve"> Gatvė </param>
/// <returns> Parduodamų objektų kiekis gatvėje </returns>
private int ParduodamuObjektuKiekisGatveje(Agentura[] agentura, int Kiekis, string
gatve)
{
    int objektuKiekis = 0;
    for (int i = 0; i < Kiekis; i++)
    {
        for (int j = 0; j < agentura[i].Butai.Kiekis; j++)
        {
            Butas obj = agentura[i].Butai.GautiObjekta(j) as Butas;
            if (obj.Gatve == gatve)
            {
                objektuKiekis++;
            }
        }
```

```
            for (int j = 0; j < agentura[i].Namai.Kiekis; j++)
            {
                Namas obj = agentura[i].Namai.GautiObjekta(j) as Namas;
                if (obj.Gatve == gatve)
                {
                    objektuKiekis++;
                }
            }
        }
        return objektuKiekis;
    }

    /// <summary>
    /// Suranda didžiausią parduodamų objektų kiekį gatvėje
    /// </summary>
    /// <param name="agentura"> Agentūrų masyvas </param>
    /// <param name="Kiekis"> Agentūrų kiekis </param>
    /// <returns> Didžiausias parduodamų objektų kiekis gatvėje </returns>
    private int Daugiausia(Agentura[] agentura, int Kiekis)
    {
        int daugiausia = 0;
        int parduodama;
        for (int i = 0; i < Kiekis; i++)
        {
            for (int j = 0; j < agentura[i].Namai.Kiekis; j++)
            {
                Namas obj = agentura[i].Namai.GautiObjekta(j) as Namas;
                parduodama = ParduodamuObjektuKiekisGatveje(agentura, Kiekis, obj.Gatve);
                if (parduodama > daugiausia)
                {
                    daugiausia = parduodama;
                }
            }
            for (int j = 0; j < agentura[i].Butai.Kiekis; j++)
            {
                Butas obj = agentura[i].Butai.GautiObjekta(j) as Butas;
                parduodama = ParduodamuObjektuKiekisGatveje(agentura, Kiekis, obj.Gatve);
                if (parduodama > daugiausia)
                {
                    daugiausia = parduodama;
                }
            }
        }
        return daugiausia;
    }

    /// <summary>
    /// Sudaro populiariausių gatvių konteinerį
    /// </summary>
    /// <param name="agentura"> Agentūrų masyvas </param>
    /// <param name="Kiekis"> Agentūrų kiekis </param>
    /// <returns></returns>
    private PopuliariausiuGatviuKonteineris DaugiausiaiParduodamu(Agentura[] agentura, int
    Kiekis)
    {
          PopuliariausiuGatviuKonteineris parduodama = new
          PopuliariausiuGatviuKonteineris();
        for (int i = 0; i < Kiekis; i++)
        {
            for (int j = 0; j < agentura[i].Butai.Kiekis; j++)
            {
                Butas obj = agentura[i].Butai.GautiObjekta(j) as Butas;
                 if (ParduodamuObjektuKiekisGatveje(agentura, Kiekis, obj.Gatve) ==
                 Daugiausia(agentura, Kiekis))
                {
                    if (!parduodama.Contains(obj.Gatve))
                    {
                        parduodama.PridetiGatve(obj.Gatve);
```

```
                }
            }
        }
        for (int j = 0; j < agentura[i].Namai.Kiekis; j++)
        {
             Namas obj = agentura[i].Namai.GautiObjekta(j) as Namas;
             if (ParduodamuObjektuKiekisGatveje(agentura, Kiekis, obj.Gatve) ==
             Daugiausia(agentura, Kiekis))
            {
                if (!parduodama.Contains(obj.Gatve))
                {
                    parduodama.PridetiGatve(obj.Gatve);
                }
            }
        }
    }
    return parduodama;
}

/// <summary>
/// Spausdina populiariausias gatves į ekraną
/// </summary>
/// <param name="PopGatves"> Populiariausių gatvių konteineris </param>
private void PopuliariausiosGatvesSpausdinimas(PopuliariausiuGatviuKonteineris
PopGatves)
{
    for (int i = 0; i < PopGatves.Kiekis; i++)
    {
        Console.WriteLine("{0} gatvėje", PopGatves.GautiGatve(i));
    }
}

/// <summary>
/// Suranda metus, kada buvo pastatytas seniausias objektas
/// </summary>
/// <param name="agenturos"> Agentūrų masyvas </param>
/// <param name="Kiekis"> Agentūrų skaičius </param>
/// <returns> Seniausi objekto pastatymo metai </returns>
private static int SeniausiPastatymoMetai(Agentura[] agenturos, int Kiekis)
{
    int seniausiMetai = 10000;
    for (int i = 0; i < Kiekis; i++)
    {
        for (int j = 0; j < agenturos[i].Namai.Kiekis; j++)
        {
            Namas namas = agenturos[i].Namai.GautiObjekta(j) as Namas;
            if (namas.PastatymoMetai <= seniausiMetai)
            {
                seniausiMetai = namas.PastatymoMetai;
            }
        }
        for (int j = 0; j < agenturos[i].Butai.Kiekis; j++)
        {
            Butas butas = agenturos[i].Butai.GautiObjekta(j) as Butas;
            if (butas.PastatymoMetai <= seniausiMetai)
            {
                seniausiMetai = butas.PastatymoMetai;
            }
        }
    }
    return seniausiMetai;
}


/// <summary>
/// Į masyvą sudedami objektai, kurių pastatymo metai sutampa su seniausiais metais
/// </summary>
/// <param name="agenturos"> Agentūrų masyvas </param>
```

```
 /// <param name="Kiekis"> Agentūrų skaičius </param>
 /// <param name="SeniausiMetai"> Seniausi pastatymo metai </param>
 /// <returns> Seniausių objektų konteineris </returns>
private ObjektuKonteineris SeniausiObjektai(Agentura[] agenturos, int Kiekis, int
SeniausiMetai)
 {
     ObjektuKonteineris senObjektai = new ObjektuKonteineris();
     for (int i = 0; i < Kiekis; i++)
     {
         Agentura agentura = agenturos[i];
         for (int j = 0; j < agentura.Namai.Kiekis; j++)
         {
             Namas namas = agentura.Namai.GautiObjekta(j) as Namas;
             if (namas.PastatymoMetai.Equals(SeniausiMetai))
             {
                 if (!senObjektai.Contains(agentura.Namai.GautiObjekta(j)))
                 {
                     senObjektai.PridetiObjekta(agentura.Namai.GautiObjekta(j));
                 }
             }
         }
         for (int j = 0; j < agentura.Butai.Kiekis; j++)
         {
             Butas butas = agentura.Butai.GautiObjekta(j) as Butas;
             if (butas.PastatymoMetai.Equals(SeniausiMetai))
             {
                 if (!senObjektai.Contains(agentura.Butai.GautiObjekta(j)))
                 {
                     senObjektai.PridetiObjekta(agentura.Butai.GautiObjekta(j));
                 }
             }
         }
     }
     return senObjektai;
 }

 /// <summary>
 /// Išveda seniausius namus
 /// </summary>
 /// <param name="senNamai"> Seniausių namų masyvas </param>
 private void SeniausioObjektoIsvedimas(ObjektuKonteineris senObjektai)
 {
     for (int i = 0; i < senObjektai.Kiekis; i++)
     {
         Namas namas = senObjektai.GautiObjekta(i) as Namas;
         Butas butas = senObjektai.GautiObjekta(i) as Butas;
         if (namas != null)
              Console.WriteLine("Namo amžius: {0}, adresas: {1} gatvė - {2} numeris,
              tipas: {3}, plotas: {4} kv/m, šildymo būdas: {5}", 2018 -
              namas.PastatymoMetai,
             namas.Gatve, namas.NamoNumeris, namas.Tipas, namas.Plotas,
             namas.SildymoBudas);
         else
              Console.WriteLine("Namo amžius: {0}, adresas: {1} gatvė - {2} numeris,
              tipas: {3}, plotas: {4} kv/m, aukštas: {5}", 2018 -
              butas.PastatymoMetai,
              butas.Gatve, butas.NamoNumeris, butas.Tipas, butas.Plotas,
              butas.Aukstas);

     }
 }

 /// <summary>
 /// Sudaro pasikartojančių namų konteinerį
 /// </summary>
 /// <param name="agenturos"> Agentūrų konteineris </param>
 /// <param name="Kiekis"> Agentūrų kiekis </param>
 /// <returns> Pasikartojančių namų konteineris </returns>
```

```
private ObjektuKonteineris KartojasiNamai(Agentura[] agenturos, int Kiekis)
{
    ObjektuKonteineris pasikartoja = new ObjektuKonteineris();
    for (int i = 0; i < Kiekis; i++)
    {
        Agentura agentura = agenturos[i];
        for (int j = 0; j < agentura.Namai.Kiekis; j++)
        {
            Namas namas = agentura.Namai.GautiObjekta(j) as Namas;
            for (int k = i + 1; k < Kiekis; k++)
            {
                Agentura agentura2 = agenturos[k];
                for (int g = 0; g < agentura2.Namai.Kiekis; g++)
                {
                    Namas namas2 = agentura2.Namai.GautiObjekta(g) as Namas;
                    if (namas.Equals(namas2))
                    {
                        if (!pasikartoja.Contains(namas))
                        {
                            pasikartoja.PridetiObjekta(namas);
                        }
                    }
                }
            }
        }
    }
    return pasikartoja;
}

/// <summary>
/// Sudaro pasikartojančių butų konteinerinį
/// </summary>
/// <param name="agenturos"> Agentūrų masyvas </param>
/// <param name="Kiekis"> Agentūrų kiekis </param>
/// <returns> Pasikartojančių butų konteineris </returns>
private ObjektuKonteineris KartojasiButai(Agentura[] agenturos, int Kiekis)
{
    ObjektuKonteineris pasikartoja = new ObjektuKonteineris();
    for (int i = 0; i < Kiekis; i++)
    {
        Agentura agentura = agenturos[i];
        for (int j = 0; j < agentura.Butai.Kiekis; j++)
        {
            Butas butas = agentura.Butai.GautiObjekta(j) as Butas;
            for (int k = i + 1; k < Kiekis; k++)
            {
                Agentura agentura2 = agenturos[k];
                for (int g = 0; g < agentura2.Butai.Kiekis; g++)
                {
                    Butas butas2 = agentura2.Butai.GautiObjekta(g) as Butas;
                    if (butas.Equals(butas2))
                    {
                        if (!pasikartoja.Contains(butas))
                        {
                            pasikartoja.PridetiObjekta(butas);
                        }
                    }
                }
            }
        }
    }
    return pasikartoja;
}

/// <summary>
/// Pasikartojančių objektų spausdinimas
/// </summary>
/// <param name="PasikartojantysNamai"> Pasikartojančių namų konteineris </param>
```

```
 /// <param name="PasikartojantysButai"> Pasikartojančių butų konteineris </param>
private void PasikartojanciuNamuSpausdinimas(ObjektuKonteineris PasikartojantysNamai,
ObjektuKonteineris PasikartojantysButai)
 {
       using (StreamWriter rasyti = new StreamWriter(@"Kartojasi.csv", false,
       Encoding.UTF8))
     {
         if (PasikartojantysButai.Kiekis == 0 && PasikartojantysNamai.Kiekis == 0)
         {
             Console.WriteLine("Pasikartojančių objektų nėra");
         }
         else
         {
             for (int i = 0; i < PasikartojantysNamai.Kiekis; i++)
             {
                 Namas namas = PasikartojantysNamai.GautiObjekta(i) as Namas;
                 rasyti.WriteLine(namas.ToString());
             }
             for (int i = 0; i < PasikartojantysButai.Kiekis; i++)
             {
                 Butas butas = PasikartojantysButai.GautiObjekta(i) as Butas;
                 rasyti.WriteLine(butas.ToString());
             }
         }
     }
 }

