5. Susieti rinkiniai

Susipažinsite su:

- dvimatės konteinerinės klasės aprašymu;
- matricos elementų reikšmių įvedimu, spausdinimu, sumos, kiekio ir vidurkio skaičiavimu;
- matricos eilučių/stulpelių elementų reikšmių sumos, kiekio, didžiausios reikšmės radimu;
- susietais masyvais.

5.1. Veiksmai su sveikųjų skaičių dvimačiu masyvu

- Konteinerinės klasės aprašymas.
- Duomenų skaitymas ir spausdinimas.
- Veiksmai su visomis konteinerio reikšmėmis.

Užduotis. Prekybos bazė.
Tekstiniame faile yra surašyti prekybos bazės kasose per tam tikrą laikotarpį aptarnautų pirkėjų skaičiai. Pirmoje failo eilutėje yra užrašyti du skaičiai: n - kasų skaičius ir m - dienų skaičius. Tolesnėse n eilutėse užrašyta po m skaičių - atitinkamai kasose aptarnautų pirkėjų skaičiai. Parašykite programą, kuri įvestų duomenis iš failo, juos išspausdintų faile ir suskaičiuotų, kiek iš viso buvo aptarnauta pirkėjų.

Pradiniai duomenys ir rezultatai.

Pradiniai duomenys	Rezultatai
6 7 5; 9; 8; 7; 3; 5; 6; 6; 9; 8; 2; 1; 5; 2; 8; 9; 0; 8; 8; 8; 8; 4; 5; 6; 2; 3; 4; 3; 5; 6; 9; 1; 0; 5; 8; 3; 4; 5; 4; 6; 5; 7;	Pradiniai duomenys Kasų kiekis 6 Darbo dienų kiekis 7 Aptarnautų klientų kiekiai 5 9 8 7 3 5 6 6 9 8 2 1 5 2 8 9 0 8 8 8 8 4 5 6 2 3 4 3 5 6 9 1 0 5 8 3 4 5 4 6 5 7 Rezultatai Viso aptarnauta: 220 klientų.

Programos kūrimo eiga.

- Paruošiamas pradinių duomenų failas.
- Sukuriama klasė dvimačio sveikųjų skaičių masyvo duomenims saugoti.
- Pagrindiniame metode `Main()` skelbiamas objektas, skirtas prekybos bazės kasose aptarnautų pirkėjų skaičiams saugoti.
- Sukuriamas metodas duomenims iš failo skaityti.
- Sukuriamas metodas pradiniams duomenims faile spausdinti.
- Sukuriamas metodas prekybos bazėje aptarnautų pirkėjų skaičiui rasti.

Pirmas žingsnis.

- Sukurkite klasę, skirtą dvimačiam sveikųjų skaičių masyvui – prekybos bazės kasose aptarnautų pirkėjų skaičiams saugoti:

```
//------------------------------------------------------------
/** Klasė kasų duomenims saugoti
    @class Matrica */
    class Matrica
    {
        const int CMaxEil = 10;     // didžiausias galimas eilučių (kasų) skaičius
        const int CMaxSt = 100;     // didžiausias galimas stulpelių (pirkėjų) skaičius
        private int[,] A;           // duomenų matrica
        public int n { get; set; }  // eilučių skaičius (kasų skaičius)
```

```
        public int m { get; set; }  // stulpelių skaičius (dienų skaičius)

        //-----------------------------------------------------
        /** Pradinių matricos duomenų nustatymas          */
        //-----------------------------------------------------
        public Matrica()
        {
            n = 0;
            m = 0;
            A = new int[CMaxEil, CMaxSt];
        }

        //-----------------------------------------------------
        /** Priskiria klasės matricos kintamajam reikšmę.
        @param i - eilutės (kasos) indeksas
        @param j - stulpelio (dienos) indeksas
        @param pirk - pirkėjų skaičius */
        //-----------------------------------------------------
        public void Deti(int i, int j, int pirk)
        {
            A[i, j] = pirk;
        }

        //-----------------------------------------------------
        /** Grąžina pirkėjų kiekį.
        @param i - eilutės (kasos) indeksas
        @param j - stulpelio (dienos) indeksas */
        //-----------------------------------------------------
        public int ImtiReiksme(int i, int j)
        {
            return A[i, j];
        }
    }
//-------------------------------------------------------------
```

- Parašykite pagrindinį metodą `Main()`, kurioje būtų paskelbtas prekybos bazėje aptarnautų pirkėjų skaičių objektas.

```
//-------------------------------------------------------------
class Program
    {
        static void Main(string[] args)
        {
            Matrica prekybosBaze = new Matrica();

            Console.WriteLine("Programa baigė darbą!");
        }
    }
//-------------------------------------------------------------
```

- Patikrinkite, kaip programa dirba. Ekrane turite matyti:

```
Programa baigė darbą!
```

Antras žingsnis.

- Parašykite metodą, kuris užpildytų objektą – konteinerį duomenimis iš failo:

```
//-------------------------------------------------------------
        /** Failo duomenis surašo į konteinerį.
        @param fd           - duomenų failo vardas
        @param prekybosBaze - dvimatis konteineris */
        //-------------------------------------------------------
        static void Skaityti(string fd, ref Matrica prekybosBaze)
        {
            int nn, mm, skaic;
            string line;
            using (StreamReader reader = new StreamReader(fd))
            {
                line = reader.ReadLine();
                string[] parts;
```

```
                nn = int.Parse(line);
                line = reader.ReadLine();
                mm = int.Parse(line);
                prekybosBaze.n = nn;
                prekybosBaze.m = mm;
                for (int i = 0; i < nn; i++)
                {
                    line = reader.ReadLine();
                    parts = line.Split(';');
                    for (int j = 0; j < mm; j++)
                    {
                        skaic = int.Parse(parts[j]);
                        prekybosBaze.Deti(i, j, skaic);
                    }
                }
            }
        }

//---------------------------------------------------------------
```

- Parašykite metodą, kuris išspausdintų objekto duomenis faile:

```
//---------------------------------------------------------------
        /** Spausdina konteinerio duomenis faile.
        @param fv           - rezultatų failo vardas
        @param prekybosBaze - matricos konteineris
        @param antraste     - užrašas virš lentelės */
        //-----------------------------------------------------------
        static void Spausdinti(string fv, Matrica prekybosBaze, string antraštė)
        {
            using (var fr = File.AppendText(fv))
            {
                fr.WriteLine(antraštė);
                fr.WriteLine();
                fr.WriteLine(" Kasų kiekis {0}", prekybosBaze.n);
                fr.WriteLine(" Darbo dienų kiekis {0}", prekybosBaze.m);
                fr.WriteLine(" Aptarnautų klientų kiekiai");
                for (int i = 0; i < prekybosBaze.n; i++)
                {
                    for (int j = 0; j < prekybosBaze.m; j++)
                        fr.Write("{0,4:d}", prekybosBaze.ImtiReiksme(i, j));
                    fr.WriteLine();
                }
            }
        }

//---------------------------------------------------------------
```

- Papildykite programą duomenų įvedimo iš failo `Duomenys.txt` ir spausdinimo faile `Rezultatai.txt` veiksmais – užrašykite kreipinius į sukurtus metodus:

```
//---------------------------------------------------------------
class Program
    {
        const string CFd = "..\\..\\Duomenys.txt";
        const string CFr = "..\\..\\Rezultatai.txt";

        static void Main(string[] args)
        {
            Matrica prekybosBaze = new Matrica();
            Skaityti(CFd, ref prekybosBaze);
            if (File.Exists(CFr))
                File.Delete(CFr);
            Spausdinti(CFr, prekybosBaze, " Pradiniai duomenys");

            Console.WriteLine("Programa baigė darbą!");
        }
//---------------------------------------------------------------
```

- Išbandykite, kaip veikia programa. Rezultatų faile turite matyti:

```
Pradiniai duomenys

Kasų kiekis 6
Darbo dienų kiekis 7
Aptarnautų klientų kiekiai
  5   9   8   7   3   5   6
  6   9   8   2   1   5   2
  8   9   0   8   8   8   8
  4   5   6   2   3   4   3
  5   6   9   1   0   5   8
  3   4   5   4   6   5   7
```

Trečias žingsnis.

- Parašykite metodą, kuris suskaičiuotų, kiek iš viso pirkėjų aptarnavo prekybos bazė:

```
//------------------------------------------------------------
        /** Suskaičiuoja ir grąžina prekybos bazėje aptarnautų pirkėjų skaičių.
        @param  A - konteinerio vardas */
        //------------------------------------------------------------
        static int VisoAptarnauta(Matrica A)
        {
            int suma = 0;
            for (int i = 0; i < A.n; i++)
                for (int j = 0; j < A.m; j++)
                    suma = suma + A.ImtiReiksme(i, j);
            return suma;
        }
//------------------------------------------------------------
```

- Papildykite programą: atverkite rezultatų failą papildymui ir užrašykite kreipinį į aukščiau sukurtą metodą:

```
using (var fr = File.AppendText(CFr))
{
    fr.WriteLine();
    fr.WriteLine(" Rezultatai");
    fr.WriteLine();
    fr.WriteLine(" Viso aptarnauta: {0} klientų.",
                 VisoAptarnauta(prekybosBaze));
}
```

- Išbandykite programą. Rezultatų faile, be pradinių duomenų, matysite, kad iš viso buvo aptarnauta 220 pirkėjų.

Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitus duomenų failus. Patikrinkite, kaip dirba programa, kai prekybos bazėje:

a) viena kasa (`n = 1`);
b) `n` kasų, o aptarnavimo laikotarpis viena diena (`m = 1`);
c) viena kasa (`n = 1`), o aptarnavimo laikotarpis viena diena (`m = 1`).

Programos papildymas.

Papildykite spausdinimo metodą veiksmais, kurie padarytų skaičiavimo rezultatus vaizdesniais: juos įrėmintų, o taip pat užrašytų eilučių (kasų) numerius ir stulpelių (dienų) numerius.

Savarankiško darbo užduotis.

Parašykite ir išbandykite metodą, kuris suskaičiuotų:

a) kiek vidutiniškai pirkėjų aptarnavo viena kasa per vieną dieną;
b) kiek dienų kuri iš kasų nedirbo (duomenyse yra skaičius 0).

5.2. Veiksmai su sveikųjų skaičių dvimačio masyvo eilutėmis ir stulpeliais

- Veiksmai su konteinerio eilutėmis.
- Veiksmai su konteinerio stulpeliais.
- Didžiausios reikšmės paieška.

Užduotis.
Papildykite ankstesnio pratimo programą veiksmais, kurie apskaičiuotų:

- kiek pirkėjų aptarnavo kiekviena kasa;
- kiek pirkėjų buvo aptarnauta kiekvieną dieną;
- kuri kasa, kurią dieną, aptarnavo daugiausia pirkėjų.

Pirmas žingsnis.

- Parašykite metodą, kuris suskaičiuotų ir išspausdintų, kiek pirkėjų aptarnavo kiekviena kasa:

```
//------------------------------------------------------------
        /** Suskaičiuoja ir išspausdina, kiek pirkėjų aptarnavo kiekviena kasa
        @param  CFr - rezultatų failo vardas
        @param  A   - konteinerio vardas   */
        //------------------------------------------------------------
        static void KiekvienaKasaAptarnavo(string CFr, Matrica A)
        {
            using (var fr = File.AppendText(CFr))
            {
                for (int i = 0; i < A.n; i++)
                {
                    int suma = 0;
                    for (int j = 0; j < A.m; j++)
                        suma = suma + A.ImtiReiksme(i, j);
                    fr.WriteLine(" Kasa nr. {0} aptarnavo {1} klientų.", i + 1, suma);
                }
            }
        }
//------------------------------------------------------------
```

- Papildykite programą skaičiavimais – kreipiniu į metodą:

```
KiekvienaKasaAptarnavo(CFr, prekybosBaze);
```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

```
Kasa nr. 1 aptarnavo 43 klientų.
Kasa nr. 2 aptarnavo 33 klientų.
Kasa nr. 3 aptarnavo 49 klientų.
Kasa nr. 4 aptarnavo 27 klientų.
Kasa nr. 5 aptarnavo 34 klientų.
Kasa nr. 6 aptarnavo 34 klientų.
```

Antras žingsnis.

- Parašykite metodą, kuris suskaičiuotų ir išspausdintų, kiek pirkėjų buvo aptarnauta kiekvieną dieną:

```
//------------------------------------------------------------
        /** Suskaičiuoja ir išspausdina, kiek pirkėjų buvo aptarnauta kiekvieną dieną
        @param CFr - rezultatų failo vardas
        @param A   - konteinerio vardas */
        //------------------------------------------------------------
        static void KiekvienąDienąAptarnauta(string CFr, Matrica A)
        {
            using (var fr = File.AppendText(CFr))
            {
                fr.WriteLine();
                for (int j = 0; j < A.m; j++)
                {
                    int suma = 0;
                    for (int i = 0; i < A.n; i++)
                        suma = suma + A.ImtiReiksme(i, j);
                    fr.WriteLine(" Diena nr. {0}: aptarnauta klientų - {1}.", j + 1, suma);
                }
            }
        }
//------------------------------------------------------------
```

- Papildykite programą kreipiniu į metodą:

```
KiekvienąDienąAptarnauta(CFr, prekybosBaze);
```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

```
Diena nr. 1: aptarnauta klientų - 31.
Diena nr. 2: aptarnauta klientų - 42.
Diena nr. 3: aptarnauta klientų - 36.
Diena nr. 4: aptarnauta klientų - 24.
Diena nr. 5: aptarnauta klientų - 21.
Diena nr. 6: aptarnauta klientų - 32.
Diena nr. 7: aptarnauta klientų - 34.
```

Trečias žingsnis.

- Pirmo ir antro žingsnio metoduose skaičiavimo metoduose yra ir atsakymų spausdinimas. Užrašykite nurodytų metodų užklotus metodus, kai atsakymai surašomi į masyvą.

```
//------------------------------------------------------------
        /** Suskaičiuoja ir įrašo į masyvą, kiek pirkėjų aptarnavo kiekviena kasa
        @param  A     - konteinerio vardas
        @param  Sumos - suskaičiuotų sumų masyvas*/
        //------------------------------------------------------------
        static void KiekvienaKasaAptarnavo(Matrica A, int [] Sumos)
        {
            for (int i = 0; i < A.n; i++)
            {
                int suma = 0;
                for (int j = 0; j < A.m; j++)
                    suma = suma + A.ImtiReiksme(i, j);
                Sumos[i] = suma;
            }
        }
        //------------------------------------------------------------
        /** Suskaičiuoja ir įrašo į masyvą, kiek pirkėjų buvo aptarnauta kiekvieną dieną
        @param  A      - konteinerio vardas
        @param  Sumos  - suskaičiuotų sumų masyvas*/
        //------------------------------------------------------------
        static void KiekvienąDienąAptarnauta(Matrica A, int [] Sumos)
        {
            for (int j = 0; j < A.m; j++)
            {
                int suma = 0;
                for (int i = 0; i < A.n; i++)
                    suma = suma + A.ImtiReiksme(i, j);
                Sumos[j] = suma;
            }
        }
```

- Užrašykite suformuotų masyvų spausdinimo metodą.

```
        /** Išspausdina, kiek pirkėjų buvo aptarnauta kiekvieną dieną  arba
         *  kiek pirkėjų aptarnavo kiekviena kasa
        @param  CFr   - rezultatų failo vardas
        @param  n     - sumų masyvo dydis
        @param  pav   - parametras, nurodantis dienų ar kasų informaciją reikia spausdinti
        @param  Sumos - suskaičiuotų sumų masyvas*/
        //------------------------------------------------------------
        static void SpausdintiSumas(string CFr, int[] Sumos, int n, string pav)
        {
            using (var fr = File.AppendText(CFr))
            {
                fr.WriteLine();
                for (int i = 0; i < n; i++)
                {
                    if (pav == "Dienos")
                        fr.WriteLine(" Diena nr. {0}: aptarnauta klientų - {1}.", i + 1,
                                     Sumos[i]);
                    else
                        fr.WriteLine(" Kasa nr. {0} aptarnavo {1} klientų.", i + 1,
                                     Sumos[i]);
                }
            }
        }
```

- Papildykite programą naujais masyvais. Pataisykite programą kreipiniais į naujus metodus:

```
int[] KasuSumos = new int[prekybosBaze.n];
int[] DienuSumos = new int[prekybosBaze.m];

KiekvienaKasaAptarnavo(prekybosBaze, KasuSumos);
SpausdintiSumas(CFr, KasuSumos, prekybosBaze.n, "Kasos");

KiekvienąDienąAptarnauta(prekybosBaze, DienuSumos);
SpausdintiSumas(CFr, DienuSumos, prekybosBaze.m, "Dienos");
```

- Išbandykite programą. Palyginkite gautus atsakymus su pirmame ir antrame žingsniuose gautais.

Ketvirtas žingsnis.

- Parašykite metodą, kuris surastų, kuri kasa aptarnavo daugiausiai pirkėjų:

```
//-------------------------------------------------------------
        /** Suranda ir grąžina, kuri kasa patarnavo daugiausiai pirkėjų
        @param A – konteinerio vardas */
        //-------------------------------------------------------------
        static int KasosNumerisMaxPirkėjų(Matrica A)
        {
            int max = 0;
            int nr  = 0;
            for (int i = 0; i < A.n; i++)
            {
                int suma = 0;
                for (int j = 0; j < A.m; j++)
                     suma = suma + A.ImtiReiksme(i, j);
                if (suma > max)
                {
                     max = suma;
                     nr = i+1;
                }
            }
            return nr;
        }
//-------------------------------------------------------------
```

- Papildykite programą kreipiniu į metodą (po spausdinimo `Viso aptarnauta:`):

```
fr.WriteLine(" Daugiausia pirkėjų aptarnavo (kasa): {0}",
             KasosNumerisMaxPirkėjų(prekybosBaze));
```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

```
Daugiausia pirkėjų aptarnavo (kasa): 3
```

Tuo galima įsitikinti ir iš prieš tai gautų rezultatų.

Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitus duomenų failus. Patikrinkite, kaip dirba programa, kai prekybos bazėje:

a) viena kasa (`n = 1`);
b) n kasų, o aptarnavimo laikotarpis viena diena (`m = 1`);
c) viena kasa (`n = 1`), o aptarnavimo laikotarpis viena diena (`m = 1`).

Programos papildymas.

Pakeiskite metodą `KasosNumerisMaxPirkėjų()` taip, kad ji surastų ir aptarnautų pirkėjų skaičių. Atitinkamai pakeiskite ir kreipinį metodą.

Savarankiško darbo užduotis.

Parašykite ir išbandykite metodą, kuris suskaičiuotų:

a) kurią dieną buvo aptarnauta mažiausiai pirkėjų ir keli pirkėjai buvo aptarnauti tą dieną;
b) kiek pirkėjų vidutiniškai aptarnavo kiekviena kasa.

5.3. Dvimatis objektų masyvas

- Objektai konteineryje.
- Veiksmai su objektais visame konteineryje.
- Objekto pasiekiamumas.

Užduotis. Šeimos išlaidos.
Tekstiniame faile yra surašyta šeimos nario tam tikro laikotarpio (pvz.: pusės metų) išlaidos pirkiniams kiekvieną dieną savaitėmis (`7` dienos). Pirmoje failo eilutėje yra užrašytas savaičių skaičius `n` ir skaičius `7`. Tolesnėse n eilutėse yra užrašyta po `7` poras: šeimos nario pavadinimas (`vyras` arba `žmona`) ir išlaidos (jei išlaidų tą dieną nebuvo: simboliai `"-----"` ir skaičius `0.0`).
Parašykite programą, kuri įvestų duomenis iš failo, išspausdintų faile ir apskaičiuotų, kiek iš viso šeima turėjo išlaidų.

Pradiniai duomenys ir rezultatai.

Pradiniai duomenys

```
3
7
vyras; 10,40;vyras; 15,20;žmona; 50,50;žmona; 100,20;-----; 0,0;žmona; 10,20;-----; 0,0;
žmona; 15,30;-----; 0,0;-----;0,0;žmona; 20,50;vyras; 55,50;vyras; 10,10;žmona; 30,30;
vyras; 10,10;vyras; 20,20;vyras; 30,30;vyras; 50,50;vyras; 20,10;vyras; 30,10;vyras; 30,10;
```

Rezultatai

```
Pradiniai duomenys

Savaičių kiekis 3
Dienų kiekis 7

1-dienis       2-dienis       3-dienis       4-dienis       5-dienis       6-dienis       7-dienis
vyras  10,40   vyras  15,20   žmona  50,50   žmona 100,20   -----   0,00   žmona  10,20   -----   0,00
žmona  15,30   -----   0,00   -----   0,00   žmona  20,50   vyras  55,50   vyras  10,10   žmona  30,30
vyras  10,10   vyras  20,20   vyras  30,30   vyras  50,50   vyras  20,10   vyras  30,10   vyras  30,10

Rezultatai

Viso išleista: 509,60 €.
```

Programos kūrimo eiga.

- Paruošiamas pradinių duomenų failas.
- Sukuriama klasė `Asmuo`, skirta simbolių eilutei (`string`) ir realiam skaičiui (`double`) saugoti.
- Sukuriama klasė `Matrica` klasės `Asmuo` objektams saugoti.
- Pagrindiniame metode `Main()` skelbiamas objektas, skirtas šeimos išlaidoms saugoti.
- Sukuriamas metodas duomenims iš failo skaityti.
- Sukuriamas metodas pradiniams duomenims rezultatų faile spausdinti.
- Sukuriamas metodas visoms šeimos išlaidoms skaičiuoti.

Pirmas žingsnis.

- Sukurkite klasę, skirtą simbolių eilutei (`string`) ir realiam skaičiui (`double`) – šeimos asmens duomenims saugoti:

```
//------------------------------------------------------------
/** Klasė asmens duomenims saugoti
    @class Asmuo */
    class Asmuo
    {
        private string vardas;            // pirkusio asmens vardas
        private double pinigai;           // išlaidos per dieną
        //-----------------------------------------------
        /** Asmens duomenys
        @param vardas   - pirkusio asmens vardas
        @param pinigai  - išleistų pinigų reikšmė  */
        //-----------------------------------------------
        public Asmuo(string vardas, double pinigai)
        {
            this.vardas = vardas;
```

```
            this.pinigai = pinigai;
        }

        /** grąžina pirkusio asmens vardą */
        public string ImtiVarda() { return vardas; }

        /** grąžina išlaidų kiekį */
        public double ImtiPinigus() { return pinigai; }
    }
//---------------------------------------------------------------
```

- Sukurkite klasę (galite klasę kopijuoti iš ankstesnio darbo ir po to ją nežymiai modifikuoti), skirtą dvimačiam klasės `Asmuo` objektų masyvui – šeimos išlaidoms saugoti:

```
//---------------------------------------------------------------
/** Klasė šeimos duomenims saugoti
    @class Matrica */
    class Matrica
    {
        const int CMaxEil = 100;    // didžiausias galimas savaičių skaičius
        const int CMaxSt = 7;       // didžiausias galimas stulpelių (dienų) skaičius
        private Asmuo[,] A;         // duomenų matrica
        public int n { get; set; }  // eilučių skaičius (savaičių skaičius)
        public int m { get; set; }  // stulpelių skaičius (dienų skaičius)

        //-----------------------------------------------------
        /** Pradinių matricos duomenų nustatymas          */
        //-----------------------------------------------------
        public Matrica()
        {
            n = 0;
            m = 0;
            A = new Asmuo[CMaxEil, CMaxSt];
        }

        //-----------------------------------------------------
        /** Priskiria klasės matricos kintamajam reikšmę.
        @param i - eilutės (savaitės) indeksas
        @param j - stulpelio (dienos) indeksas
        @param islaidos - išlaidos atitinkamą dieną */
        //-----------------------------------------------------
        public void Deti(int i, int j, Asmuo asmuo)
        {
            A[i, j] = asmuo;
        }

        //-----------------------------------------------------
        /** Grąžina išlaidų kiekį.
        @param i - eilutės (kasos) indeksas
        @param j - stulpelio (dienos) indeksas */
        //-----------------------------------------------------
        public Asmuo ImtiReiksme(int i, int j)
        {
            return A[i, j];
        }
    }
//---------------------------------------------------------------
```

- Parašykite pagrindinį metodą `Main()`, kurioje būtų šeimos išlaidas aprašantis objektas.

```
//---------------------------------------------------------------
class Program
    {
        static void Main(string[] args)
        {
            Matrica seimosIslaidos = new Matrica();
            Console.WriteLine("Programa baigė darbą!");
        }
    }
//---------------------------------------------------------------
```

- Patikrinkite, kaip programa dirba. Ekrane turite matyti:

```
Programa baigė darbą!
```

Antras žingsnis.