 /// <summary>
 /// Sudaro namų konteinerį, kurių plotas didesnis už 100 kv. m.
 /// </summary>
 /// <param name="agenturos"> Agentūrų konteineris </param>
 /// <param name="Kiekis"> Agentūrų kiekis </param>
 /// <returns> Namų konteineris </returns>
 private ObjektuKonteineris NamuSarasas(Agentura[] agenturos, int Kiekis)
 {
     ObjektuKonteineris namai = new ObjektuKonteineris();
     for (int i = 0; i < Kiekis; i++)
     {
         Agentura agentura = agenturos[i];
         for (int g = 0; g < agentura.Namai.Kiekis; g++)
         {
             Namas namas = agentura.Namai.GautiObjekta(g) as Namas;
             if (namas != null && namas.Plotas > 100)
             {
                 if (!namai.Contains(namas))
                 {
                     namai.PridetiObjekta(namas);
                 }
             }
         }
     }
     return namai;
 }

 /// <summary>
 /// Sudaro butų konteinerį, kurių plotas didesnis už 50 kv. m.
 /// </summary>
 /// <param name="agenturos"> Agentūrų konteineris </param>
 /// <param name="Kiekis"> Agentūrų kiekis </param>
 /// <returns> Butų konteineris </returns>
 private ObjektuKonteineris ButuSaras(Agentura[] agenturos, int Kiekis)
 {
     ObjektuKonteineris butai = new ObjektuKonteineris();
     for (int i = 0; i < Kiekis; i++)
     {
         Agentura agentura = agenturos[i];
         for (int g = 0; g < agentura.Butai.Kiekis; g++)
         {
```

```
            Butas butas = agentura.Butai.GautiObjekta(g) as Butas;
            if (butas != null && butas.Plotas > 50)
            {
                if (!butai.Contains(butas))
                {
                    butai.PridetiObjekta(butas);
                }
            }
        }
    }
    return butai;
}

/// <summary>
/// Objektų, atrinktų pagal plotus, spausdinimas
/// </summary>
/// <param name="objektai"> Objektų konteineris </param>
/// <param name="file"> Failo pavadinimas </param>
private void ObjektuPlotuSpausdinimas(ObjektuKonteineris objektai, string file)
{
    using (StreamWriter rasyti = new StreamWriter(file, false, Encoding.UTF8))
    {
        if (objektai.Kiekis == 0)
        {
            Console.WriteLine("Tokių objektų nėra");
        }
        else
        {
            for (int i = 0; i < objektai.Kiekis; i++)
            {
                Butas butas = objektai.GautiObjekta(i) as Butas;
                Namas namas = objektai.GautiObjekta(i) as Namas;

                if (butas != null)
                    rasyti.WriteLine(butas.ToString());
                else
                    rasyti.WriteLine(namas.ToString());
            }
        }
    }
}

private void DuomenysLenteleje(Agentura agentura, string failas)
{
    using (StreamWriter rasyti = new StreamWriter(@failas, false, Encoding.UTF8))
    {

        rasyti.WriteLine("Duomenys apie agentūras ir jų parduodamus objektus");
        rasyti.WriteLine(new String('-', 135));
        rasyti.WriteLine(" Agentūros pavadinimas: {0, -74} ", agentura.Pavadinimas);
        rasyti.WriteLine(" Agentūros adresas: {0, -78} ", agentura.Adresas);
        rasyti.WriteLine(" Agentūros telefono numeris: {0, -69} ", agentura.Numeris);
        rasyti.WriteLine(new String('-', 135));

        rasyti.WriteLine("Namai:");
        rasyti.WriteLine(new String('-', 135));
       rasyti.WriteLine("| {0, -15} | {1, -15} | {2, -10} | {3, 10} | {4, 15} | {5,
    10} | {6, 20} | {7,-15}", "Mikrorajonas",
    "Gatvė", "Namo Nr.", "Tipas", "Pastatymo metai", "Plotas", "Kambarių skaičius",
    "Šildymo būdas");
        rasyti.WriteLine(new String('-', 135));

        for (int i = 0; i < agentura.Namai.Kiekis; i++)
        {
             Namas obj = agentura.Namai.GautiObjekta(i) as Namas;
            rasyti.WriteLine("| {0, -15} | {1, -15} | {2, -10} | {3, 10} | {4, 15} |
            {5, 10} | {6, 20} | {7,-15}",
```

```
                    obj.Mikrorajonas, obj.Gatve, obj.NamoNumeris, obj.Tipas,
                    obj.PastatymoMetai, obj.Plotas,
                    obj.KambariuSkaicius, obj.SildymoBudas);
                    rasyti.WriteLine(new String('-', 135));
                }

                rasyti.WriteLine("Butai:");
                rasyti.WriteLine(new String('-', 135));
                rasyti.WriteLine("| {0, -15} | {1, -15} | {2, -10} | {3, 10} | {4, 15} | {5,
            10} | {6, 20} | {7,5}", "Mikrorajonas",
            "Gatvė", "Namo Nr.", "Tipas", "Pastatymo metai", "Plotas", "Kambarių skaičius",
            "Aukštas");
                rasyti.WriteLine(new String('-', 135));

                for (int i = 0; i < agentura.Butai.Kiekis; i++)
                {
                    Butas obj = agentura.Butai.GautiObjekta(i) as Butas;
                    rasyti.WriteLine("| {0, -15} | {1, -15} | {2, -10} | {3, 10} | {4, 15} |
                    {5, 10} | {6, 20} | {7,5}",
                    obj.Mikrorajonas, obj.Gatve, obj.NamoNumeris, obj.Tipas,
                    obj.PastatymoMetai, obj.Plotas,
                    obj.KambariuSkaicius, obj.Aukstas);
                    rasyti.WriteLine(new String('-', 135));
                }
            }
        }
    }
}
```

3.3. Pradiniai duomenys ir rezultatai

Pirmas testas:

Namai1.csv

```
UAB "Ponai"
Senukų gatvė 13, Kaunas
+37056952919
B,Saulėtekis,Šeimyniškių,15,Blokinis,1980,50,3,5
N,Saulėtekis,Konstitucijos,124,Medinis,1975,100,6,Dujinis
B,Pilaitė,Gėlių,222,Mūrinis,1899,32,2,15
B,Šeškinė,Tulpių,64,Mūrinis,2000,65,4,9
N,Saulėtekis,Tuskulėnų,81,Mūrinis,2005,90,4,Elektra
N,Žirmūnai,Sodų,34A,Medinis,1946,120,7,Kietasis kuras
B,Pilaitė,Ivanausko,245,Mūrinis,1993,42,2,7
N,Šeškinė,Saulės,98B,Blokinis,1990,260,9,Kietasis kuras
B,Saulėtekis,Minties,24,Mūrinis,1995,49,2,12
N,Žirmūnai,Karalių,75,Medinis,2001,250,8,Elektra
N,Pilaitė,Kmynų,63,Medinis,2004,150,7,Dujinis
N,Šeškinė,Eglių,38,Mūrinis,2009,180,8,Dujinis
N,Saulėtekis,Lvovo,94,Blokinis,1979,145,6,Kietasis kuras
N,Žirmūnai,Lašo,72,Medinis,1980,71,3,Elektra
B,Pilaitė,Kviečių,34,Mūrinis,2010,43,2,8
B,Šeškinė,Šiaulių,46,Mūrinis,2012,29,1,1
B,Saulėtekis,Konstitucijos,85,Mūrinis,1975,81,5,17
B,Saulėtekis,Konstitucijos,115,Mūrinis,1975,81,5,17
```

Namai2.csv

```
UAB "Dragai"
Persų gatvė 13; Kaunas
+378451541
N,Žirmūnai,Karalių,75,Medinis,1900,250,8,Dujinis
B,Pilaitė,Kmynų,63,Medinis,2004,50,2,5
B,Saulėtekis,Lvovo,94,Blokinis,1979,27,1,48
B,Žirmūnai,Lašo,72,Medinis,1980,90,5,1
```

```
B,Šeškinė,Šiaulių,46,Mūrinis,2012,80,4,6
B,Saulėtekis,Konstitucijos,85,Mūrinis,1975,39,2,25
B,Šeškinė,Eglių,38,Mūrinis,2009,30,1,13
B,Saulėtekis,Minties,24,Mūrinis,1995,75,4,4
B,Pilaitė,Kviečių,34,Mūrinis,2010,54,2,20
B,Saulėtekis,Konstitucijos,115,Mūrinis,1899,35,1,41
N,Pilaitė,Ivanausko,222,Medinis,1990,120,7,Dujinis
B,Žirmūnai,Lašo,72,Medinis,1980,63,3,31
B,Saulėtekis,Konstitucijos,85,Mūrinis,1975,81,5,17
```

Namai3.csv

```
UAB "Persai"
Džiaulių gatvė 3; Kaunas
8692561611
N,Dainava,Taikos,42,Medinis,1920,190,8,Dujinis
N,Vilijampolė,Savanorių,124,Medinis,1975,100,6,Elektrinis
B,Šeškinė,Tulpių,64,Mūrinis,2000,20,1,5
B,Saulėtekis,Tuskulėnų,81,Mūrinis,2005,90,4,1
B,Dainava,Taikos,34A,Mūrinis,1946,80,5,12
N,Šeškinė,Saulės,98B,Medinis,1990,260,9,Elektrinis
B,Saulėtekis,Minties,24,Mūrinis,1995,37,2,4
N,Žirmūnai,Karalių,75,Medinis,1900,250,8,Dujinis
B,Pilaitė,Ivanausko,63,Mūrinis,2004,50,2,3
N,Pilaitė,Ivanausko,222,Medinis,1990,120,7,Dujinis
N,Pilaitė,Ivanausko,245,Medinis,1920,150,6,Dujinis
N,Pilaitė,Ivanausko,201,Medinis,1980,90,4,Dujinis
```

Kartojasi.csv

```
Mikrorajonas: Pilaitė         Gatvė: Ivanausko       Namo numeris:        222
Tipas: Medinis         Pastatymo metai:       1990 Plotas:        120 Kambarių
skaičius:     7 Šildymo būdas: Dujinis
Mikrorajonas: Žirmūnai        Gatvė: Karalių         Namo numeris:         75
Tipas: Medinis         Pastatymo metai:       1900 Plotas:        250 Kambarių
skaičius:     8 Šildymo būdas: Dujinis
Mikrorajonas: Saulėtekis      Gatvė: Konstitucijos   Namo numeris:         85
Tipas: Mūrinis         Pastatymo metai:       1975 Plotas:         81 Kambarių
skaičius:     5 Aukštas:    17
```

Namas100.csv

```
Mikrorajonas: Šeškinė         Gatvė: Saulės          Namo numeris:        98B
Tipas: Blokinis        Pastatymo metai:       1990 Plotas:        260 Kambarių
skaičius:     9 Šildymo būdas: Kietasis kuras
Mikrorajonas: Šeškinė         Gatvė: Saulės          Namo numeris:        98B
Tipas: Medinis         Pastatymo metai:       1990 Plotas:        260 Kambarių
skaičius:     9 Šildymo būdas: Elektrinis
Mikrorajonas: Žirmūnai        Gatvė: Karalių         Namo numeris:         75
Tipas: Medinis         Pastatymo metai:       1900 Plotas:        250 Kambarių
skaičius:     8 Šildymo būdas: Dujinis
Mikrorajonas: Žirmūnai        Gatvė: Karalių         Namo numeris:         75
Tipas: Medinis         Pastatymo metai:       2001 Plotas:        250 Kambarių
skaičius:     8 Šildymo būdas: Elektra
Mikrorajonas: Dainava         Gatvė: Taikos          Namo numeris:         42
Tipas: Medinis         Pastatymo metai:       1920 Plotas:        190 Kambarių
skaičius:     8 Šildymo būdas: Dujinis
Mikrorajonas: Šeškinė         Gatvė: Eglių           Namo numeris:         38
Tipas: Mūrinis         Pastatymo metai:       2009 Plotas:        180 Kambarių
skaičius:     8 Šildymo būdas: Dujinis
Mikrorajonas: Pilaitė         Gatvė: Kmynų           Namo numeris:         63
Tipas: Medinis         Pastatymo metai:       2004 Plotas:        150 Kambarių
skaičius:     7 Šildymo būdas: Dujinis
Mikrorajonas: Pilaitė         Gatvė: Ivanausko       Namo numeris:        245
Tipas: Medinis         Pastatymo metai:       1920 Plotas:        150 Kambarių
skaičius:     6 Šildymo būdas: Dujinis
Mikrorajonas: Saulėtekis      Gatvė: Lvovo           Namo numeris:         94
Tipas: Blokinis        Pastatymo metai:       1979 Plotas:        145 Kambarių
skaičius:     6 Šildymo būdas: Kietasis kuras
```

Mikrorajonas: Pilaitė Gatvė: Ivanausko Namo numeris: 222
Tipas: Medinis Pastatymo metai: 1990 Plotas: 120 Kambarių skaičius: 7 Šildymo būdas: Dujinis
Mikrorajonas: Žirmūnai Gatvė: Sodų Namo numeris: 34A
Tipas: Medinis Pastatymo metai: 1946 Plotas: 120 Kambarių skaičius: 7 Šildymo būdas: Kietasis kuras

Butas50.csv

Mikrorajonas: Žirmūnai Gatvė: Lašo Namo numeris: 72
Tipas: Medinis Pastatymo metai: 1980 Plotas: 90 Kambarių skaičius: 5 Aukštas: 1
Mikrorajonas: Saulėtekis Gatvė: Tuskulėnų Namo numeris: 81
Tipas: Mūrinis Pastatymo metai: 2005 Plotas: 90 Kambarių skaičius: 4 Aukštas: 1
Mikrorajonas: Saulėtekis Gatvė: Konstitucijos Namo numeris: 85
Tipas: Mūrinis Pastatymo metai: 1975 Plotas: 81 Kambarių skaičius: 5 Aukštas: 17
Mikrorajonas: Saulėtekis Gatvė: Konstitucijos Namo numeris: 115
Tipas: Mūrinis Pastatymo metai: 1975 Plotas: 81 Kambarių skaičius: 5 Aukštas: 17
Mikrorajonas: Dainava Gatvė: Taikos Namo numeris: 34A
Tipas: Mūrinis Pastatymo metai: 1946 Plotas: 80 Kambarių skaičius: 5 Aukštas: 12
Mikrorajonas: Šeškinė Gatvė: Šiaulių Namo numeris: 46
Tipas: Mūrinis Pastatymo metai: 2012 Plotas: 80 Kambarių skaičius: 4 Aukštas: 6
Mikrorajonas: Saulėtekis Gatvė: Minties Namo numeris: 24
Tipas: Mūrinis Pastatymo metai: 1995 Plotas: 75 Kambarių skaičius: 4 Aukštas: 4
Mikrorajonas: Šeškinė Gatvė: Tulpių Namo numeris: 64
Tipas: Mūrinis Pastatymo metai: 2000 Plotas: 65 Kambarių skaičius: 4 Aukštas: 9
Mikrorajonas: Žirmūnai Gatvė: Lašo Namo numeris: 72
Tipas: Medinis Pastatymo metai: 1980 Plotas: 63 Kambarių skaičius: 3 Aukštas: 31
Mikrorajonas: Pilaitė Gatvė: Kviečių Namo numeris: 34
Tipas: Mūrinis Pastatymo metai: 2010 Plotas: 54 Kambarių skaičius: 2 Aukštas: 20

DuomenysTekstiniame1.txt

Duomenys apie agentūras ir jų parduodamus objektus

Agentūros pavadinimas: UAB "Ponai"
Agentūros adresas: Senukų gatvė 13, Kaunas
Agentūros telefono numeris: +37056952919

Namai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Šildymo būdas
Saulėtekis	Konstitucijos	124	Medinis	1975	100	6	Dujinis
Saulėtekis	Tuskulėnų	81	Mūrinis	2005	90	4	Elektra
Žirmūnai	Sodų	34A	Medinis	1946	120	7	Kietasis kuras
Šeškinė	Saulės	98B	Blokinis	1990	260	9	Kietasis kuras
Žirmūnai	Karalių	75	Medinis	2001	250	8	Elektra
Pilaitė	Kmynų	63	Medinis	2004	150	7	Dujinis
Šeškinė	Eglių	38	Mūrinis	2009	180	8	Dujinis
Saulėtekis	Lvovo	94	Blokinis	1979	145	6	Kietasis kuras
Žirmūnai	Lašo	72	Medinis	1980	71	3	Elektra

Butai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Aukštas
Saulėtekis	Šeimyniškių	15	Blokinis	1980	50	3	5
Pilaitė	Gėlių	222	Mūrinis	1899	32	2	15
Šeškinė	Tulpių	64	Mūrinis	2000	65	4	9
Pilaitė	Ivanausko	245	Mūrinis	1993	42	2	7
Saulėtekis	Minties	24	Mūrinis	1995	49	2	12
Pilaitė	Kviečių	34	Mūrinis	2010	43	2	8
Šeškinė	Šiaulių	46	Mūrinis	2012	29	1	1
Saulėtekis	Konstitucijos	85	Mūrinis	1975	81	5	17
Saulėtekis	Konstitucijos	115	Mūrinis	1975	81	5	17

DuomenysTekstiniame2.txt

Duomenys apie agentūras ir jų parduodamus objektus

Agentūros pavadinimas: UAB "Dragai"
Agentūros adresas: Persų gatvė 13; Kaunas
Agentūros telefono numeris: +378451541

Namai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Šildymo būdas
Žirmūnai	Karalių	75	Medinis	1900	250	8	Dujinis
Pilaitė	Ivanausko	222	Medinis	1990	120	7	Dujinis

Butai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Aukštas
Pilaitė	Kmynų	63	Medinis	2004	50	2	5
Saulėtekis	Lvovo	94	Blokinis	1979	27	1	48
Žirmūnai	Lašo	72	Medinis	1980	90	5	1
Šeškinė	Šiaulių	46	Mūrinis	2012	80	4	6
Saulėtekis	Konstitucijos	85	Mūrinis	1975	39	2	25
Šeškinė	Eglių	38	Mūrinis	2009	30	1	13
Saulėtekis	Minties	24	Mūrinis	1995	75	4	4
Pilaitė	Kviečių	34	Mūrinis	2010	54	2	20
Saulėtekis	Konstitucijos	115	Mūrinis	1899	35	1	41
Žirmūnai	Lašo	72	Medinis	1980	63	3	31
Saulėtekis	Konstitucijos	85	Mūrinis	1975	81	5	17

DuomenysTekstiniame3.txt

Duomenys apie agentūras ir jų parduodamus objektus

Agentūros pavadinimas: UAB "Persai"
Agentūros adresas: Džiaulių gatvė 3; Kaunas
Agentūros telefono numeris: 8692561611

Namai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Šildymo būdas
Dainava	Taikos	42	Medinis	1920	190	8	Dujinis
Vilijampolė	Savanorių	124	Medinis	1975	100	6	Elektrinis
Šeškinė	Saulės	98B	Medinis	1990	260	9	Elektrinis
Žirmūnai	Karalių	75	Medinis	1900	250	8	Dujinis
Pilaitė	Ivanausko	222	Medinis	1990	120	7	Dujinis
Pilaitė	Ivanausko	245	Medinis	1920	150	6	Dujinis
Pilaitė	Ivanausko	201	Medinis	1980	90	4	Dujinis

Butai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Aukštas
Šeškinė	Tulpių	64	Mūrinis	2000	20	1	5
Saulėtekis	Tuskulėnų	81	Mūrinis	2005	90	4	1
Dainava	Taikos	34A	Mūrinis	1946	80	5	12
Saulėtekis	Minties	24	Mūrinis	1995	37	2	4
Pilaitė	Ivanausko	63	Mūrinis	2004	50	2	3

Duomenys, išvedami į ekraną:

Daugiausia 6 namai yra parduodami šiose gatvėse:
Ivanausko gatvėje
Konstitucijos gatvėje
Seniausi (pastatyti 1899 metais) namai:

Namo amžius: 119, adresas: Gėlių gatvė - 222 numeris, tipas: Mūrinis, plotas: 32 kv/m, aukštas: 15
Namo amžius: 119, adresas: Konstitucijos gatvė - 115 numeris, tipas: Mūrinis, plotas: 35 kv/m, aukštas: 41

Antras testas:

Namai1.csv:
UAB "Ponai"
Senukų gatvė 13, Kaunas
+37056952919
N,Saulėtekis,Ponų,15,Blokinis,1980,190,8,Elektra
N,Saulėtekis,Konstitucijos,124,Medinis,1975,100,6,Kietasis kuras
N,Pilaitė,Konstitucijos,222,Mūrinis,1899,120,7,Elektra

Namai2.csv
UAB "Dragai"
Persų gatvė 13; Kaunas
+378451541
N,Žirmūnai,Karalių,75,Medinis,1900,250,8,Dujinis
N,Saulėtekis,Konstitucijos,124,Medinis,1975,100,6,Kietasis kuras
B,Pilaitė,Kmynų,63,Medinis,2004,49,2,3

Kartojasi.csv
Mikrorajonas: Saulėtekis Gatvė: Konstitucijos Namo numeris: 124 Tipas: Medinis Pastatymo metai: 1975 Plotas: 100 Kambarių skaičius: 6 Šildymo būdas: Kietasis kuras

Namas100.csv
Mikrorajonas: Žirmūnai Gatvė: Karalių Namo numeris: 75 Tipas: Medinis Pastatymo metai: 1900 Plotas: 250 Kambarių skaičius: 8 Šildymo būdas: Dujinis
Mikrorajonas: Saulėtekis Gatvė: Ponų Namo numeris: 15 Tipas: Blokinis Pastatymo metai: 1980 Plotas: 190 Kambarių skaičius: 8 Šildymo būdas: Elektra
Mikrorajonas: Pilaitė Gatvė: Konstitucijos Namo numeris: 222 Tipas: Mūrinis Pastatymo metai: 1899 Plotas: 120 Kambarių skaičius: 7 Šildymo būdas: Elektra

Butas50.csv

Rezultatai, išvedami į ekraną:
Daugiausia 3 namai yra parduodami šiose gatvėse:
Konstitucijos gatvėje
Seniausi (pastatyti 1899 metais) namai:
Namo amžius: 119, adresas: Konstitucijos gatvė - 222 numeris, tipas: Mūrinis, plotas: 120 kv/m, šildymo būdas: Elektra
Tokių objektų nėra

DuomenysTekstiniame1.txt
Duomenys apie agentūras ir jų parduodamus objektus

Agentūros pavadinimas: UAB "Ponai"
Agentūros adresas: Senukų gatvė 13, Kaunas
Agentūros telefono numeris: +37056952919

Namai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Šildymo būdas
Saulėtekis	Ponų	15	Blokinis	1980	190	8	Elektra
Saulėtekis	Konstitucijos	124	Medinis	1975	100	6	Kietasis kuras
Pilaitė	Konstitucijos	222	Mūrinis	1899	120	7	Elektra

Butai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Aukštas

DuomenysTekstiniame2.txt

Duomenys apie agentūras ir jų parduodamus objektus

Agentūros pavadinimas: UAB "Dragai"
Agentūros adresas: Persų gatvė 13; Kaunas
Agentūros telefono numeris: +378451541

Namai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Šildymo būdas
Žirmūnai	Karalių	75	Medinis	1900	250	8	Dujinis
Saulėtekis	Konstitucijos	124	Medinis	1975	100	6	Kietasis kuras

Butai:

Mikrorajonas	Gatvė	Namo Nr.	Tipas	Pastatymo metai	Plotas	Kambarių skaičius	Aukštas
Pilaitė	Kmynų	63	Medinis	2004	49	2	3

3.4. Dėstytojo pastabos

4. Teksto analizė ir redagavimas

4.1. Darbo užduotis

U4-6. Skaitmenys

Tekstiniame faile Knyga.txt duotas tekstas sudarytas iš žodžių, atskirtų skyrikliais. Skyriklių aibė žinoma. Raskite ir spausdinkite faile Rodikliai.txt:

- ilgiausią sakinį (didžiausias žodžių kiekis), jo ilgį (simboliais ir žodžiais) ir vietą (sakinio pradžios eilutės numerį).