- Parašykite metodą, kuris užpildytų objektą duomenimis iš failo:

```
//-------------------------------------------------------------
        /** Failo duomenis surašo į konteinerį.
        @param fd             - duomenų failo vardas
        @param seimosIslaidos - dvimatis konteineris */
        //-------------------------------------------------------------
        static void Skaityti(string fd, ref Matrica seimosIslaidos)
        {
            int nn, mm;
            double pinigai;
            string line, vardas;
            Asmuo asmuo;
            using (StreamReader reader = new StreamReader(fd))
            {
                line = reader.ReadLine();
                string[] parts;
                nn = int.Parse(line);
                line = reader.ReadLine();
                mm = int.Parse(line);
                seimosIslaidos.n = nn;
                seimosIslaidos.m = mm;
                for (int i = 0; i < nn; i++)
                {
                    line = reader.ReadLine();
                    parts = line.Split(';');
                    for (int j = 0; j < mm; j++)
                    {
                        vardas = parts[2 * j];
                        pinigai = double.Parse(parts[2 * j + 1]);
                        asmuo = new Asmuo(vardas, pinigai);
                        seimosIslaidos.Deti(i, j, asmuo);
                    }
                }
            }
        }
//-------------------------------------------------------------
```

- Parašykite metodą, kuris išspausdintų objekto duomenis faile:

```
//-------------------------------------------------------------
        /** Spausdina konteinerio duomenis faile.
        @param fv             - rezultatų failo vardas
        @param seimosIslaidos - matricos konteineris
        @param antraste       - užrašas virš lentelės */
        //-------------------------------------------------------------
        static void Spausdinti(string fv, Matrica seimosIslaidos, string antraštė)
        {
            Asmuo asmuo;
            using (var fr = File.AppendText(fv))
            {
                fr.WriteLine(antraštė);
                fr.WriteLine();
                fr.WriteLine("Savaičių kiekis {0}", seimosIslaidos.n);
                fr.WriteLine("Dienų kiekis {0}", seimosIslaidos.m);
                fr.WriteLine();

                for (int j = 0; j < seimosIslaidos.m; j++)
                    fr.Write("{0}-dienis       ", j+1);
                fr.WriteLine();

                for (int i = 0; i < seimosIslaidos.n; i++)
                {
                    for (int j = 0; j < seimosIslaidos.m; j++)
                    {
```

```
                    asmuo = seimosIslaidos.ImtiReiksme(i, j);
                    fr.Write("{0} {1,6:f2}   ", asmuo.ImtiVarda(), asmuo.ImtiPinigus());
                }
                fr.WriteLine();
            }
        }
    }
//---------------------------------------------------------------
```

- Papildykite programą duomenų įvedimo iš failo `Duomenys.txt` ir spausdinimo rezultatų faile `Rezultatai.txt` veiksmais – užrašykite kreipinius į sukurtus metodus:

```
//---------------------------------------------------------------
class Program
    {
        const string CFd = "..\\..\\Duomenys.txt";
        const string CFr = "..\\..\\Rezultatai.txt";

        static void Main(string[] args)
        {
            Matrica seimosIslaidos = new Matrica();
            Skaityti(CFd, ref seimosIslaidos);
            if (File.Exists(CFr))
                File.Delete(CFr);
            Spausdinti(CFr, seimosIslaidos, "Pradiniai duomenys");

            Console.WriteLine("Pradiniai duomenys išspausdinti faile: {0}", CFr);
            Console.WriteLine("Programa baigė darbą!");
        }
//---------------------------------------------------------------
```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

```
Pradiniai duomenys išspausdinti faile: ..\..\ Rezultatai.txt
Programa baigė darbą!
```

- Rezultatų faile `Rezultatai.txt` bus išspausdintos šeimos išlaidos:

```
Pradiniai duomenys

Savaičių kiekis 3
Dienų kiekis 7

1-dienis       2-dienis       3-dienis       4-dienis       5-dienis       6-dienis       7-dienis
vyras  10,40   vyras  15,20   žmona  50,50   žmona 100,20   -----   0,00   žmona  10,20   -----   0,00
žmona  15,30   -----   0,00   -----   0,00   žmona  20,50   vyras  55,50   vyras  10,10   žmona  30,30
vyras  10,10   vyras  20,20   vyras  30,30   vyras  50,50   vyras  20,10   vyras  30,10   vyras  30,10
```

Trečias žingsnis.

- Parašykite metodą, kuris suskaičiuotų, kiek iš viso šeima turėjo išlaidų:

```
//---------------------------------------------------------------
        /** Suskaičiuoja ir grąžina šeimos visas išlaidas.
        @param  A - konteinerio vardas */
        //-----------------------------------------------------------
        static decimal VisosIslaidos(Matrica A)
        {
            Asmuo asmuo;
            double suma = 0;
            for (int i = 0; i < A.n; i++)
                for (int j = 0; j < A.m; j++)
                {
                    asmuo = A.ImtiReiksme(i, j);
                    suma = suma + asmuo.ImtiPinigus();
                }
            return (decimal)suma;
        }
//---------------------------------------------------------------
```

- Papildykite programą: atverkite rezultatų failą papildymui ir užrašykite kreipinį į aukščiau sukurtą metodą:

```
            using (var fr = File.AppendText(CFr))
            {
                fr.WriteLine();
                fr.WriteLine("Rezultatai");
                fr.WriteLine();
                fr.WriteLine("Viso išleista: {0,5:c2}.",
                             VisosIslaidos(seimosIslaidos));
            }
```

- Išbandykite programą.

Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitą duomenų failą. Patikrinkite, kaip dirba programa, kai šeimos išlaidos skaičiuojamos vieną savaitę (`n = 1`).

Programos papildymas.

Papildykite spausdinimo metodą veiksmais, kurie padarytų skaičiavimo rezultatus vaizdesniais: juos įrėmintų, o taip pat užrašytų eilučių (savaičių) numerius.

Užduotis savarankiškam darbui.

Parašykite ir išbandykite metodą, kuris suskaičiuotų:

a) kelias dienas šeima neturėjo išlaidų (duomenyse skaičius lygus 0.0);
b) kiek išlaidų turėjo vienas šeimos narys (bus reikalingi du kreipiniai į šį metodą: žmonos ir vyro išlaidoms skaičiuoti).

5.4. Veiksmai su objektais dvimačio masyvo eilutėse ir stulpeliuose

- Veiksmai su objektais konteinerio eilutėje.
- Veiksmai su objektais konteinerio stulpelyje.
- Veiksmai su konteinerio objektu.

Užduotis.
Papildykite ankstesnio pratimo programą, kuri papildomai apskaičiuotų:

- kiek išlaidų šeima turėjo kiekvieną savaitę;
- kiek išlaidų šeima turėjo nurodytomis savaitės dienomis, pavyzdžiui antradieniais, šeštadieniais ir t. t.;
- kuris šeimos narys per vieną dieną išleido didžiausią pinigų sumą ir kiek.

Pirmas žingsnis.

- Parašykite metodą, kuris suskaičiuotų ir išspausdintų, kiek išlaidų šeima turėjo kiekvieną savaitę:

```
//-------------------------------------------------------------
        /** Suskaičiuoja ir išspausdina, kiek šeima turėjo išlaidų kiekvieną savaitę.
        @param fv - rezultatų failo vardas
        @param A  - konteinerio vardas */
        //-------------------------------------------------------------
        static void IslaidosSavaitemis(string fv, ref Matrica A)
        {
            using (var fr = File.AppendText(fv))
            {
                for (int i = 0; i < A.n; i++)
                {
                    double suma = 0;
                    for (int j = 0; j < A.m; j++)
                    {
                        Asmuo x = A.ImtiReiksme(i, j);
                        suma = suma + x.ImtiPinigus();
                    }
                    fr.WriteLine("Savaitės nr. {0} išlaidos {1,5:c2}.", i + 1,
                                 (decimal)suma);                }
            }
        }
//-------------------------------------------------------------
```

- Papildykite programą kreipiniu į metodą:

```
IslaidosSavaitemis(CFr, ref seimosIslaidos);
```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

```
Savaitės nr. 1 išlaidos 186,50 €.
Savaitės nr. 2 išlaidos 131,70 €.
Savaitės nr. 3 išlaidos 191,40 €.
```

Antras žingsnis.

- Pirmo žingsnio skaičiavimo metode yra ir atsakymų spausdinimas. Užrašykite nurodyto metodo užklotą metodą, kai atsakymai surašomi į masyvą.

```
//---------------------------------------------------------
        /** Suskaičiuoja, kiek šeima turėjo išlaidų kiekvieną savaitę.
        @param A        - konteinerio vardas
        @param Islaidos - išlaidų masyvas*/
        //---------------------------------------------------------
        static void IslaidosSavaitemis(Matrica A, double [] Islaidos)
        {
                for (int i = 0; i < A.n; i++)
                {
                    double suma = 0;
                    for (int j = 0; j < A.m; j++)
                    {
                        Asmuo x = A.ImtiReiksme(i, j);
                        suma = suma + x.ImtiPinigus();
                    }
                    Islaidos[i] = suma;
            }
        }
```

- Užrašykite suformuoto masyvo spausdinimo metodą.

```
//---------------------------------------------------------
        /** Spausdina, kiek šeima turėjo išlaidų kiekvieną savaitę.
        @param CFr      - failo vardas
        @param n        - savaičių kiekis
        @param Islaidos - išlaidų masyvas*/
        //---------------------------------------------------------
        static void SpausdintiIslaidosSavaitemis1(string Cfr, double[] Islaidos, int n)
        {
            using (var fr = File.AppendText(CFr))
            {
                for (int i = 0; i < n; i++)
                {
                    fr.WriteLine("Savaitės nr. {0} išlaidos {1,5:c2}.", i + 1,
                                 (decimal)Islaidos[i]);
                }
            }
        }
```

- Papildykite programą nauju masyvu. Pataisykite programą kreipiniais į naujus metodus:

```
double[] Islaidos = new double[seimosIslaidos.n];
IslaidosSavaitemis(seimosIslaidos, Islaidos);
SpausdintiIslaidosSavaitemis1(CFr, Islaidos, seimosIslaidos.n);
```

- Išbandykite programą. Palyginkite gautus atsakymus su pirmame žingsnyje gautais.

Trečias žingsnis.