- Žodžių, kuriuos sudaro tik skaitmenys, kiekį. Suskaičiuokite tokių skaičių bendrą sumą. Reikia teksto žodžius sulygiuoti, kad kiekvienos eilutės kiekvienas žodis prasidėtų fiksuotoje toje pačioje pozicijoje. Galima įterpti tik minimalų būtiną tarpų skaičių. Galima šalinti kelis iš eilės einančius vienodus skyriklius, paliekant tik vieną jų atstovą. Įterpimo ir šalinimo taisykles taikome, siekdami gauti lygiuotą minimalų tekstą. Šalinimo taisyklės netaikome, jei nėra poreikio. Pradinio teksto eilutės ilgis neviršija 80 simbolių. Spausdinkite faile ManoKnyga.txt pertvarkytą tekstą pagal tokias taisykles:
- kiekvienos eilutės pirmasis žodis turi prasidėti pozicijoje p1=1.
- antrasis kiekvienos eilutės žodis turi prasidėti minimalioje galimoje pozicijoje p2, tokioje, kad kiekvienos eilutės pirmasis žodis kartu su už jo esančiais skyrikliais baigiasi iki p2-2 arba p2-1.
- trečiasis kiekvienos eilutės žodis turi prasidėti minimalioje galimoje pozicijoje p3, tokioje, kad kiekvienos eilutės antrasis žodis kartu su už jo esančiais skyrikliais baigiasi iki p3-2 arba p3-1.
- ir t.t.

4.2. Programos tekstas

Program.cs:

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

namespace laboras
{
    class Program
    {
        static void Main(string[] args)
        {
            const string CFd = "2Knyga.txt";
            const string CFr = "Rodikliai.txt";
            const string CFa = "ManoKnyga.txt";

            char[] skyrikliai = { ' ', ',', '!', '?', ':', ';', };//Įvairūs skyrikliai kurie
            gali atskirti žodžius.
            char[] sakinioskyr = { '!', '?', ';', '.', };//Įvairūs skyrikliai kurie gali
            atskirti sakinius.

            Program p = new Program();

            string tekstas = File.ReadAllText(@fv, Encoding.GetEncoding(1257));
            string ilgiausiassak = p.IlgSak(tekstas, sakinioskyr, skyrikliai);

            p.Spausdinti(CFr, tekstas, skyrikliai, ilgiausiassak, sakinioskyr);
            p.SpausdintiSulygiuotaTeksta(tekstas, CFa, skyrikliai);
        }

        /// <summary>
        /// Apskaičiuoja tekste esančių skaičių sumą
        /// </summary>
        /// <param name="tekstas"></param>Tekstas
        /// <param name="skyr"></param>Skyrikliai
```

```
/// <param name="sakinioskyr"></param>Sakinio skyrikliai
/// <returns></returns>Gražina sumą
double SkaiciuSuma(string tekstas, char[] skyr, char[] sakinioskyr)
{
    double suma = 0;
    string[] reiksme = tekstas.Split(sakinioskyr);

    for (int i = 0; i < reiksme.Length; i++)
    {
    string[] zodziai = reiksme[i].Split(skyr, StringSplitOptions.RemoveEmptyEntries);

        foreach(string testas in zodziai)
        {
            if(Regex.IsMatch(testas, @"^\d+$"))
            {
            suma += Convert.ToDouble(testas);
            }
        }
    }
    return suma;
}

/// <summary>
/// Apskaičuoja kiek yra skaitmenų tekste
/// </summary>
/// <param name="tekstas"></param>Tekstas
/// <param name="skyr"></param>Skyrikliai
/// <param name="sakinioskyr"></param>Sakinio skyrikliai
/// <returns></returns>Gražina kiekį
static int KiekSkaiciu(string tekstas, char[] skyr, char[] sakinioskyr)
{
    int kiekskait = 0;
    string[] reiksme = tekstas.Split(sakinioskyr);

    for (int i = 0; i < reiksme.Length; i++)
    {
      string[] zodziai = reiksme[i].Split(skyr,
      StringSplitOptions.RemoveEmptyEntries);

        foreach (string testas in zodziai)
        {
            if (Regex.IsMatch(testas, @"^\d+$"))
            {
                kiekskait++;
            }
        }
    }

    return kiekskait;
}

/// <summary>
/// Apskaičiuoja eilutės ilgį
/// </summary>
/// <param name="tekstas"></param>Tekstas
/// <param name="p"></param>
/// <returns></returns>Gražina ilgį
static int EilutesIlgis(string tekstas, int p)
{
    int ilgis = 0;

    string[] eilutes = Regex.Split(tekstas, "\r\n|\r|\n");

    foreach (var eilute in eilutes)
    {
        string[] zodziai = eilute.Split(' ');

        if (zodziai.Count() > p && zodziai[p].Count() > ilgis)
```

```
            ilgis = zodziai[p].Count();
        }

        return ilgis;
    }

    /// <summary>
    /// Suranda ilgiausio sakinio eilute
    /// </summary>
    /// <param name="ilgiausiassak"></param>Ilgiausas sakinys
    /// <param name="tekstas"></param>Tekstas
    /// <param name="skyrikliai"></param>Skyrikliai
    /// <returns></returns>Gražina ilgiausio sakinio eilutės reiškmę
    static int IlgEil(string ilgiausiassak, string tekstas, char[] skyrikliai)
    {
        string[] eilutes = tekstas.Split(skyrikliai);
        int eilnr = 2;

        for (int i = 0; i < eilutes.Count(); i++)
        {
            if (eilutes[i] == ilgiausiassak)
            {
                eilnr = i;
            }
        }

        return eilnr;
    }

    /// <summary>
    /// Apskaičiuoja ilgiausio sakinio žodžių kiekį
    /// </summary>
    /// <param name="skyr"></param>Skyrikliai
    /// <param name="ilgiausiassak"></param>Ilgiausias sakinys
    /// <returns></returns>Gražina žodžių kiekį
    static int IlgSakZodziai(char[] skyr, string ilgiausiassak)
    {
         int zodzskaic;
          string[] zodziai = ilgiausiassak.Split(skyr,
          StringSplitOptions.RemoveEmptyEntries);

        zodzskaic = zodziai.Count();

        foreach (string zodis in zodziai)
        if (Regex.IsMatch(zodis, @"^\d+$"))
        {
            zodzskaic--;
        }
        return zodzskaic;
    }

    /// <summary>
    /// Apskaičiuoja kiek yra žodžių sakiny.
    /// </summary>
    /// <param name="tekstas">Tekstas</param>
    /// <param name="sakinioskyr">Sakinio skyrikliai</param>
    /// <param name="skyrikliai">Žodžių skyrikliai</param>
    /// <returns></returns>
    static int SakinioZodziai(string tekstas, char[] sakinioskyr, char[] skyrikliai)
    {
        int kiekis = 0;

        string[] eilutes = tekstas.Split(sakinioskyr);

        foreach (var eilute in eilutes)
        {
            string[] zodziai = eilute.Split(skyrikliai);
```

```
                kiekis = zodziai.Count();
            }

            return kiekis;
        }

        /// <summary>
        /// Randa ilgiausią sakinį.
        /// </summary>
        /// <param name="tekstas"></param>Tekstas
        /// <param name="sakinioskyr"></param>Sakinio skyrikliai
        /// <returns></returns>Gražina ilgiausią sakinį
        string IlgSak(string tekstas, char[] sakinioskyr, char[] skyr)
        {
            string ilgiausiassak = "";

            int maxi = 0;
            int max = 0;
            string[] reiksmes = tekstas.Split(sakinioskyr);

            for (int i = 0; i < reiksmes.Length; i++)
            {
                int sakinio = SakinioZodziai(reiksmes[i], sakinioskyr, skyr);
                if (sakinio > max)
                {
                    max = sakinio;
                    maxi = i;
                }
            }

            string[] eilutes = Regex.Split(reiksmes[maxi], "\r\n|\r|\n");
            for (int k = 0; k < eilutes.Count(); k++)
            {
                ilgiausiassak += eilutes[k] + " ";
            }

            return ilgiausiassak;
        }

        /// <summary>
        /// Spausdina rezultatus
        /// </summary>
        /// <param name="fvr"></param>Rezultatų failo pavadinimas
        /// <param name="tekstas"></param>Tekstas
        /// <param name="skyrikliai"></param>Skyrikliai
        /// <param name="ilgiausiassak"></param>Ilgiausias sakinys
        /// <param name="sakinioskyr"></param>Sakinio skyrikliai
        void Spausdinti(string fvr, string tekstas, char[] skyrikliai, string ilgiausiassak,
        char[] sakinioskyr)
        {
              using (StreamWriter rasyti = new StreamWriter(fvr, false,
              Encoding.GetEncoding(1257)))
            {
              rasyti.WriteLine("{0}", IlgSak(tekstas, sakinioskyr, skyrikliai));
              rasyti.WriteLine("- simboliai: {0}", IlgSak(tekstas,   sakinioskyr,
              skyrikliai).Count());
               rasyti.WriteLine("- žodžiai: {0}", IlgSakZodziai(skyrikliai, ilgiausiassak));
               rasyti.WriteLine("- eilutė: {0}", IlgEil(ilgiausiassak, tekstas, sakinioskyr));
              rasyti.WriteLine("- viso teksto skaitmenų yra {0}, jų suma: {1}",
              KiekSkaiciu(tekstas, skyrikliai, sakinioskyr), SkaiciuSuma(tekstas, skyrikliai,
              sakinioskyr));
            }
        }

        /// <summary>
        /// Spausdina sulygiuota tekstą
        /// </summary>
        /// <param name="tekstas"></param>Tekstas
```

```
        /// <param name="failas"></param>Rezultatų failas
        /// <param name="skyr"></param>Skyrikliai
        void SpausdintiSulygiuotaTeksta(string tekstas, string failas, char[] skyr)
        {

            int n = 0;
            string[] sakiniai = Regex.Split(tekstas, "\r\n|\r|\n");

            using (StreamWriter Lygiuotarasyti = new StreamWriter(failas, false,
            Encoding.GetEncoding(1257)))
            {
                foreach (string sakinys in sakiniai)
                {
                     string[] zodziai = sakinys.Split(skyr,
                     StringSplitOptions.RemoveEmptyEntries);

                    for (int i = 0; i < zodziai.Count(); i++)
                    {
                        for (int a = zodziai[i].Length; a < EilutesIlgis(tekstas, i); a++)
                        {
                            zodziai[i] += " ";
                        }

                        if (zodziai[i].Length != EilutesIlgis(tekstas, i))
                            zodziai[i] += " ";
                    }

                    sakiniai[n++] = string.Join(" ", zodziai);

                    Lygiuotarasyti.WriteLine("{0}", sakiniai[n - 1]);
                }
            }
        }
    }
}
```

4.3. Pradiniai duomenys ir rezultatai

Pirmas testas:

Knyga.txt:

```
Dursliai turejo viska, ko tiktai geide, bet jie turejo ir paslapti ir labiausiai
bijojo, kad jos kas nors neatkapstytu.
Mane, jog neištvertu, jei žmones sužinotu apie Poterius. Ponia Dursli buvo ponios
Poter sesuo,
bet jos nesimate jau 20 metu iš teisybes, ponia Dursli apsimete visai neturinti
sesers, nes ta sesuo ir jos niekam tikes vyras buvo visiška Dursliu priešingybe.
Dursliai net
sudrebedavo pagalvoje, ka pasakytu kaimynai, jei 13 gatveje pasirodytu Poteriai.
Dursliai
žinojo, kad Poteriai irgi turi maža suneli, bet jo nebuvo net mate. Jie nenorejo
giminiuotis su
Poteriais ir del to berniuko: to dar betruko, kad Dudlis žaistu su tokiu vaiku.
```

ManoKnyga.txt:

```
Dursliai    turejo     viska       ko       tiktai    geide    bet     jie       turejo     ir        paslapti ir      labiausiai
bijojo      kad        jos         kas      nors      neatkapstytu.
Mane        jog        neištvertu  jei      žmones    sužinotu apie    Poterius. Ponia      Dursli    buvo     ponios  Poter
sesuo
bet         jos        nesimate    jau      20        metu     iš      teisybes  ponia      Dursli    apsimete visai   neturinti
sesers      nes        ta          sesuo    ir        jos      niekam  tikes     vyras      buvo      visiška  Dursliu priešingybe.
Dursliai    net
sudrebedavo pagalvoje  ka          pasakytu kaimynai  jei      13      gatveje   pasirodytu Poteriai.
Dursliai
žinojo      kad        Poteriai    irgi     turi      maža     suneli  bet       jo         nebuvo    net      mate.   Jie
nenorejo    giminiuotissu
Poteriais   ir         del         to       berniuko  to       dar     betruko   kad        Dudlis    žaistu   su      tokiu
vaiku.
```

Rodikliai.txt:

Ponia Dursli buvo ponios Poter sesuo, bet jos nesimate jau 20 metu iš teisybes, ponia Dursli apsimete visai neturinti sesers, nes ta sesuo ir jos niekam tikes vyras buvo visiška Dursliu priešingybe.
- simboliai: 199
- þodþiai: 31
- eilutë: 2
- viso teksto skaitmenø yra 2, jø suma: 33

Antras testas:

2Knyga.txt:

Taciau ivažiavus i miesta, mintis apie gražtus išgaravo. Stovedamas rytmetineje spustyje,
negalejo nepastebeti, kad aplinkui pilna keistai apsirengusiu praeiviu. Su mantijomis. Ponas
Durslis negalejo pakesti neiprastai apsirengusiu žmoniu - o jau to jaunimelio apranga! Pamane,
jog atejo kažkokia kvaila nauja mada. Barbendamas 5 pirštais i vaira, nužvelge arciausiai stovinti
tu keistuoliu 12 bureli. Jie susijaudine šnibždejosi. Ponas Durslis pasipiktino pamates, kad 10 tu
žmoniu visai ne jaunikliai: va, tasai tikriausiai vyresnis už ji pati, taciau su skaisciai žalia
mantija! Koks ižulumas! Paskui ponui Dursliui dingtelejo, kad cia, matyt, kažkoks paikas
pokštas, - tie žmones, be abejo, kam nors renka aukas... taip, aišku. 30 Automobiliu kolona
pajudejo, ir po 28 minuciu ponas Durslis, vel užsigalvojes apie gražtus, ivažiavo i „Graningso"
kiema.

ManoKnyga.txt:

Taciau	ivažiavus	i	miesta	mintis	apie	gražtus	išgaravo.	Stovedamas	rytmetineje	spustyje	
negalejo	nepastebeti	kad	aplinkui	pilna	keistai	apsirengusiu	praeiviu.	Su	mantijomis.	Ponas	
Durslis	negalejo	pakesti	neiprastai	apsirengusiu	žmoniu	-	o	jau	to	jaunimelio	
apranga	Pamane										
jog	atejo	kažkokia	kvaila	nauja	mada.	Barbendamas	5	pirštais	i	vaira	
nužvelge	arciausiai	stovinti									
tu	keistuoliu	12	bureli.	Jie	susijaudine	šnibždejosi.	Ponas	Durslis	pasipiktino	pamates	kad
10	tu										
žmoniu	visai	ne	jaunikliai	va	tasai	tikriausiai	vyresnis	už	ji	pati	taciau
su	skaisciai	žalia									
mantija	Koks	ižulumas	Paskui	ponui	Dursliui	dingtelejo	kad	cia	matyt	kažkoks	paikas
pokštas	-	tie	žmones	be	abejo	kam	nors	renka	aukas...	taip	aišku.
30	Automobiliu	kolona									
pajudejo	ir	po	28	minuciu	ponas	Durslis	vel	užsigalvojes	apie	gražtus	
ivažiavo	i	„Graningso"									
kiema.											

Rodikliai.txt:

Ponas Durslis pasipiktino pamates, kad 10 tu žmoniu visai ne jaunikliai: va, tasai tikriausiai vyresnis už ji pati, taciau su skaisciai žalia mantija
- simboliai: 151
- þodþiai: 22
- eilutë: 5
- viso teksto skaitmenø yra 5, jø suma: 85

4.4. Dėstytojo pastabos

5. Polimorfizmas

5.1. Darbo užduotis

U5_8. Turistų informacijos centras.

Turizmo informacijos centre perorganizuoti ir atskirai surašyti duomenys apie kiekviename mieste veikiančius muziejus. Pirmoje eilutėje – miestas, antroje – atsakingo asmens vardas ir pavardė. Turizmo informacijos centras teikia informaciją apie lankytinas vietas – muziejus, paminklus ir kita. Sukurkite abstrakčiąją klasę „LankytinaVieta“ (laukai - pavadinimas, adresas, įkūrimo ar pastatymo metai), kurią paveldės klasės “Muziejus” (papildomas laukas – tipas, 7 savaitės dienos (1 – darbo, 0 – nedarbo), požymis „turi gidą“, bilieto kaina) ir “Paminklas” (papildomas laukas – autorius, kam skirtas).

- Suskaičiuokite, kiek muziejų turi gidus, rezultatą atspausdinkite ekrane.
- Raskite seniausią lankytiną vietą, visą informaciją apie ją atspausdinkite ekrane.
- Sudarykite visų lankytinų vietų sąrašą ir įrašykite į failą „VisosVietos.csv“.
- Sudarykite ir surikiuokite naujų lankytinų vietų sąrašą, pateikdami pilną informaciją apie juos. Muziejus yra naujas, jei nuo įkūrimo prabėgo mažiau, nei 2 metai. Paminklas yra naujas, jei nuo pastatymo prabėgo mažiau nei metai. Muziejus rikiuokite pagal bilieto kainas, paminklus – pagal autorius. Rezultatus įrašykite į failą „Nauji.csv“.

5.2. Programos tekstas

AllPlaces.cs

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

namespace U3_8_Turistu_informacijos_centras
{
    // Klasė skirta visoms lankytinoms vietoms
    class AllPlaces : LankytinaVieta
    {
        public AllPlaces()
        {

        }

        /// <param name="title">Pavadinimas</param>
        /// <param name="addr">Adresas</param>
        /// <param name="year">Pastatymo/įkurimo Metai</param>
        public AllPlaces(string title, string addr, int year) : base(title, addr, year)
        {

        }

        public override string ToString()
        {
            throw new NotImplementedException();
        }

        public override bool GetGuide()
        {
            throw new NotImplementedException();
        }
    }
}
```

LankytinaVieta.cs

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

namespace U3_8_Turistu_informacijos_centras
{
    // Lankytinų vietų bazinė klasė
    abstract class LankytinaVieta : Object
    {
        // Pavadinimas
        public string Title { get; set; }
        // Adresas
        public string Addr { get; set; }
        // Metai
        public int Year { get; set; }

        public LankytinaVieta()
        {

        }

        /// <param name="title">Paminklo/muziejaus pavadinimas</param>
        /// <param name="addr">Adresas</param>
        /// <param name="year">Pastatymo/įkūrimo Metai</param>
        public LankytinaVieta(string title, string addr, int year)
        {
            Title = title;
            Addr = addr;
            Year = year;
        }
        public LankytinaVieta(string data)
        {
            SetData(data);
        }
        public virtual void SetData(string line)
        {
            string[] values = line.Split(',');
            Title = values[1];
            Addr = values[2];
        }

        public override bool Equals(object obj)
        {
            return this.Equals(obj as LankytinaVieta);
        }

        public bool Equals(LankytinaVieta lankytinaVieta)
        {
            if (Object.ReferenceEquals(lankytinaVieta, null))
            {
                return false;
            }
            if (this.GetType() != lankytinaVieta.GetType())
            {
                return false;
            }
            return (Addr == lankytinaVieta.Addr) && (Title == lankytinaVieta.Title);
        }

        public override int GetHashCode()
        {
            return Addr.GetHashCode() ^ Title.GetHashCode();
        }
```

```
        public static bool operator <(LankytinaVieta objektas1, LankytinaVieta objektas2)
        {
        if (String.Compare((objektas1 as Paminklas).Author, (objektas2 as Paminklas).Author,
        StringComparison.CurrentCulture) < 0)
                return true;
            return false;
        }

        public static bool operator >(LankytinaVieta objektas1, LankytinaVieta objektas2)
        {
        if (String.Compare((objektas1 as Paminklas).Author, (objektas2 as Paminklas).Author,
        StringComparison.CurrentCulture) > 0)
                return true;
            return false;
        }
        abstract public override string ToString();

        abstract public bool GetGuide();
    }
}
```

Museum.cs

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

namespace U3_8_Turistu_informacijos_centras
{
    // Klasė skirta Muziejams
    class Museum : LankytinaVieta
    {
        // Muziejaus tipas
        public string Type { get; set; }
        // Darbo dienos
        public bool[] Open { get; set; } = new bool[7];
        // Ar turi gidą
        public bool Guide { get; set; }
        // Bilieto kaina
        public double Price { get; set; }

        public Museum()
        {

        }

        /// <param name="type">Muziejaus tipas</param>
        /// <param name="open">Darbo dienos</param>
        /// <param name="guide">Ar turi gidą</param>
        /// <param name="price">Bilieto kaina</param>
        public Museum(string type, bool[] open, bool guide, double price, string title, string
        addr, int year) : base(title, addr, year)
        {
            Type = type;
            Open = open;
            Guide = guide;
            Price = price;
        }
        public Museum(string data)
        : base(data)
        {
            SetData(data);
        }
        public override string ToString()
        {
```

```
string eilute;
eilute = string.Format("| {0, -36} | {1, -25} | {2, 5} | {3, -10} | {4} | {5} | {6} | {7} |
{8} | {9} | {10} | {11, -5} |  {12:f} |",
Title, Addr, Year, Type, Open[0].Equals(true) ? "+" : "-", Open[1].Equals(true) ? "+" : "-",
Open[2].Equals(true) ? "+" : "-",
Open[3].Equals(true) ? "+" : "-", Open[4].Equals(true) ? "+" : "-", Open[5].Equals(true) ?
"+" : "-",
Open[6].Equals(true) ? "+" : "-", Guide.Equals(true) ? "Yra" : "Nėra", Price);
return eilute;
        }