- Parašykite metodą, kuris suskaičiuotų, kiek išlaidų šeima turėjo nurodytą savaitės dieną:

```
//-------------------------------------------------------------
        /** Suskaičiuoja ir grąžina, kiek išlaidų šeima turėjo nurodytą savaitė dieną.
        @param A  - konteinerio vardas
        @param nr - savaitės dienos numeris  */
        //-------------------------------------------------------------
        static decimal IslaidosSavaitesDienaX(Matrica A, int nr)
        {
            double suma = 0;
            for (int i = 0; i < A.n; i++)
            {
                Asmuo x = A.ImtiReiksme(i, nr-1);
                suma = suma + x.ImtiPinigus();
            }
        return (decimal)suma;
        }
//-------------------------------------------------------------
```

- Papildykite programą kreipiniais į metodą (prieš spausdinimus į konsolę). Patikrinkite, kiek pinigų šeima išleido antradieniais ir šeštadieniais:

```
using (var fr = File.AppendText(CFr))
{
    fr.WriteLine();
    fr.WriteLine("Antradienių bendros išlaidos {0,5:c2}.",
                 IslaidosSavaitesDienaX(seimosIslaidos, 2));
    fr.WriteLine("Šeštadienių bendros išlaidos {0,5:c2}.",
                 IslaidosSavaitesDienaX(seimosIslaidos, 6));
}
```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

```
Antradienių bendros išlaidos 35,40 €.
Šeštadienių bendros išlaidos 50,40 €.
```

Ketvirtas žingsnis.

- Parašykite metodą, kuris suskaičiuotų, kada (kurią savaitę ir kurią savaitės dieną) buvo išleista didžiausia pinigų suma:

```
//-------------------------------------------------------------
        /** Suskaičiuoja, kurią savaitę ir kurią savaitės dieną
        // buvo išleista didžiausia pinigų suma.
        // A     - konteinerio vardas
        // eilNr - savaitės numeris
        // stNr  - savaitės dienos numeris  */
        //-------------------------------------------------------------
        static void DienaMaxIslaidos(Matrica A, out int eilNr, out int stNr)
        {
            eilNr = -1;
            stNr = -1;
            double max = 0;
            for (int i = 0; i < A.n; i++)
            {
                for (int j = 0; j < A.m; j++)
                {
                    double x = A.ImtiReiksme(i, j).ImtiPinigus();
                    if (x > max)
                    {
                        max = x;
                        eilNr = i + 1;
                        stNr = j + 1;
                    }
                }
            }
        }
//-------------------------------------------------------------
```

- Papildykite programą kintamaisiais ir skaičiavimais, kada ir kas išleido daugiausia (po pasirinktų dienų išlaidų spausdinimo):

```
fr.WriteLine();
int savaite, diena;
Asmuo a;
DienaMaxIslaidos(seimosIslaidos, out savaite, out diena);
fr.Write("Daugiausia išleista {0} sav. {1} dieną.", savaite, diena);
a = seimosIslaidos.ImtiReiksme(savaite - 1, diena - 1);
fr.WriteLine(" Pinigus išleido {0}: {1,5:c2}.", a.ImtiVarda(),
             a.ImtiPinigus());
```

- Išbandykite programą. Rezultatų faile, be ankstesnių rezultatų, matysite:

```
Daugiausia išleista 1 sav. 4 dieną. Pinigus išleido žmona: 100,20 €.
```

Programos patikrinimas.

Pakeiskite duomenų failo duomenis arba sukurkite kitą duomenų failą. Patikrinkite, kaip dirba programa, kai šeimos išlaidos skaičiuojamos vieną savaitę (`n = 1`) ir tą savaitę:

a) visai nebuvo išlaidų;
b) buvo tik žmonos išlaidos;
c) buvo tik vyro išlaidos.

Programos papildymas.

Pakeiskite metodą `IslaidosSavaitemis()` taip, kad šio metodo rezultatai būtų spausdinami rezultatų faile lentele, sudaryta iš dviejų skilčių: savaitės numeris ir išlaidos.

Savarankiško darbo užduotis.

Parašykite ir išbandykite metodą:

a) kuris suskaičiuotų, kurią savaitę išlaidos buvo mažiausios;
b) kuris suskaičiuotų ir išspausdintų rezultatų faile lentele šeimos išlaidas visomis savaitės dienomis: pirmadieniais, antradieniais ir t. t. Pasinaudokite metodu `IslaidosSavaitesDienaX()`.

5.5. Objektų masyvas ir dvimatis sveikųjų skaičių masyvas

- Klasės su dviem susietais masyvais aprašymas.
- Duomenų skaitymas ir spausdinimas.
- Veiksmai susietuose masyvuose.

Užduotis. Mokinių laikas, praleistas internete.
Duoti du tekstiniai failai. Pirmame tekstiniams faile yra mokyklos 5-12 klasių mokinių sąrašas. Pirmoje failo eilutėje užrašytas mokinių skaičius n. Tolesnėse failo eilutėse užrašyta informacija apie mokinius: pavardė, vardas, klasė, pažangumas (mokymosi vidurkis). Antro tekstinio failo pirmoje eilutėje užrašytas skaičius n ir dienų skaičius m. Žemiau pateikta informacija apie mokinių kiekvieną dieną praleistą laiką minutėmis internete: mokiniai (eilutės), dienos (stulpeliai).
Parašykite programą, kuri įvestų duomenis iš failų, išspausdintų faile, surikiuotų mokinius pagal klases ir praleistą laiką internete, suskaičiuotų, kiek vidutiniškai nurodytos klasės mokiniai praleido internete.

Pradiniai duomenys ir rezultatai.

Pradiniai duomenys
Pirmas duomenų failas
7 Jonaitis;Jonas; 6; 7,5; Aleksaitė;Alina; 6; 9,5; Petraitis;Petras; 5; 8,5; Antanaitis;Antanas; 6; 5,5; Juozaitis;Juozas; 5; 8,5; Rimaitis;Rimas; 6; 7,5; Rasaitė;Rasa; 5; 6,0;
Antras duomenų failas
7 5 120 100 90 60 50

100 200 150 200 10
80 90 80 90 80
120 80 60 140 70
60 60 60 60 60
0 0 0 0 0
0 50 60 120 40

Rezultatai

```
        Pradiniai duomenys

Mokinių kiekis 7
Dienų kiekis 5

      Mokyklos mokiniai (laikai = 0)

-----------------------------------------------
 Nr. Pavardė         Vardas     Klasė   Laikas
-----------------------------------------------
  1. Jonaitis        Jonas          6      0,00
  2. Aleksaitė       Alina          6      0,00
  3. Petraitis       Petras         5      0,00
  4. Antanaitis      Antanas        6      0,00
  5. Juozaitis       Juozas         5      0,00
  6. Rimaitis        Rimas          6      0,00
  7. Rasaitė         Rasa           5      0,00
-----------------------------------------------

Mokinių laikai, praleisti internete per 5 dienas.

   1.  120 100  90  60  50
   2.  100 200 150 200  10
   3.   80  90  80  90  80
   4.  120  80  60 140  70
   5.   60  60  60  60  60
   6.    0   0   0   0   0
   7.    0  50  60 120  40

       Rezultatai

      Mokyklos mokiniai (papildyta, laikai != 0)

-----------------------------------------------
 Nr. Pavardė         Vardas     Klasė   Laikas
-----------------------------------------------
  1. Jonaitis        Jonas          6    420,00
  2. Aleksaitė       Alina          6    660,00
  3. Petraitis       Petras         5    420,00
  4. Antanaitis      Antanas        6    470,00
  5. Juozaitis       Juozas         5    300,00
  6. Rimaitis        Rimas          6      0,00
  7. Rasaitė         Rasa           5    270,00
-----------------------------------------------

      Mokyklos mokiniai (surikiuoti)

-----------------------------------------------
 Nr. Pavardė         Vardas     Klasė   Laikas
-----------------------------------------------
  1. Rasaitė         Rasa           5    270,00
  2. Juozaitis       Juozas         5    300,00
  3. Petraitis       Petras         5    420,00
  4. Rimaitis        Rimas          6      0,00
  5. Jonaitis        Jonas          6    420,00
  6. Antanaitis      Antanas        6    470,00
  7. Aleksaitė       Alina          6    660,00
-----------------------------------------------
```

```
Mokinių laikai, praleisti internete (po rikiavimo) per 5 dienas.

   1.    0  50  60 120  40
   2.   60  60  60  60  60
   3.   80  90  80  90  80
   4.    0   0   0   0   0
   5.  120 100  90  60  50
   6.  120  80  60 140  70
   7.  100 200 150 200  10

5 klasės mokiniai internete vidutiniškai praleido330,00 minučių.
```

Programos kūrimo eiga.

- Paruošiami pradinių duomenų failai.
- Sukuriama klasė `Mokinys`, skirta vieno mokinio duomenims saugoti.
- Sukuriama klasė `Mokykla` klasės `Mokinys` objektams ir dvimačiam sveikųjų skaičių masyvui (mokinių laikams praleistiems internete) saugoti.
- Pagrindiniame metode `Main()` skelbiamas objektas, skirtas mokyklos mokinių duomenims saugoti.
- Sukuriami du metodai duomenims – mokiniams ir jų internete praleistiems laikams – iš failų skaityti.
- Sukuriami du metodai pradiniams duomenims – mokinių sąrašui ir jų internete praleistiems laikams – rezultatų faile spausdinti.
- Sukuriamas metodas klasės `Mokinys` masyvo objektams papildyti laikais, praleistais internete.
- Sukuriamas mokinių rikiavimo pagal klases ir laiką praleistą internete metodas.
- Sukuriamas metodas nurodytos klasės mokinių vidutinam praleistam laikui internete skaičiuoti.

Pirmas žingsnis.

- Sukurkite klasę mokinio duomenims saugoti:

```
//-------------------------------------------------------------
    /** Klasė mokinio duomenims saugoti
    @class Mokinys */
    class Mokinys
    {
        private string pav,     // mokinio pavardė
                       vard;    // mokinio vardas
        private int klas,       // klasė
                    laikas;     // laikas, praleistas internete
        private double vid;     // mokymosi vidurkis

        //-----------------------------------------------------
        /** Pradiniai mokinio duomenys */
        //-----------------------------------------------------
        public Mokinys()
        {
            pav = "";
            vard = "";
            klas = 0;
            laikas = 0;
            vid = 0.0;
        }

        //-----------------------------------------------------
        /** Mokinio duomenų įrašymas
        @param pav  - nauja pavardės reikšmė
        @param vard - nauja vardo reikšmė
        @param klas - nauja klasės reikšmė
        @param vid  - naujos vidurkio reikšmė */
        //-----------------------------------------------------
        public void Dėti(string pav, string vard, int klas, double vid)
        {
```

```
            this.pav = pav;
            this.vard = vard;
            this.klas = klas;
            this.vid = vid;
        }

        /** įrašo laiką */
        public void DėtiLaiką(int laik) { laikas = laik; }

        /** Grąžina mokinio pavardę */
        public string ImtiPav() { return pav; }

        /** Grąžina mokinio vardą */
        public string ImtiVard() { return vard; }

        /** Grąžina mokinio klasę */
        public int ImtiKlas() { return klas; }

        /** Grąžina mokinio vidurkį */
        public double ImtiVid() { return vid; }

        /** Grąžina mokinio laiką, praleistą internete */
        public int ImtiLaiką() { return laikas; }

        //---------------------------------------------------
        // Spausdinimo metodas
        //---------------------------------------------------
        public override string ToString()
        {
            string eilute;
            eilute = string.Format("{0, -15} {1, -10}   {2,2:d}    {3, 6:f2}",
                                   pav, vard, klas, laikas);
            return eilute;
        }

        //-----------------------------------------------------------
        /** Operatorius grąžina
        //  true, jeigu klasė yra mažesnė už kitą klasę, arba klasės yra lygios,
        //  o laikas yra mažesnis už kitą laiką;
        //  false - kitais atvejais. */
        //-----------------------------------------------------------
        public static bool operator <=(Mokinys pirmas, Mokinys antras)
        {
            return pirmas.klas < antras.klas ||
            pirmas.klas == antras.klas && pirmas.laikas < antras.laikas;
        }

        //-----------------------------------------------------------
        /** Operatorius grąžina
        //  true, jeigu klasė yra didesnė už kitą klasę, arba klasės yra lygios,
        //  o laikas yra didesnis už kitą laiką;
        //  false - kitais atvejais. */
        //-----------------------------------------------------------
        public static bool operator >=(Mokinys pirmas, Mokinys antras)
        {
            return pirmas.klas > antras.klas ||
            pirmas.klas == antras.klas && pirmas.laikas > antras.laikas;
        }
    }
//-------------------------------------------------------------
```