        // Grąžina ar turi gidą
        public override bool GetGuide() {
            return Guide;
        }
    }
}
```

Paminklas.cs

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

namespace U3_8_Turistu_informacijos_centras
{
    // Klasė skirta Paminklams
    class Paminklas : LankytinaVieta
    {
        // Autorius
        public string Author { get; set; }
        // Paminklas skirtas
        public string Dedic { get; set; }

        public Paminklas()
        {

        }

        /// <param name="author">Paminklo autorius</param>
        /// <param name="dedic">Paminklas skirtas</param>
       public Paminklas(string author, string dedic, string title, string addr, int year) :
       base(title, addr, year)
        {
            Author = author;
            Dedic = dedic;
        }
        public Paminklas(string data)
        : base(data)
        {
            SetData(data);
        }
        public override string ToString()
        {
              string eilute;
              eilute = string.Format("| {0, -29} | {1, -21} | {2, 5} | {3, -20} | {4, -22}
              |", Title, Addr, Year, Dedic, Author);

            return eilute;
        }

        public override bool GetGuide()
        {
            throw new NotImplementedException();
        }
    }
```

```
}
```

PContainer.cs

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

namespace U3_8_Turistu_informacijos_centras
{
    // Muziejų, paminklų, lankytinų vietų konteineris
    class PContainer
    {
        private LankytinaVieta[] LankVietos;

        public int Count { get; private set; }

        public PContainer(int dydis = 999)
        {
            LankVietos = new LankytinaVieta[dydis];
            Count = 0;
        }

        public void AddPlace(LankytinaVieta lankVieta)
        {
            LankVietos[Count++] = lankVieta;
        }

        public LankytinaVieta GetPlace(int indeksas)
        {
            return LankVietos[indeksas];
        }
        public bool Contains(LankytinaVieta objektas)
        {
            return LankVietos.Contains(objektas);
        }

        public void Rikiuoti_Pagal_Autoriu()
        {
            for (int i = 0; i < Count - 1; i++)
            {
                for (int j = i + 1; j < Count; j++)
                {
                    LankytinaVieta laikinas = LankVietos[i];
                    if (GetPlace(j) < GetPlace(i))
                    {
                        LankVietos[i] = LankVietos[j];
                        LankVietos[j] = laikinas;
                    }
                }
            }
        }
        public void Rikiuoti_Pagal_Kaina()
        {
            for (int i = 0; i < Count; i++)
            {
                for (int j = 0; j < Count; j++)
                {
                    LankytinaVieta laikinas = LankVietos[i];
                     if ((GetPlace(j) as Museum).Price < (GetPlace(i) as Museum).Price ||
                     ((GetPlace(j) as Museum).Price == (GetPlace(i) as Museum).Price))
                    {
                        LankVietos[i] = LankVietos[j];
                        LankVietos[j] = laikinas;
                    }
                }
```

```
            }
        }
    }
}
```

Program.cs

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

namespace U3_8_Turistu_informacijos_centras
{
    class Program
    {
        static void Main(string[] args)
        {
            // Randa visus duomenų failus
              string[] dFiles = Directory.GetFiles(@"Miestai",
              "*.txt").Select(Path.GetFileName).ToArray();

            if (dFiles.Count() != 0)
            {
                int MYears = 2;//Naujo muziejaus senumo metai
                int PYears = 1;//Naujo paminklo senumo metai

                // Muziejų konteineris
                PContainer MusCon = new PContainer();

                // Paminklų konteineris
                PContainer PamCon = new PContainer();

                //Naujų muziejų konteineris
                PContainer NewMuseumPlaces = new PContainer();

                //Naujų paminklų konteineris
                PContainer NewMonumentPlaces = new PContainer();

                // Visos lankytinos vietos
                PContainer AllPlaces = new PContainer();

                // Duomenų skaitymas
                ReadingData(MusCon, PamCon, dFiles);

                // Perskaitytų duomenų išvedimas
                ReadDataOutput(MusCon, PamCon);

                // Suskaičiuoja kiek muziejų turi gidus
                int GuidesNum = CountGuides(MusCon);
                Console.WriteLine("Gidus turi {0} muziejus(-ų)", GuidesNum);

                // Randa seniausią lankytiną vietą
                int[] op = OldestPlace(MusCon, PamCon);

                // Išveda seniausią lankytiną vietą į konsolę
                OlddestPlaceOutput(MusCon, PamCon, op);

                // Sukelia muziejus ir paminklus į vieną konteinerį
                FillAllPlacesCon(AllPlaces, MusCon, PamCon);

                // Visų lankytinų vietų rikiavimas
                SortingPlaces(AllPlaces);

                // Spausdina surikiuotas visas lankytinas vietas į VisosVietos.csv
                AllPlacesOutput(AllPlaces);
```

```
            // Sukelia visas naujas lankytinas vietas į viena konteinerį
          GetNewestPlaces(MusCon, PamCon, MYears, PYears,   NewMuseumPlaces,
          NewMonumentPlaces);

            // Visų naujų muziejų rikiavimas
            NewMuseumPlaces.Rikiuoti_Pagal_Kaina();

            // Visų naujų paminklų rikiavimas
            NewMonumentPlaces.Rikiuoti_Pagal_Autoriu();

            // Spausdina surikiuotas visas naujas lankytinas vietas į Nauji.csv
            AllNewPlacesOutput(NewMuseumPlaces, NewMonumentPlaces);
        }
        else
        {
            Console.WriteLine("Nėra duomenų failų");
        }

        Console.ReadKey();
    }

    /// <summary>
    /// Duomenų skaitymas
    /// </summary>
    /// <param name="MusCon">Muziejų konteineris</param>
    /// <param name="PamCon">Paminklų konteineris</param>
    /// <param name="dFiles">Skaitomo failo pavadinimas</param>
    static void ReadingData(PContainer MusCon, PContainer PamCon, string[] dFiles)
    {
        foreach (var file in dFiles)
        {
            string[] lines = File.ReadAllLines(@"Miestai/" + file);

            foreach (var line in lines)
            {
                string[] value = line.Split(';');

                if(value[0].Equals("M"))
                {
                    // Pridedam muziejus i konteineri
                    string title = value[1].Trim();
                    string addr = value[2].Trim();
                    int year = int.Parse(value[3]);
                    string type = value[4].Trim();
                    bool[] open = new bool[7];
                    for (int i = 0; i < 7; i++)
                    {
                        open[i] = value[5 + i].Equals(" 1") ? true : false;
                    }
                    bool guide = value[12].Trim().Equals("taip") ? true : false;
                    double price = double.Parse(value[13]);

                    Museum museum = new Museum(type, open, guide, price, title, addr,
                    year);

                    MusCon.AddPlace(museum);
                }
                else if (value[0].Equals("P"))
                {
                    // Pridedam paminklus i konteineri
                    string title = value[1].Trim();
                    string addr = value[2].Trim();
                    int year = int.Parse(value[3]);
                    string dedic = value[4].Trim();
                    string author = value[5].Trim();
```

```
                Paminklas paminklas = new Paminklas(author, dedic, title, addr,
                year);

                PamCon.AddPlace(paminklas);
            }
        }
    }
}

/// <summary>
/// Perskaitytų duomenų išvedimas
/// </summary>
/// <param name="MusCon">Muziejų konteineris</param>
/// <param name="PamCon">Paminklų konteineris</param>
static void ReadDataOutput(PContainer MusCon, PContainer PamCon)
{
    using (var writer = new StreamWriter("Duomenys.txt", false, Encoding.UTF8))
    {
      writer.WriteLine("Muziejai:");
      writer.WriteLine("+-----------------------------------------------------------
      ------------------------------------------------------------------------+");
        writer.WriteLine("|             Pavadinimas              |          Adresas
      | Metai |   Tipas    | P | A | T | K | P | Š | S | Gidas | Kaina |");

        for (int i = 0; i < MusCon.Count; i++)
        {
                        writer.WriteLine("|--------------------------------------|-
      -------------------------|-------|-----------|---|---|---|---|---|---|---|---
      ----|-------|");
            writer.WriteLine(MusCon.GetPlace(i).ToString());
        }
                    writer.WriteLine("+--------------------------------------------
      ------------------------------------------------------------------------------
      --------+");


        writer.WriteLine();
        writer.WriteLine("Paminklai:");
                    writer.WriteLine("+--------------------------------------------
      ------------------------------------------------------------------+");
        writer.WriteLine("|          Pavadinimas          |        Adresas        |
      Metai |        Skirta        |        Autorius        |");

        for (int i = 0; i < PamCon.Count; i++)
        {
                        writer.WriteLine("|-------------------------------|--------
      --------------|-------|----------------------|----------------------|");
            writer.WriteLine(PamCon.GetPlace(i).ToString());
        }
                    writer.WriteLine("+--------------------------------------------
      ------------------------------------------------------------------+");
    }
}

/// <summary>
/// Suskaičiuoja kiek muziejų turi gidus
/// </summary>
/// <param name="MusCont">Muziejų konteineris</param>
/// <returns>Gražina suskaičiuotų gidų skaičių</returns>
static int CountGuides(PContainer MusCont)
{
    int num = 0;

    for (int i = 0; i < MusCont.Count; i++)
    {
        if(MusCont.GetPlace(i).GetGuide() == true)
        {
            num++;
```

```
            }
        }

        return num;
    }

    /// <summary>
    /// Randa seniausią lankytiną vietą
    /// </summary>
    /// <param name="MusCon">Muziejų konteineris</param>
    /// <param name="PamCon">Paminklų konteineris</param>
    /// <returns>Gražina seniausią lankytina vietą</returns>
    static int[] OldestPlace(PContainer MusCon, PContainer PamCon)
    {
        // Pirmoj vietoj: 0/1 - muziejus/paminklas, antroj - jo id;
        int[] op = new int[2];
        int year = 9999;

        for (int i = 0; i < MusCon.Count; i++)
        {
            if(year > MusCon.GetPlace(i).Year)
            {
                year = MusCon.GetPlace(i).Year;
                op[0] = 0;
                op[1] = i;
            }
        }

        for (int i = 0; i < PamCon.Count; i++)
        {
            if (year > PamCon.GetPlace(i).Year)
            {
                year = PamCon.GetPlace(i).Year;
                op[0] = 1;
                op[1] = i;
            }
        }

        return op;
    }

    /// <summary>
    /// Išveda seniausią lankytiną vietą į konsolę
    /// </summary>
    /// <param name="MusCon">Muziejų konteineris</param>
    /// <param name="PamCon">Paminklų konteineris</param>
    /// <param name="op">Seniausia lankytina vieta</param>
    static void OlddestPlaceOutput(PContainer MusCon, PContainer PamCon, int[] op)
    {
        if (op[0] == 0)
        {
            Console.WriteLine();
            Console.WriteLine("Seniausia lankytina vieta yra muziejus:");
          Console.WriteLine("+-------------------------------------------------------------
          ----------------------------------------------------------------------+");
            Console.WriteLine("|             Pavadinimas              |          Adresas
          | Metai |   Tipas    | P | A | T | K | P | Š | S | Gidas | Kaina |");
            Console.WriteLine("|--------------------------------------|-----------------
          ---------|-------|------------|---|---|---|---|---|---|---|-------|-------|");
            Console.WriteLine(MusCon.GetPlace(op[1]).ToString());
                        Console.WriteLine("+---------------------------------------
          ----------------------------------------------------------------------------
          ---------+");
        }
        else
        {
            Console.WriteLine();
            Console.WriteLine("Seniausia lankytina vieta yra paminklas:");
```

```
            Console.WriteLine("+-------------------------------------------------------------
            ----------------------------------------------------+");
              Console.WriteLine("|          Pavadinimas          |        Adresas        |
            Metai |        Skirta        |        Autorius        |");
              Console.WriteLine("|-------------------------------|-----------------------|-
            ------|----------------------|------------------------|");
              Console.WriteLine(PamCon.GetPlace(op[1]).ToString());
                          Console.WriteLine("+-----------------------------------------
            ----------------------------------------------------------------+");
        }
    }

    /// <summary>
    /// Sukelia muziejus ir paminklus į vieną konteinerį
    /// </summary>
    /// <param name="AllPlaces">Visų lankytinų vietų konteineris</param>
    /// <param name="MusCon">Muziejų konteineris</param>
    /// <param name="PamCon">Paminklų konteineris</param>
    static void FillAllPlacesCon(PContainer AllPlaces, PContainer MusCon, PContainer
    PamCon)
    {
        for (int i = 0; i < MusCon.Count; i++)
        {
          AllPlaces lv = new AllPlaces(MusCon.GetPlace(i).Title, "",
          MusCon.GetPlace(i).Year);
            AllPlaces.AddPlace(lv);
        }
        for (int i = 0; i < PamCon.Count; i++)
        {
          AllPlaces lv = new AllPlaces(PamCon.GetPlace(i).Title, "",
          PamCon.GetPlace(i).Year);
            AllPlaces.AddPlace(lv);
        }
    }
        /// <summary>
        /// Visų lankytinų vietų rikiavimas
        /// </summary>
        /// <param name="AllPlaces">Visų lankytinų vietų konteineris</param>
        static void SortingPlaces(PContainer AllPlaces)
        {
        int year;
        string title;
        for (int i = 0; i < AllPlaces.Count; i++)
        {
            for (int j = 0; j < AllPlaces.Count; j++)
            {
                if (AllPlaces.GetPlace(i).Year < AllPlaces.GetPlace(j).Year)
                {
                    year = AllPlaces.GetPlace(i).Year;
                    AllPlaces.GetPlace(i).Year = AllPlaces.GetPlace(j).Year;
                    AllPlaces.GetPlace(j).Year = year;

                    title = AllPlaces.GetPlace(i).Title;
                    AllPlaces.GetPlace(i).Title = AllPlaces.GetPlace(j).Title;
                    AllPlaces.GetPlace(j).Title = title;
                }
            }
        }

        for (int i = 0; i < AllPlaces.Count; i++)
        {
            for (int j = 0; j < AllPlaces.Count; j++)
            {
              if (AllPlaces.GetPlace(i).Title.CompareTo(AllPlaces.GetPlace(j).Title) < 0)
                {
                    title = AllPlaces.GetPlace(i).Title;
                    AllPlaces.GetPlace(i).Title = AllPlaces.GetPlace(j).Title;
                    AllPlaces.GetPlace(j).Title = title;
```

```
                year = AllPlaces.GetPlace(i).Year;
                AllPlaces.GetPlace(i).Year = AllPlaces.GetPlace(j).Year;
                AllPlaces.GetPlace(j).Year = year;
            }
        }
    }
}

/// <summary>
/// // Spausdina surikiuotas visas lankytinas vietas į VisosVietos.csv
/// </summary>
/// <param name="AllPlaces">Visų lankytinų vietų konteineris</param>
static void AllPlacesOutput(PContainer AllPlaces)
{
    using (var writer = new StreamWriter("VisosVietos.csv", false, Encoding.UTF8))
    {
        for (int i = 0; i < AllPlaces.Count; i++)
        {
            writer.WriteLine(AllPlaces.GetPlace(i).Title);
        }

        writer.Close();
    }
}

/// <summary>
/// // Spausdina surikiuotas visas lankytinas vietas į Nauji.csv
/// </summary>
/// <param name="NewMuseumPlaces">Naujų muziejų konteineris</param>
/// <param name="NewMonumentPlaces">Naujų paminklų konteineris</param>
static void AllNewPlacesOutput(PContainer NewMuseumPlaces, PContainer
NewMonumentPlaces)
{
    using (var writer = new StreamWriter("Nauji.csv", false, Encoding.UTF8))
    {
        string rez = "nera";

        for (int i = 0; i < NewMuseumPlaces.Count; i++)
        {
            rez = NewMuseumPlaces.GetPlace(i).ToString();

            writer.WriteLine(NewMuseumPlaces.GetPlace(i).ToString().Trim('|').Replac
            e('|', ';'));
        }

        for (int i = 0; i < NewMonumentPlaces.Count; i++)
        {
            rez = NewMonumentPlaces.GetPlace(i).ToString();

            writer.WriteLine(NewMonumentPlaces.GetPlace(i).ToString().Trim('|').Repl
            ace('|', ';'));
        }

        if (rez == "nera")
            writer.WriteLine("Nėra nei vienos naujos lankytinos vietos");
        writer.Close();
    }
}
/// <summary>
/// Suveda naujus muziejus/paminklus į atskirus konteinerius
/// </summary>
/// <param name="MusCon">Muziejų konteineris</param>
/// <param name="PamCon">Paminklų konteineris</param>
/// <param name="MYears">Metai kiek dar būna naujas muziejus</param>
/// <param name="PYears">Metai kiek dar būna naujas paminklas</param>
/// <param name="NewMuseumPlaces">Naujų muziejų konteineris</param>
/// <param name="NewMonumentPlaces">Naujų paminklų konteineris</param>
```

```
        static void GetNewestPlaces(PContainer MusCon, PContainer PamCon, int MYears, int
        PYears, PContainer NewMuseumPlaces, PContainer NewMonumentPlaces)
         {
             double years = DateTime.Now.Year;

             for (int i = 0; i < MusCon.Count; i++)
             {
                 if ((years - MusCon.GetPlace(i).Year) < MYears)
                 {
                     NewMuseumPlaces.AddPlace(MusCon.GetPlace(i));
                 }
             }

             for (int i = 0; i < PamCon.Count; i++)
             {
                 if ((years - PamCon.GetPlace(i).Year) < PYears)
                 {
                     NewMonumentPlaces.AddPlace(PamCon.GetPlace(i));
                 }
             }
         }
    }
}
```

5.3. Pradiniai duomenys ir rezultatai

Pirmas testas:

Kaunas.txt

```
Kaunas
Virgis Jovaisa
M; Baltu botanikos muziejus; Geliu g. 2, Kaunas; 1925; Istorija; 0; 1; 1; 1; 0;
0; 0; taip; 3,00
M; Senoviniu masinu muziejus; Uzupio g. 9, Kaunas; 1968; Istorija; 1; 1; 0; 1; 1;
0; 0; taip; 5,00
P; Vytautas Didysis; Kaunas 44251; 1932; Vytautui; Aardenis Pavardenis
P; Martynas Maþvydas; Kaunas 44251; 2018; Martynui; Vardenis Pavardenis
M; Muziejus; Uzupio g. 9, Kaunas; 2017; Istorija; 1; 1; 0; 1; 1; 0; 0; taip; 4,00
M; Senoviniu ginklu muziejus; Uzupio g. 9, Kaunas; 2016; Istorija; 1; 1; 0; 1; 1;
0; 0; taip; 5,00
```

Vilnius.txt

```
Vilnius
Jonas Jonaitis
M; Dailes muziejus; Kranto g. 8, Vilnius; 1979; Menas; 0; 0; 1; 0; 0; 1; 1; taip;
0,00
M; Vytauto Didziojo muziejus; Tevynes pr. 9, Vilnius; 2018; Istorija; 0; 0; 0; 0;
0; 1; 1; taip; 7,50
M; Lietuvos teatro ir muzikos muziejus; Pilies g. 46, Vilnius; 1939; Istorija; 0;
1; 1; 1; 1; 0; 0; taip; 2,50
M; Jotvario kino muziejus; Lentvario g. 6, Vilnius; 1990; Zoologija; 0; 0; 0; 0;
0; 1; 1; taip; 0,00
M; Manto atminimo muziejus; Lentvario g. 6, Vilnius; 2017; Zoologija; 0; 0; 0; 0;
0; 1; 1; taip; 0,00
P; Zulius Asimas; Kaunas 44251; 2018; Vytautui; Zardenis Pavardenis
P; Antanas Vatas; Kaunas 44251; 2018; Martynui; Aardenis Pavardenis
```

Nauji.csv

```
Vytauto Didziojo muziejus            ; Tevynes pr. 9, Vilnius    ;  2018 ;
Istorija   ; - ; - ; - ; - ; - ; + ; + ; Yra   ;  7.50
 Muziejus                             ; Uzupio g. 9, Kaunas       ;  2017 ;
Istorija   ; + ; + ; - ; + ; + ; - ; - ; Yra   ;  4.00
 Manto atminimo muziejus              ; Lentvario g. 6, Vilnius   ;  2017 ;
Zoologija  ; - ; - ; - ; - ; - ; + ; + ; Yra   ;  0.00
 Antanas Vatas                  ; Kaunas 44251          ;  2018 ; Martynui
; Aardenis Pavardenis
 Martynas Mažvydas              ; Kaunas 44251          ;  2018 ; Martynui
; Vardenis Pavardenis
 Zulius Asimas                  ; Kaunas 44251          ;  2018 ; Vytautui
; Zardenis Pavardenis
```

VisoVietos.csv

```
Antanas Vatas
Baltu botanikos muziejus
Dailes muziejus
Jotvario kino muziejus
Lietuvos teatro ir muzikos muziejus
Manto atminimo muziejus
Martynas Mažvydas
Muziejus
Senoviniu ginklu muziejus
Senoviniu masinu muziejus
Vytautas Didysis
Vytauto Didziojo muziejus
Zulius Asimas
```

Konsolė

```
Gidus turi 9 muziejus(-u)

Seniausia lankytina vieta yra muziejus:
+-----------------------------------------------------------------------------------------------------------------------+
|            Pavadinimas            |         Adresas          | Metai |   Tipas    | P | A | T | K | P | S | S | Gidas | Kaina |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Baltu botanikos muziejus          | Geliu g. 2, Kaunas       |  1925 | Istorija   | - | + | + | + | - | - | - | Yra   |  3.00 |
+-----------------------------------------------------------------------------------------------------------------------+
```