- Sukurkite klasę `Mokykla`, skirtą klasės `Mokinys` objektų masyvui ir dvimačiam sveikųjų skaičių masyvui – laikams saugoti:

```
//-------------------------------------------------------------
    /** Klasė mokinių duomenims saugoti
    @class Mokykla */
    class Mokykla
    {
        const int CMaxMk = 1000;      // didžiausias galimas mokinių skaičius
        const int CMaxDn = 30;        // didžiausias galimas dienų skaičius
        private Mokinys[] Mokiniai;   // mokinių duomenys
        public int n { get; set; }    // mokinių skaičius
        private int[,] WWW;           // laikas, praleistas internete
        public int m { get; set; }    // dienų skaičius

        public Mokykla()
        {
            n = 0;
            Mokiniai = new Mokinys[CMaxMk];

            m = 0;
            WWW = new int[CMaxMk, CMaxDn];
        }
        /** Grąžina nurodyto indekso mokinio objektą
        @param nr - mokinio indeksas */
        public Mokinys Imti(int nr) { return Mokiniai[nr]; }

        /** Padeda į mokinių objektų masyvą naują mokinį ir
        //  masyvo dydį padidina vienetu
        @param ob - mokinio objektas */
        public void Dėti(Mokinys ob) { Mokiniai[n++] = ob; }

        /** Pakeičia mokinių objektų masyvo mokinį,
        //  kurio numeris nr
        @param nr - keičiamo mokinio numeris
        @param mok - mokinių objekto masyvas */
        public void PakeistiMokinį(int nr, Mokinys mok) { Mokiniai[nr] = mok; }

        /** Pakeičia laikų matricos elementą
        @param i - mokinio numeris
        @param j - dienos numeris
        @param r - naujas laikas */
        public void DėtiWWW(int i, int j, int r) { WWW[i, j] = r; }

        /** Grąžina laikų matricos elementą
        @param i - mokinio numeris
        @param j - dienos numeris */
        public int ImtiWWW(int i, int j) { return WWW[i, j]; }

        //-------------------------------------------------------------
        /** Sukeičia dvi eilutes vietomis dvimačiame masyve WWW(n,m)
        @param nr1 – pirmos eilutės numeris
        @param nr2 – antros eilutės numeris */
        //-------------------------------------------------------------
        public void SukeistiEilutesWWW(int nr1, int nr2)
        {
            for (int j = 0; j < m; j++)
            {
                int d = WWW[nr1, j];
                WWW[nr1, j] = WWW[nr2, j];
                WWW[nr2, j] = d;
            }
        }
    }
//-------------------------------------------------------------
```

- Parašykite pagrindinį `Main()` metodą, kurioje būtų mokyklos mokinius aprašantis objektas.

```
//-------------------------------------------------------------
class Program
    {
        static void Main(string[] args)
        {
            Mokykla mokykl = new Mokykla();       // mokyklos mokinių duomenys

            Console.WriteLine("Programa baigė darbą!");
        }
    }
//-------------------------------------------------------------
```

- Patikrinkite, kaip dirba programa. Ekrane turite matyti:

```
Programa baigė darbą!
```

Antras žingsnis.

- Parašykite pirmąjį metodą, kuris užpildytų objektų masyvą `Mokiniai(n)` duomenimis iš pirmo duomenų failo:

```
//-------------------------------------------------------------
        /** Failo duomenis surašo į konteinerį.
        @param fd     - duomenų failo vardas
        @param mokykl - konteineris */
        //---------------------------------------------------------
        static void SkaitytiMok(string fd, ref Mokykla mokykl)
        {
            string pav, vard;
            int klas, nn;
            double vid;
            string line;
            using (StreamReader reader = new StreamReader(fd))
            {
                line = reader.ReadLine();
                string[] parts;
                nn = int.Parse(line);
                for (int i = 0; i < nn; i++)
                {
                    line = reader.ReadLine();
                    parts = line.Split(';');
                    pav = parts[0];
                    vard = parts[1];
                    klas = int.Parse(parts[2]);
                    vid = double.Parse(parts[3]);
                    Mokinys mok;
                    mok = new Mokinys();
                    mok.Dėti(pav, vard, klas, vid);
                    mokykl.Dėti(mok);
                }
            }
        }
//-------------------------------------------------------------
```

- Parašykite antrąjį metodą, kuris užpildytų dvimatį masyvą `WWW(n, m)` duomenimis iš antro duomenų failo:

```
//-------------------------------------------------------------
        /** Įveda duomenis į dvimatį skaičių masyvą WWW(n,m)
        @param fd – duomenų failo vardas
        @param mokykl – objekto, kuriame yra dvimatis laikų masyvas, vardas */
        //---------------------------------------------------------
        static void SkaitytiLaik(string fd, ref Mokykla mokykl)
        {
            int laikas, nn, mm;
            string line;
            using (StreamReader reader = new StreamReader(fd))
            {
                line = reader.ReadLine();
                string[] parts;
                nn = int.Parse(line);
```

```
                line = reader.ReadLine();
                mm = int.Parse(line);
                mokykl.m = mm;
                for (int i = 0; i < mokykl.n; i++)
                {
                    line = reader.ReadLine();
                    parts = line.Split(' ');
                    for (int j = 0; j < mokykl.m; j++)
                    {
                        laikas = int.Parse(parts[j]);
                        mokykl.DėtiWWW(i, j, laikas);
                    }
                }
            }
        }
//---------------------------------------------------------------
```

- Parašykite metodą, kuris išspausdintų objektų masyvo `Mokiniai(n)` duomenis faile:

```
//---------------------------------------------------------------
        /** Spausdina konteinerio duomenis faile.
        @param fv        - rezultatų failo vardas
        @param mokykl    - mokinio duomenų konteineris
        @param antraste  - užrašas virš lentelės */
        //-----------------------------------------------------------
        static void Spausdinti(string fv, Mokykla mokykl, string antraštė)
        {
            using (var fr = File.AppendText(fv))
            {
                string bruksnys = new string('-', 46);
                fr.WriteLine(antraštė);
                fr.WriteLine();

                fr.WriteLine(bruksnys);
                fr.WriteLine(" Nr. Pavardė         Vardas     Klasė   Laikas  ");
                fr.WriteLine(bruksnys);
                for (int i = 0; i < mokykl.n; i++)
                    fr.WriteLine("  {0}. {1}   ", i + 1, mokykl.Imti(i).ToString());
                fr.WriteLine(bruksnys);
                fr.WriteLine();
            }
        }
//---------------------------------------------------------------
```

- Parašykite metodą, kuris išspausdintų dvimačio masyvo `WWW(n, m)` duomenis faile:

```
//---------------------------------------------------------------
        /** Spausdina internete praleistų laikų matricą faile.
        @param fv     - rezultatų failo vardas
        @param mokykl - mokinio duomenų konteineris
        @param koment - užrašas virš matricos */
        //-----------------------------------------------------------
        static void SpausdintiLaik(string fv, Mokykla mokykl, string koment)
        {
            using (var fr = File.AppendText(fv))
            {
                fr.WriteLine("{0} per {1} dienas.", koment, mokykl.m);
                fr.WriteLine();
                for (int i = 0; i < mokykl.n; i++)
                {
                    fr.Write("{0,4:d}.  ", i + 1);
                    for (int j = 0; j < mokykl.m; j++)
                        fr.Write("{0,3:d} ", mokykl.ImtiWWW(i, j));
                    fr.WriteLine();
                }
            }
        }
//---------------------------------------------------------------
```

- Papildykite programą duomenų įvedimo iš failų `Duomenys.txt`, `Duomenys1.txt` ir spausdinimo rezultatų faile `Atsakymai.txt` veiksmais – užrašykite kreipinius į sukurtus metodus:

```
//---------------------------------------------------------------
class Program
    {
        const string CFd = "..\\..\\Duomenys.txt";
        const string CFd1 = "..\\..\\Duomenys1.txt";
        const string CFr = "..\\..\\Atsakymai.txt";

        static void Main(string[] args)
        {
            Mokykla mokykl = new Mokykla();       // mokyklos mokinių duomenys
            SkaitytiMok(CFd, ref mokykl);
            SkaitytiLaik(CFd1, ref mokykl);
            using (var fr = File.CreateText(CFr))
            {
                fr.WriteLine("       Pradiniai duomenys");
                fr.WriteLine();
                fr.WriteLine("Mokinių kiekis {0}", mokykl.n);
                fr.WriteLine("Dienų kiekis {0}", mokykl.m);
                fr.WriteLine();
            }
            Spausdinti(CFr, mokykl, "      Mokyklos mokiniai (laikai = 0)");
            SpausdintiLaik(CFr, mokykl, "Mokinių laikai, praleisti internete");

            Console.WriteLine("Pradiniai duomenys išspausdinti faile: {0}", CFr);
            Console.WriteLine("Programa baigė darbą!");
        }
//---------------------------------------------------------------
```

- Išbandykite, kaip veikia programa. Ekrane turėtumėte matyti:

```
Pradiniai duomenys išspausdinti faile: ..\..\Atsakymai.txt
Programa baigė darbą!
```

- Rezultatų faile `Atsakymai.txt` bus išspausdintas mokinių sąrašas ir dvimačiame masyve esantys duomenys (žiūr. rezultatus šio skyrelio pradžioje).

Trečias žingsnis.

- Parašykite klasės `Mokykla` metodą, masyvo `Mokiniai(n)` objektams papildyti laikais, praleistais internete iš masyvo `WWW(n,m)`:

```
//---------------------------------------------------------------
        /** Objektų masyvo papildymas laikais, praleistais internete,
        // iš dvimačio masyvo  */
        //---------------------------------------------------------
        public void PapildytiMokiniųDuomenis()
        {
            int suma;
            Mokinys mok;
            for (int i = 0; i < n; i++)
            {
                suma = 0;
                for (int j = 0; j < m; j++)
                    suma = suma + WWW[i, j];
                mok = Imti(i);
                mok.DėtiLaiką(suma);
                PakeistiMokinį(i, mok);
            }
        }
//---------------------------------------------------------------
```

- Išbandykite aukščiau parašytą metodą – parašykite pagrindiniame metode `Main()` kreipinį į šį metodą ir kreipinį į pirmąjį spausdinimo metodą (prieš spausdinimus į konsolę):

```
using (var fr = File.AppendText(CFr))
{
    fr.WriteLine();
    fr.WriteLine("       Rezultatai");
```

```
            fr.WriteLine();
        }
        mokykl.PapildytiMokiniųDuomenis();
        Spausdinti(CFr, mokykl, "      Mokyklos mokiniai (papildyta, laikai != 0)");
```

- Rezultatų failas `Atsakymai.txt` bus papildytas dar viena lentele (žiūr. rezultatus šio skyrelio pradžioje).

Ketvirtas žingsnis.

- Parašykite klasės `Mokykla` metodą masyvui `Mokiniai(n)` ir dvimačiam masyvui `WWW(n,m)` rikiuoti pagal klases ir laiką praleistą internete:

```
//-------------------------------------------------------------
        /** Surikiuoja objektų masyvą pagal klases ir laikus
        // praleistus internete
        // Pastaba: kartu atliekami pakeitimai ir dvimačiame skaičių masyve WWW(n,m)  */
        //-------------------------------------------------------------
        public void RikiuotiMinMax()
        {
            Mokinys mok;
            for (int i = 0; i < n-1; i++)
            {
                int minnr = i;
                for (int j = i+1; j < n; j++)
                    if (Imti(j) <= Imti(minnr))
                        minnr = j;
                mok = Imti(i);
                // pakeitimai masyvuose Mokiniai ir WWW
                PakeistiMokinį(i, Imti(minnr));
                PakeistiMokinį(minnr, mok);
                SukeistiEilutesWWW(i, minnr);
            }
        }
//-------------------------------------------------------------
```

- Išbandykite aukščiau parašytą metodą – parašykite pagrindiniame metode `Main()` kreipinį į šį metodą ir kreipinį į abu spausdinimo metodus (prieš spausdinimus į konsolę):

```
mokykl.RikiuotiMinMax();
Spausdinti(CFr, mokykl, "      Mokyklos mokiniai (surikiuoti)");
SpausdintiLaik(CFr, mokykl,
                   "Mokinių laikai, praleisti internete (po rikiavimo)");
```

- Rezultatų failas `Atsakymai.txt` bus papildytas dviems lentelėmis (žiūr. rezultatus šio skyrelio pradžioje).

Penktas žingsnis.

- Parašykite metodą nurodytos klasės mokinių vidutinam praleistam laikui internete skaičiuoti:

```
//-------------------------------------------------------------
        /** Suskaičiuoja ir grąžina nurodytos klasės mokinių vidutinį laiką,
        // praleistą internete
        @param mokykl – objekto vardas
        @param klasė  – klasės numeris  */
        //-------------------------------------------------------------
        static double VidLaikasKl(Mokykla mokykl, int klasė)
        {
            double suma = 0;
            int kiek = 0;
            for (int i = 0; i < mokykl.n; i++)
            if (mokykl.Imti(i).ImtiKlas() == klasė)
            {
                kiek++;
                suma = suma + mokykl.Imti(i).ImtiLaiką();
            }
           if (kiek !=0)
               return suma / kiek;
           else
               return 0;
        }
```

```
//---------------------------------------------------------
```

- Išbandykite aukščiau parašytą metodą – parašykite pagrindiniame metode `Main()` kreipinį į šį metodą:

```
using (var fr = File.AppendText(CFr))
{
    int klasė;
    Console.WriteLine("Užrašykite klasę (1-12): ");
    klasė = int.Parse(Console.ReadLine());
    fr.WriteLine();
    if (VidLaikasKl(mokykl, klasė) != 0)
        fr.WriteLine("{0} klasės mokiniai internete vidutiniškai praleido "
                     + "{1,6:f2} minučių.", klasė, VidLaikasKl(mokykl, klasė));
    else
        fr.WriteLine("{0} klasės mokinių sąraše nėra.", klasė);
}
```

- Paleidus programą ir klaviatūra įvedus 5 (penkta klasė) rezultatų failas `Atsakymai.txt` bus papildytas viena eilute (žiūr. rezultatus šio skyrelio pradžioje).

Programos patikrinimas.

Pakeiskite duomenų failų duomenis arba susikurkite kitus duomenų failus. Patikrinkite, kaip dirba programa, kai dienų skaičius `m = 1`.

Programos papildymas.

Pakeiskite metodą `SpausdintiLaik()` taip, kad būtų sunumeruoti ir mokinių laikai praleisti internete, t. y. virš stulpelių būtų užrašyti dienų numeriai.

Savarankiško darbo užduotis.

Parašykite ir išbandykite metodą, kuris suskaičiuotų, kiek mokykloje yra mokinių, kurie nesinaudoja internetu.

5.6. Objektų konteineris ir dvimatis sveikųjų skaičių konteineris

Skyriuje 5.5 išspręstą užduotį išspręsime dar kartą. Panaudosime ne dvi, o tris klases. Aprašysime dvi konteinerines klases: mokinio duomenų ir dvimatį sveikųjų skaičių Todėl veiksmai bus atliekami susietuose konteineriuose. Taip pat naudosime gilų kopijavimą, kad negrąžintume objektų nuorodų.

```
using System;
using System.IO;
using System.Collections;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
//---------------------------------------------------------
// Trys klasės. Du duomenų failai.
// Duomenų įvedimas iš failų. Atsakymų spausdinimas į failą.
// Apskaičiuoja visą laiką, praleistą internete.
// Rikiuoja duomenis. Randa norimos klasės laikų vidurkį
//---------------------------------------------------------
namespace mokiniai
{   /** Klasė mokinio duomenims saugoti
    @class Mokinys */
    class Mokinys
    {
        private string pav,     // mokinio pavardė
                       vard;    // mokinio vardas
        private int klas,       // klasė
                    laikas;     // laikas, praleistas internete
        private double vid;     // mokymosi vidurkis

        //-----------------------------------------------
        /** Pradiniai mokinio duomenys */
        //-----------------------------------------------
```

```
        public Mokinys()
        {
            pav = "";
            vard = "";
            klas = 0;
            laikas = 0;
            vid = 0.0;
        }

        public Mokinys(string pav, string vard, int klas, int laikas, double vid)
        {
            this.pav = pav;
            this.vard = vard;
            this.klas = klas;
            this.laikas = laikas;
            this.vid = vid;
        }

        //-----------------------------------------------------
        /** Mokinio duomenų įrašymas
        @param pav  - nauja pavardės reikšmė
        @param vard - nauja vardo reikšmė
        @param klas - nauja klasės reikšmė
        @param vid  - naujo vidurkio reikšmė */
        //-----------------------------------------------------
        public void Dėti(string pav, string vard, int klas, double vid)
        {
            this.pav = pav;
            this.vard = vard;
            this.klas = klas;
            this.vid = vid;
        }

        /** įrašo laiką */
        public void DėtiLaiką(int laik) { laikas = laik; }

        /** Grąžina mokinio pavardę */
        public string ImtiPav() { return pav; }

        /** Grąžina mokinio vardą */
        public string ImtiVard() { return vard; }

        /** Grąžina mokinio klasę */
        public int ImtiKlas() { return klas; }

        /** Grąžina mokinio vidurkį */
        public double ImtiVid() { return vid; }

        /** Grąžina mokinio laiką, praleistą internete */
        public int ImtiLaiką() { return laikas; }

        //-----------------------------------------------------
        // Spausdinimo metodas
        //-----------------------------------------------------
        public override string ToString()
        {
            string eilute;
            eilute = string.Format("{0, -15} {1, -10}   {2,2:d}    {3, 6:f2}",
                                   pav, vard, klas, laikas);
            return eilute;
        }

        //-----------------------------------------------------------
        /** Operatorius grąžina
        //  true, jeigu klasė yra mažesnė už kitą klasę, arba klasės yra lygios,
        //  o laikas yra mažesnis už kitą laiką;
        //  false - kitais atvejais. */
        //-----------------------------------------------------------
```

```
        public static bool operator <=(Mokinys pirmas, Mokinys antras)
        {
            return pirmas.klas < antras.klas ||
            pirmas.klas == antras.klas && pirmas.laikas < antras.laikas;
        }

        //-------------------------------------------------------------
        /** Operatorius grąžina
        //  true, jeigu klasė yra didesnė už kitą klasę, arba klasės yra lygios,
        //  o laikas yra didesnis už kitą laiką;
        //  false - kitais atvejais. */
        //-------------------------------------------------------------
        public static bool operator >=(Mokinys pirmas, Mokinys antras)
        {
            return pirmas.klas > antras.klas ||
            pirmas.klas == antras.klas && pirmas.laikas > antras.laikas;
        }
    }

    /** Klasė mokinių duomenims saugoti
    @class Mokykla */
    class Mokykla
    {
        const int CMaxMk = 1000;      // didžiausias galimas mokinių skaičius
        public int n { get; set; }    // mokinių skaičius
        private Mokinys[] Mokiniai;   // mokinių duomenys

        public Mokykla()
        {
            n = 0;
            Mokiniai = new Mokinys[CMaxMk];
        }

        /** Grąžina nurodyto indekso mokinio objekto kopiją
        //  Gilus kopijavimas
        @param nr - mokinio indeksas */
        public Mokinys Imti(int nr)
        {
            Mokinys ob1 = new Mokinys(Mokiniai[nr].ImtiPav(), Mokiniai[nr].ImtiVard(),
                                      Mokiniai[nr].ImtiKlas(),okiniai[nr].ImtiLaiką(),
                                      Mokiniai[nr].ImtiVid());
            return ob1; }

        /** Padeda į mokinių objektų masyvą naują mokinį ir
        //  masyvo dydį padidina vienetu
        //  gilus kopijavimas
        @param ob - mokinio objektas */
        public void Dėti(Mokinys ob)
        {
            Mokinys ob1 =new Mokinys(ob.ImtiPav(),ob.ImtiVard(),ob.ImtiKlas(),
                                     ob.ImtiLaiką(),ob.ImtiVid());
            Mokiniai[n++] = ob1; }

        /** Pakeičia mokinių objektų masyvo mokinį,
        //  kurio numeris nr
        //  gilus kopijavimas
        @param nr - keičiamo mokinio numeris
        @param mok - mokinių objekto masyvas */
        public void PakeistiMokinį(int nr, Mokinys mok)
        {
            Mokinys ob1 = new Mokinys(mok.ImtiPav(), mok.ImtiVard(), mok.ImtiKlas(),
                                      mok.ImtiLaiką(), mok.ImtiVid());
            Mokiniai[nr] = ob1;
        }
    }

    /** Klasė saugoti laikams, praleistiems internete
    @class Matrica */
```

```
class Matrica
{
    const int CMaxMk = 1000;      // didžiausias galimas mokinių skaičius
    const int CMaxDn = 30;        // didžiausias galimas dienų skaičius
    public int n { get; set; }    // mokinių skaičius
    public int m { get; set; }    // dienų skaičius
    private int[,] WWW;           // laikas, praleistas internete

    public Matrica()
    {
        n = 0;
        m = 0;
        WWW = new int[CMaxMk, CMaxDn];
    }

    /** Pakeičia laikų matricos elementą
    @param i - mokinio numeris
    @param j - dienos numeris
    @param r - naujas laikas */
    public void DėtiWWW(int i, int j, int r) { WWW[i, j] = r; }

    /** Grąžina laikų matricos elementą
    @param i - mokinio numeris
    @param j - dienos numeris */
    public int ImtiWWW(int i, int j) { return WWW[i, j]; }

    //-------------------------------------------------------------
    /** Sukeičia dvi eilutes vietomis dvimačiame masyve WWW(n,m)
    @param nr1 - pirmos eilutės numeris
    @param nr2 - antros eilutės numeris */
    //-------------------------------------------------------------
    public void SukeistiEilutesWWW(int nr1, int nr2)
    {
        for (int j = 0; j < m; j++)
        {
            int d = WWW[nr1, j];
            WWW[nr1, j] = WWW[nr2, j];
            WWW[nr2, j] = d;
        }
    }
}

class Program
{
    const string CFd = "..\\..\\Duomenys.txt";
    const string CFd1 = "..\\..\\Duomenys1.txt";
    const string CFr = "..\\..\\Atsakymai.txt";

    static void Main(string[] args)
    {
        Mokykla mokykl = new Mokykla();       // mokyklos mokinių duomenys
        Matrica mokykla = new Matrica();
        SkaitytiMok(CFd, ref mokykl);
        SkaitytiLaik(CFd1, ref mokykla);
        if (File.Exists(CFr))
            File.Delete(CFr);
        using (var fr = File.CreateText(CFr))
        {
            fr.WriteLine("       Pradiniai duomenys");
            fr.WriteLine();
            fr.WriteLine("Mokinių kiekis {0}", mokykl.n);
            fr.WriteLine("Dienų kiekis {0}", mokykla.m);
            fr.WriteLine();
        }
        Spausdinti(CFr, mokykl, "      Mokyklos mokiniai (laikai = 0)");
        SpausdintiLaik(CFr, mokykla, "Mokinių laikai, praleisti internete");

        using (var fr = File.AppendText(CFr))
```

```
        {
            fr.WriteLine();
            fr.WriteLine("       Rezultatai");
            fr.WriteLine();
        }
        PapildytiMokiniųDuomenis(mokykl, mokykla);
        Spausdinti(CFr, mokykl, "      Mokyklos mokiniai (papildyta, laikai != "
                         0)");
        RikiuotiMinMax(mokykl, mokykla);
        Spausdinti(CFr, mokykl, "      Mokyklos mokiniai (surikiuoti)");
        SpausdintiLaik(CFr, mokykla,
                          "Mokinių laikai, praleisti internete (po rikiavimo)");
        using (var fr = File.AppendText(CFr))
        {
            int klasė;
            Console.WriteLine("Užrašykite klasę (1-12): ");
            klasė = int.Parse(Console.ReadLine());
            fr.WriteLine();
            if (VidLaikasKl(mokykl, klasė) != 0)
               fr.WriteLine("{0} klasės mokiniai internete vidutiniškai praleido "
                            + "{1,6:f2} minučių.", klasė, VidLaikasKl(mokykl,
                            klasė));
            else
                fr.WriteLine("{0} klasės mokinių sąraše nėra.", klasė);
        }
        Console.WriteLine("Pradiniai duomenys išspausdinti faile: {0}", CFr);
        Console.WriteLine("Programa baigė darbą!");
    }

    //---------------------------------------------------------------
    /** Failo duomenis surašo į konteinerį.
    @param fd     - duomenų failo vardas
    @param mokykl - konteineris */
    //---------------------------------------------------------------
    static void SkaitytiMok(string fd, ref Mokykla mokykl)
    {
        string pav, vard;
        int klas, nn;
        double vid;
        string line;
        using (StreamReader reader = new StreamReader(fd))
        {
            line = reader.ReadLine();
            string[] parts = line.Split(';');
            nn = int.Parse(parts[0]);
            for (int i = 0; i < nn; i++)
            {
                line = reader.ReadLine();
                parts = line.Split(';');
                pav = parts[0];
                vard = parts[1];
                klas = int.Parse(parts[2]);
                vid = double.Parse(parts[3]);
                Mokinys mok;
                mok = new Mokinys();
                mok.Dėti(pav, vard, klas, vid);
                mokykl.Dėti(mok);
            }
        }
    }

    //---------------------------------------------------------------
    /** Įveda duomenis į dvimatį skaičių masyvą WWW(n,m)
    @param fd - duomenų failo vardas
    @param mokykla - matricos konteineris */
    //---------------------------------------------------------------
```

```
static void SkaitytiLaik(string fd, ref Matrica mokykla)
{
    int laikas, nn, mm;
    string line;
    using (StreamReader reader = new StreamReader(fd))
    {
        line = reader.ReadLine();
        string[] parts = line.Split(';');
        nn = int.Parse(parts[0]);
        mm = int.Parse(parts[1]);
        mokykla.n = nn;
        mokykla.m = mm;
        for (int i = 0; i < mokykla.n; i++)
        {
            line = reader.ReadLine();
            parts = line.Split(' ');
            for (int j = 0; j < mokykla.m; j++)
            {
                laikas = int.Parse(parts[j]);
                mokykla.DėtiWWW(i, j, laikas);
            }
        }
    }
}

//------------------------------------------------------------
/** Spausdina konteinerio duomenis faile.
@param fv        - rezultatų failo vardas
@param mokykl    - mokinio duomenų konteineris
@param antraste  - užrašas virš lentelės */
//------------------------------------------------------------
static void Spausdinti(string fv, Mokykla mokykl, string antraštė)
{
    using (var fr = File.AppendText(fv))
    {
        string bruksnys = new string('-', 46);
        fr.WriteLine(antraštė);
        fr.WriteLine();
        fr.WriteLine(bruksnys);
        fr.WriteLine(" Nr. Pavardė         Vardas     Klasė   Laikas  ");
        fr.WriteLine(bruksnys);
        for (int i = 0; i < mokykl.n; i++)
            fr.WriteLine("  {0}. {1}   ", i + 1, mokykl.Imti(i).ToString());
        fr.WriteLine(bruksnys);
        fr.WriteLine();
    }
}
//------------------------------------------------------------
/** Spausdina internete praleistų laikų matricą faile.
@param fv     - rezultatų failo vardas
@param mokykla - matricos duomenų konteineris
@param koment - užrašas virš matricos */
//------------------------------------------------------------
static void SpausdintiLaik(string fv, Matrica mokykla, string koment)
{
    using (var fr = File.AppendText(fv))
    {
        fr.WriteLine("{0} per {1} dienas.", koment, mokykla.m);
        fr.WriteLine();
        for (int i = 0; i < mokykla.n; i++)
        {
            fr.Write("{0,4:d}.  ", i + 1);
            for (int j = 0; j < mokykla.m; j++)
                fr.Write("{0,3:d} ", mokykla.ImtiWWW(i, j));
            fr.WriteLine();
        }
    }
}
```

```
        //-------------------------------------------------------------
        /** Suskaičiuoja ir grąžina nurodytos klasės mokinių vidutinį laiką,
        // praleistą internete
        @param mokykl - konteinerio vardas
        @param klasė  - klasės numeris  */
        //-------------------------------------------------------------
        static double VidLaikasKl(Mokykla mokykl, int klasė)
        {
            double suma = 0;
            int kiek = 0;
            for (int i = 0; i < mokykl.n; i++)
            if (mokykl.Imti(i).ImtiKlas() == klasė)
            {
                kiek++;
                suma += mokykl.Imti(i).ImtiLaiką();
            }
           if (kiek !=0)
               return suma / kiek;
           else
               return 0;
        }

        //-------------------------------------------------------------
        /** Objektų masyvo papildymas laikais, praleistais internete,
        // iš dvimačio masyvo  */
        //-------------------------------------------------------------
        static void PapildytiMokiniųDuomenis(Mokykla mokykl,Matrica mokykla)
        {
            int suma;
            Mokinys mok;
            for (int i = 0; i < mokykl.n; i++)
            {
                suma = 0;
                for (int j = 0; j < mokykla.m; j++)
                    suma = suma + mokykla.ImtiWWW(i, j);
                mok = mokykl.Imti(i);
                mok.DėtiLaiką(suma);
                mokykl.PakeistiMokinį(i, mok);
            }
        }

        //-------------------------------------------------------------
        /** Surikiuoja objektų masyvą pagal klases ir laikus
        // praleistus internete
        // Pastaba: kartu atliekami pakeitimai ir dvimačiame skaičių masyve
        // mokykla(n,m)  */
        //-------------------------------------------------------------
        static void RikiuotiMinMax(Mokykla mokykl, Matrica mokykla)
        {
            Mokinys mok;
            for (int i = 0; i < mokykl.n - 1; i++)
            {
                int minnr = i;
                for (int j = i + 1; j < mokykl.n; j++)
                    if (mokykl.Imti(j) <= mokykl.Imti(minnr))
                        minnr = j;
                mok = mokykl.Imti(i);
                // pakeitimai konteineriuose mokykl ir mokykla
                mokykl.PakeistiMokinį(i, mokykl.Imti(minnr));
                mokykl.PakeistiMokinį(minnr, mok);
                mokykla.SukeistiEilutesWWW(i, minnr);
            }
        }
    }
}
```