Duomenys.txt

```
Muziejai:
+-----------------------------------------------------------------------------------------------------------------------+
|            Pavadinimas            |         Adresas          | Metai |   Tipas    | P | A | T | K | P | Š | S | Gidas | Kaina |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Baltu botanikos muziejus          | Geliu g. 2, Kaunas       |  1925 | Istorija   | - | + | + | + | - | - | - | Yra   |  3.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Senoviniu masinu muziejus         | Uzupio g. 9, Kaunas      |  1968 | Istorija   | + | + | - | + | + | - | - | Yra   |  5.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Muziejus                          | Uzupio g. 9, Kaunas      |  2017 | Istorija   | + | + | - | + | + | - | - | Yra   |  4.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Senoviniu ginklu muziejus         | Uzupio g. 9, Kaunas      |  2016 | Istorija   | + | + | - | + | + | - | - | Yra   |  5.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Dailes muziejus                   | Kranto g. 8, Vilnius     |  1979 | Menas      | - | - | + | - | - | + | + | Yra   |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Vytauto Didziojo muziejus         | Tevynes pr. 9, Vilnius   |  2018 | Istorija   | - | - | - | - | - | + | + | Yra   |  7.50 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Lietuvos teatro ir muzikos muziejus | Pilies g. 46, Vilnius  |  1939 | Istorija   | - | + | + | + | + | - | - | Yra   |  2.50 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Jotvario kino muziejus            | Lentvario g. 6, Vilnius  |  1990 | Zoologija  | - | - | - | - | - | + | + | Yra   |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Manto atminimo muziejus           | Lentvario g. 6, Vilnius  |  2017 | Zoologija  | - | - | - | - | - | + | + | Yra   |  0.00 |
+-----------------------------------------------------------------------------------------------------------------------+

Paminklai:
+-----------------------------------------------------------------------------------------------+
|          Pavadinimas          |        Adresas        | Metai |        Skirta        |       Autorius        |
|-------------------------------|-----------------------|-------|----------------------|-----------------------|
| Vytautas Didysis              | Kaunas 44251          |  1932 | Vytautui             | Aardenis Pavardenis   |
|-------------------------------|-----------------------|-------|----------------------|-----------------------|
| Martynas Mažvydas             | Kaunas 44251          |  2018 | Martynui             | Vardenis Pavardenis   |
|-------------------------------|-----------------------|-------|----------------------|-----------------------|
| Zulius Asimas                 | Kaunas 44251          |  2018 | Vytautui             | Zardenis Pavardenis   |
|-------------------------------|-----------------------|-------|----------------------|-----------------------|
| Antanas Vatas                 | Kaunas 44251          |  2018 | Martynui             | Aardenis Pavardenis   |
+-----------------------------------------------------------------------------------------------+
```

Antras testas:

Kaunas.txt

```
Kaunas
Virgis Jovaisa
M; T. Ivanausko zoologijos muziejus; Azuolyno g. 14. Kaunas; 1998; Zoologija; 1;
1; 0; 1; 1; 1; 1; taip; 0,00
M; M. K. Ciurlionio dailes galerija; Vilniaus g. 34, Kaunas; 2018; Menas; 1; 1;
1; 1; 1; 1; 1; ne; 0,00
M; Dailes antarktidos muziejus; Centro g. 82, Kaunas; 1798; Istorija; 1; 1; 0; 1;
1; 1; 1; taip; 5,00
M; Baltu botanikos muziejus; Geliu g. 2, Kaunas; 2018; Istorija; 0; 1; 1; 1; 0;
0; 0; taip; 3,00
M; Senoviniu masinu muziejus; Uzupio g. 9, Kaunas; 1968; Istorija; 1; 1; 0; 1; 1;
0; 0; taip; 5,00
P; Vytautas Didysis; Kaunas 44251; 2018; Vytautui; Vardenis Pavardenis
```

Vilnius.txt

```
Vilnius
Jonas Jonaitis
M; Dailes muziejus; Kranto g. 8, Vilnius; 1979; Menas; 0; 0; 1; 0; 0; 1; 1; taip;
0,00
M; Vytauto Didziojo muziejus; Tevynes pr. 9, Vilnius; 2018; Istorija; 0; 0; 0; 0;
0; 1; 1; taip; 7,50
M; Energetikos ir technikos muziejus; Pilies g. 19, Vilnius; 1979; Istorija; 1;
1; 0; 1; 1; 0; 0; ne; 0,00
P; Lasives paminklas; Pilies g. 2, Vilnius; 2017; Laisvei; Nenurodyta
M; Lietuvos teatro ir muzikos muziejus; Pilies g. 46, Vilnius; 1939; Istorija; 0;
1; 1; 1; 1; 0; 0; taip; 2,50
M; Jotvario kino muziejus; Lentvario g. 6, Vilnius; 2017; Zoologija; 0; 0; 0; 0;
0; 1; 1; taip; 0,00
P; Lasives paminklas; Pilies g. 2, Vilnius; 2002; Laisvei; Nenurodyta
P; Nemuno herbas; Pilies g. 2, Vilnius; 2017; Nemunui; Nenurodyta
P; Antano Markaus; Pilies g. 2, Vilnius; 2018; Antanui; Nenurodyta
```

Panevezys.txt

```
Panevezys
Antanas Valatka
M; Antano Smetonos muziejus; Varpo g. 3, Panevezys; 2018; Istorija; 1; 1; 0; 1;
1; 1; 1; ne; 4,00
M; Rudvilos zoologijos muziejus; Lapo g. 1, Panevezys; 1967; Menas; 1; 1; 1; 1;
1; 1; 1; taip; 5,00
M; Katucio muziejus; Laktos g. 86, Panevezys; 1899; Istorija; 1; 1; 1; 1; 1; 0;
0; taip; 0,00
M; Karaliaus Mindaugo muziejus; Laktosg. 2, Panevezys; 2017; Menas; 0; 0; 0; 0;
0; 1; 1; taip; 0,00
M; Vytauto Didziojo karo muziejus; Girciupio g. 5, Panevezys; 1898; Istorija; 0;
0; 1; 0; 0; 1; 1; ne; 7,50
P; Panevežio katedra; Kaunas 44251; 2018; Panevežiui; Vardenis Pavardenis
```

Nauji.csv

```
Vytauto Didziojo muziejus            ; Tevynes pr. 9, Vilnius    ;  2018 ;
Istorija   ; - ; - ; - ; - ; - ; + ; + ; Yra   ;  7.50
 Antano Smetonos muziejus             ; Varpo g. 3, Panevezys     ;  2018 ;
Istorija   ; + ; + ; - ; + ; + ; + ; + ; Nėra  ;  4.00
 Baltu botanikos muziejus             ; Geliu g. 2, Kaunas        ;  2018 ;
Istorija   ; - ; + ; + ; + ; - ; - ; - ; Yra   ;  3.00
 Jotvario kino muziejus               ; Lentvario g. 6, Vilnius   ;  2017 ;
Zoologija  ; - ; - ; - ; - ; - ; + ; + ; Yra   ;  0.00
 Karaliaus Mindaugo muziejus          ; Laktosg. 2, Panevezys     ;  2017 ; Menas
; - ; - ; - ; - ; - ; + ; + ; Yra   ;  0.00
 M. K. Ciurlionio dailes galerija     ; Vilniaus g. 34, Kaunas    ;  2018 ; Menas
; + ; + ; + ; + ; + ; + ; + ; Nėra  ;  0.00
 Antano Markaus                ; Pilies g. 2, Vilnius  ;  2018 ; Antanui
; Nenurodyta
```

```
 Paneve▯io katedra              ; Kaunas 44251          ;  2018 ; Paneve▯iui
; Vardenis Pavardenis
 Vytautas Didysis               ; Kaunas 44251          ;  2018 ; Vytautui
; Vardenis Pavardenis
```

VisosVietos.csv

```
Antano Markaus
Antano Smetonos muziejus
Baltu botanikos muziejus
Dailes antarktidos muziejus
Dailes muziejus
Energetikos ir technikos muziejus
Jotvario kino muziejus
Karaliaus Mindaugo muziejus
Katucio muziejus
Lasives paminklas
Lasives paminklas
Lietuvos teatro ir muzikos muziejus
M. K. Ciurlionio dailes galerija
Nemuno herbas
Paneve▯io katedra
Rudvilos zoologijos muziejus
Senoviniu masinu muziejus
T. Ivanausko zoologijos muziejus
Vytautas Didysis
Vytauto Didziojo karo muziejus
Vytauto Didziojo muziejus
```

Konsolė

```
Gidus turi 11 muziejus(-u)

Seniausia lankytina vieta yra muziejus:
+-----------------------------------------------------------------------------------------------------------------+
|           Pavadinimas           |         Adresas          | Metai |   Tipas    | P | A | T | K | P | S | S | Gidas | Kaina |
|---------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Dailes antarktidos muziejus     | Centro g. 82, Kaunas     |  1798 | Istorija   | + | + | - | + | + | + | + | Yra   |  5.00 |
+-----------------------------------------------------------------------------------------------------------------+
```

Duomenys.txt

```
Muziejai:
+-----------------------------------------------------------------------------------------------------------------+
|            Pavadinimas            |         Adresas          | Metai |   Tipas    | P | A | T | K | P | Š | S | Gidas | Kaina |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| T. Ivanausko zoologijos muziejus  | Azuolyno g. 14. Kaunas   |  1998 | Zoologija  | + | + | - | + | + | + | + | Yra   |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| M. K. Ciurlionio dailes galerija  | Vilniaus g. 34, Kaunas   |  2018 | Menas      | + | + | + | + | + | + | + | Nėra  |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Dailes antarktidos muziejus       | Centro g. 82, Kaunas     |  1798 | Istorija   | + | + | - | + | + | + | + | Yra   |  5.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Baltu botanikos muziejus          | Geliu g. 2, Kaunas       |  2018 | Istorija   | - | + | + | + | - | - | - | Yra   |  3.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Senoviniu masinu muziejus         | Uzupio g. 9, Kaunas      |  1968 | Istorija   | + | + | - | + | + | + | - | Yra   |  5.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Antano Smetonos muziejus          | Varpo g. 3, Panevezys    |  2018 | Istorija   | + | + | - | + | + | + | + | Nėra  |  4.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Rudvilos zoologijos muziejus      | Lapo g. 1, Panevezys     |  1967 | Menas      | + | + | + | + | + | + | + | Yra   |  5.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Katucio muziejus                  | Laktos g. 86, Panevezys  |  1899 | Istorija   | + | + | + | + | + | + | - | Yra   |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Karaliaus Mindaugo muziejus       | Laktosg. 2, Panevezys    |  2017 | Menas      | - | - | - | - | - | + | + | Yra   |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Vytauto Didziojo karo muziejus    | Girciupio g. 5, Panevezys|  1898 | Istorija   | - | - | + | - | - | + | + | Nėra  |  7.50 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Dailes muziejus                   | Kranto g. 8, Vilnius     |  1979 | Menas      | - | - | + | - | - | + | + | Yra   |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Vytauto Didziojo muziejus         | Tevynes pr. 9, Vilnius   |  2018 | Istorija   | - | - | - | - | - | + | + | Yra   |  7.50 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Energetikos ir technikos muziejus | Pilies g. 19, Vilnius    |  1979 | Istorija   | + | + | - | + | + | - | - | Nėra  |  0.00 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Lietuvos teatro ir muzikos muziejus | Pilies g. 46, Vilnius  |  1939 | Istorija   | - | + | + | + | + | - | - | Yra   |  2.50 |
|-----------------------------------|--------------------------|-------|------------|---|---|---|---|---|---|---|-------|-------|
| Jotvario kino muziejus            | Lentvario g. 6, Vilnius  |  2017 | Zoologija  | - | - | - | - | - | + | + | Yra   |  0.00 |
+-----------------------------------------------------------------------------------------------------------------+

Paminklai:
+-----------------------------------------------------------------------------------------------+
|          Pavadinimas          |        Adresas        | Metai |        Skirta        |        Autorius        |
|-------------------------------|-----------------------|-------|----------------------|------------------------|
| Vytautas Didysis              | Kaunas 44251          |  2018 | Vytautui             | Vardenis Pavardenis    |
|-------------------------------|-----------------------|-------|----------------------|------------------------|
| Paneve▯io katedra             | Kaunas 44251          |  2018 | Paneve▯iui           | Vardenis Pavardenis    |
|-------------------------------|-----------------------|-------|----------------------|------------------------|
| Lasives paminklas             | Pilies g. 2, Vilnius  |  2017 | Laisvei              | Nenurodyta             |
```

Lasives paminklas	Pilies g. 2, Vilnius	2002	Laisvei	Nenurodyta
Nemuno herbas	Pilies g. 2, Vilnius	2017	Nemunui	Nenurodyta
Antano Markaus	Pilies g. 2, Vilnius	2018	Antanui	Nenurodyta

5.4. Dėstytojo pastabos