5.7. Kontroliniai klausimai

1. Kokie yra pagrindiniai vienmačio ir dvimačio konteinerio skirtumai?
2. Kaip kompiuterio atmintyje yra išdėstomi dvimačio konteinerio duomenys?
3. Aprašykite dvimatį sveikųjų skaičių masyvą, kuriame būtų viena eilutė ir CMax stulpelių.
4. Aprašykite dvimatį realiųjų skaičių masyvą, kuriame būtų CMax eilučių ir vienas stulpelis.
5. Užrašykite sakinius, kurie dvimačiame sveikųjų masyve A(n, m) sukeistų vietomis pirmo ir paskutinio stulpelio elemento reikšmes.
6. Duota konteinerinė klasė A, kurioje yra dvimatis sveikųjų skaičių masyvas A(n, m). Užrašykite klasės A metodą Didinti(), kuris masyve A nurodytos eilutės pirmojo elemento reikšmę padidintų 1 (vienetu).
7. Duota konteinerinė klasė A, kurioje yra dvimatis sveikųjų skaičių masyvas A(n, m). Užrašykite klasės A metodą Mažinti(), kuris masyve A nurodyto stulpelio paskutiniojo elemento reikšmę sumažintų 1 (vienetu).
8. Duota konteinerinė klasė A, kurioje yra dvimatis sveikųjų skaičių masyvas A(n, m). Užrašykite klasės A metodą ŠalintiEilutę(), kuris iš masyvo A pašalintų nurodytą eilutę.
9. Duota konteinerinė klasė A, kurioje yra dvimatis sveikųjų skaičių masyvas A(n, m). Užrašykite klasės A metodą ŠalintiStulpelį(), kuris iš masyvo A pašalintų nurodytą stulpelį.
10. Duotas metodas:

```
// Metodas suskaičiuoja ir grąžina masyvo A(n) reikšmių sumą
static int Suma(int [] A, int n);
```

Kuris iš pateiktų kreipinių į metodą grąžina dvimačio sveikųjų skaičių masyvo A(n, m) antros eilutės reikšmių sumą?

a) `Suma(A[1], m);`
b) `Suma(A[1], m-1);`
c) `Suma(A[2], n);`
d) `Suma(A[2], n-1).`

11. Duotas metodas:

```
static int Funkcija(int [] A, int n)
{
  int s = 0;
  for (int j = 0; j < n; j++)
    if (A[j] > A[n-1]) s = s + A[j];
  return s;
}
```

Ką matysite ekrane, atlikus tokius veiksmus:

```
int [][] B =
          { new int[] { 1, 3, 5, 2  },
            new int[] { 0, 2, -4, 2 },
            new int[] { 1, 4, 5, 0  } };
            int n = 3, m = 4;
            Console.WriteLine(" {0} {1}",Funkcija(B[1], m) ,Funkcija(B[2], m));
```

a) `0  10`
b) `8  0`
c) `10  8`
d) `0  8`

12. Duotas programos fragmentas:

```
int[,] A;
A = new int[10, 10];
int n = 5;
...
int s = 0;
for (int i = 0; i < n; i++)
  s = s + A[i][i];
...
```

Kurios matricos dalies elementų reikšmių suma skaičiuojama?

a) pagrindinės įstrižainės;
b) šalutinės įstrižainės;
c) i-osios eilutės;
d) virš pagrindinės įstrižainės;

5.8. Užduotys

Reikalavimas visoms užduotims: Negalima grąžinti nuorodų iš klasių. Naudokite gilų kopijavimą.

U5–1. Pelnyti taškai

Pirmo failo pirmoje failo eilutėje nurodytas krepšinio komandos pavadinimas, miestas, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: pavardė, vardas, gimimo metai, ūgis, žaidimo pozicija. Kitame faile atitinkamai krepšininkų (stulpeliai) išdėstymo tvarka pateikta kiekvienose rungtynėse (eilutės) įmestų taškų skaičius. Suraskite rezultatyviausią komandos krepšininką. Suraskite kiekvienos pozicijos rezultatyviausią krepšininką. Atleiskite iš komandos du mažiausiai taškų pelniusius krepšininkus (pašalinkite jų rezultatus ir jų duomenis).

U5–2. Futbolas

Pirmo failo pirmoje failo eilutėje nurodytas futbolo komandų skaičius. Tolesnėse eilutėse pateikta informacija apie futbolo komandas: pavadinimas, miestas, trenerio pavardė, vardas. Kitame faile pateikta I rato rezultatų lentelė, išreikšta pelnytais įvarčiais. Suskaičiuokite kiekvienos komandos surinktų taškų skaičių, jei už pergalę skiriami 3 taškai, o už lygiąsias – 1 taškas. Sudarykite komandų turnyrinę lentelę – surikiuokite surinktų taškų mažėjimo tvarka. Jei komandos surinko taškų vienodai, aukščiau ta komanda, kuri turi daugiau pergalių. Suraskite daugiausiai įvarčių pelniusią komandą. Suraskite komandas, kurios daugiausiai rungtynių nepraleido įvarčių.

U5–3. Studentai

Pirmo failo pirmoje failo eilutėje nurodytas studentų kiekis ir mėnesio dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, fakultetas, specialybė. Kitame faile pateikta informacija apie studentų paskaitų lankomumą: studentai (stulpeliai), praleista paskaitų per dieną (eilutės). Nustatykite, ar yra dienų, kai paskaitose dalyvavo visi studentai, jei taip, kurios tai dienos. Surikiuokite studentus pagal fakultetus. Suskaičiuokite, kiek kiekvieno fakulteto studentai praleido paskaitų. Nustatykite, ar yra studentų, kuriuos reikia šalinti (nelankė paskaitų daugiau nei pusę mėnesio dienų). Pašalinkite tokius studentus.

U5–4. Prenumerata

Paštas atlieka leidinių prenumeratą. Pirmo failo pirmoje failo eilutėje nurodytas leidinių kiekis ir mėnesio dienų skaičius. Tolesnėse eilutėse pateikta informacija apie leidinius: pavadinimas, mėnesio prenumeratos kaina, banko pavadinimas, sąskaitos numeris, procentai, atiduodami prenumeratos rinkėjui. Kitame faile pateikta informacija apie kiekvieno leidinio prenumeratos eigą: leidiniai (stulpeliai), kiek jų užsakyta (eilutės). Suskaičiuokite, kiek per mėnesį kurio leidinio užsakyta. Nustatykite, kuris leidinys yra lyderis. Kiekvienam bankui sudarykite pavedimų sąrašą, kad pervesti leidiniui prenumeratos pinigus. Nustatykite, ar buvo leidinių, kurie nebuvo užsakomi daugiau nei vieną dieną.

U5–5. Darbininkai

Pirmo failo pirmoje failo eilutėje nurodytas darbininkų skaičius, mėnesio dirbtų dienų skaičius, vienos detalės įkainis. Tolesnėse eilutėse pateikta informacija apie darbininkus: pavardė, vardas, banko pavadinimas ir sąskaitos numeris. Kitame faile pateikta, kiek kiekvieną dieną (eilutės) darbininkas (stulpeliai) pagamino detalių. Suskaičiuokite kiekvienam darbininkui jo atlyginimą. Nustatykite, kuriam darbininkui blogiausiai sekėsi dirbti. Nustatykite, kurią mėnesio dieną buvo pagaminta daugiausiai detalių. Kiekvienam bankui atskirai sudarykite pavedimų sąrašą, kur nurodysite darbininko pavardę, vardą, sąskaitos numerį ir pervedamą sumą.

U5–6. Žaistos minutės

Pirmo failo pirmoje failo eilutėje nurodytas komandos pavadinimas ir miestas, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: pavardė, vardas, gimimo metai, ūgis, žaidimo pozicija. Kitame faile atitinkamai krepšininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) žaistų minučių skaičius. Suraskite daugiausiai laiko žaidusį komandos krepšininką. Suraskite kiekvienos pozicijos daugiausiai laiko žaidusį krepšininką. Atleiskite iš komandos du mažiausiai laiko žaidusius krepšininkus (pašalinkite jų rezultatus ir jų duomenis).

U5–7. Aukščiausia lyga

Pirmo failo pirmoje failo eilutėje pateiktas aukščiausios futbolo lygos komandų skaičius. Tolesnėse eilutėse pateikta informacija apie futbolo komandas: pavadinimas, miestas, trenerio pavardė, vardas. Kitame faile pateikta I rato rezultatų lentelė, išreikšta pelnytais įvarčiais. Suskaičiuokite kiekvienos komandos surinktų taškų skaičių, jei už pergalę skiriami 3 taškai, o už lygiąsias – 1 taškas. Sudarykite komandų turnyrinę lentelę – surikiuokite surinktų taškų mažėjimo tvarka. Jei komandos surinko taškų vienodai, aukščiau ta komanda, kuri turi daugiau pergalių. Suraskite mažiausiai pralaimėjimų turinčią komandą (-as).

U5–8. Krepšinis

Pirmo failo pirmoje failo eilutėje nurodytas krepšinio komandų skaičius. Tolesnėse eilutėse pateikta informacija apie krepšinio komandas: pavadinimas, miestas, trenerio pavardė, vardas. Kitame faile pateikta I rato rezultatų lentelė, išreikšta įmestais taškais. Suskaičiuokite kiekvienai komandai bendrą įmestų ir praleistų taškų skirtumą. Surikiuokite komandas pagal šį skirtumą mažėjimo tvarka. Sudarykite komandų turnyrinę lentelę (surikiuokite pagal pergalių kiekį mažėjimo tvarka), jei už pergalę skiriami 2 taškai, o už pralaimėjimą – 1 taškas, lygiųjų nebūna. Jei komandos surinko po vienodai taškų, tai aukščiau ta komanda, kuri įmetė daugiau taškų tarpusavio rungtynėse, jei vienodai taškų, vertinkite tarpusavio taškų skirtumą. Suraskite daugiausiai taškų įmetusią komandą. Nustatykite, ar yra komanda, kuri iškovojo visas pergales.

U5–9. Detalės

Pirmo failo pirmoje failo eilutėje nurodytas darbininkų skaičius, mėnesio dirbtų dienų skaičius, vienos detalės įkainis. Tolesnėse eilutėse pateikta informacija apie darbininkus: pavardė, vardas, banko pavadinimas ir sąskaitos numeris. Kitame faile pateikta, kiek kiekvieną dieną (stulpeliai) darbininkas (eilutės) pagamino detalių. Suskaičiuokite kiekvienam darbininkui jo atlyginimą. Nustatykite, kuriam darbininkui geriausiai sekėsi dirbti. Nustatykite, kurią mėnesio dieną buvo pagaminta mažiausiai detalių. Kiekvienam bankui atskirai sudarykite pavedimų sąrašą, kur nurodysite darbininko pavardę, vardą, sąskaitos numerį ir pervedamą sumą.

U5–10. Leidiniai

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

U5–11. Paskaitos

Pirmo failo pirmoje failo eilutėje nurodytas studentų kiekis ir mėnesio dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, fakultetas, specialybė. Kitame faile pateikta informacija apie studentų paskaitų lankomumą: studentai (eilutės), praleista paskaitų per dieną (stulpeliai). Nustatykite, ar yra dienų, kai paskaitose dalyvavo visi studentai, jei taip, kurios tai dienos. Surikiuokite studentus pagal specialybes. Suskaičiuokite kiekvienai specialybei praleistų valandų kiekį. Nustatykite, ar yra studentų, kuriuos reikia šalinti (nelankė paskaitų daugiau nei pusę mėnesio dienų). Pašalinkite tokius studentus.

U5–12. Dėstytojai

Pirmo failo pirmoje failo eilutėje nurodytas dėstytojų kiekis, mėnesio dienų skaičius, vidutinis dėstytojo mėnesio valandų krūvis. Tolesnėse failo eilutėse nurodyta informacija apie dėstytojus: dėstytojo pavardė, vardas, fakultetas, katedra. Kitame faile pateikta informacija apie dėstytojų skaitytų paskaitų kiekį kiekvieną dieną: dėstytojai (eilutės), valandų kiekis per dieną (stulpeliai). Nustatykite, ar yra dienų, kai paskaitas skaitė visi dėstytojai, jei taip, kurios tai dienos. Surikiuokite dėstytojus pagal fakultetus. Nustatykite, kiekvieno fakulteto dėstytojų dirbtų valandų skaičių. Nustatykite, ar yra dėstytojų, kurie dirbo mažiau, nei vidutinis mėnesio dėstytojo darbo krūvis. Jei taip, kurie tai dėstytojai.

U5–13. Rankinis

Pirmo failo pirmoje failo eilutėje nurodytas rankinio komandos pavadinimas, vyr. treneris, rankininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie rankininkus: pavardė ir vardas, gimimo metai, masė, žaidimo pozicija. Kitame faile atitinkamai rankininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) įmestų taškų skaičius. Suraskite du rezultatyviausius komandos rankininkus. Atleiskite iš komandos du mažiausiai taškų pelniusius rankininkus (pašalinkite jų rezultatus).

U5–14. Formulė 1

Pirmo failo pirmoje failo eilutėje nurodytas lenktynėse dalyvaujančių pilotų skaičius ir trasų skaičius. Tolesnėse eilutėse pateikta informacija apie pilotus: pavardė ir vardas, komanda, variklio pavadinimas. Kitame faile pateikta sezono lenktynių atskirose trasose rezultatų lentelė, išreikšta pilotų užimtomis vietomis. Sudarykite pilotų turnyrinę lentelę. Taškai skiriami: 1 vieta – 10 taškų, 2 – 8 t., 3 – 6 t., 4 – 5 t., 5 – 4 t., 6 – 3 t., 7 – 2 t., 8 – 1 t.. Surikiuokite lentelę surinktų taškų mažėjimo tvarka. Suraskite, kuris pilotas daugiausiai kartų lenktynėse buvo antras. Suraskite, kiek ir kokios komandos dalyvavo sezono varžybose.

U5–15. Semestro darbai

Pirmo failo pirmoje failo eilutėje nurodytas studentų skaičius ir laboratorinių darbų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, grupė, specialybė. Kitame faile pateikta informacija apie studentų laboratorinių darbų įvertinimus (realūs skaičiai): studentai (eilutės), įvertinimai (stulpeliai). Nustatykite, ar yra tokių laboratorinių darbų, kurių visi įvertinimai teigiami (> 4.5), jei taip, tai kurie. Surikiuokite studentus pagal vidurkius. Suskaičiuokite kiekvienos grupės kiekvieno laboratorinio darbo įvertinimų vidurkį. Nustatykite, ar yra studentų, kuriuos reikia šalinti (vidurkis < 3). Pašalinkite tokius studentus.

U5–16. Gripas Lietuvoje

Pirmo failo pirmoje failo eilutėje nurodytas savivaldybių skaičius ir dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie savivaldybes: savivaldybės pavadinimas, savivaldybės dydis kvadratiniais kilometrais, gyventojų skaičius tūkstančiais. Kitame faile pateikta informacija apie gyventojų sergamumą – kiek gyventojų kreipėsi į gydymo įstaigas atitinkamomis dienomis: savivaldybės (eilutės), dienos (stulpeliai). Nustatykite, kuriose savivaldybėse ir kuriomis dienomis buvo paskelbta gripo epidemija. Gripo epidemija skelbiama, kai 10 tūkstančių gyventojų tenka x sergančiųjų. Kuriose savivaldybėse gripo epidemija tęsėsi ilgiausiai. Surikiuokite savivaldybes pagal epidemijos dienų skaičių ir savivaldybių pavadinimus.

U5–17. Gimstamumas Europoje

Pirmo failo pirmoje failo eilutėje nurodytas valstybių skaičius ir metų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie valstybes: valstybės pavadinimas, valstybės pavadinimo sutrumpinimas, gyventojų skaičius tūkstančiais. Kitame faile pateikta informacija apie gimstamumą – kiek naujagimių gimė atitinkamais metais: valsybės (eilutės), metai (stulpeliai). Nustatykite, kuriais metais buvo mažiausias gimstamumas. Suraskite, kuriais metais ir kurioje valstybėje gimstamumas buvo didžiausias 100-tui tūkstančių gyventojų. Surikiuokite valstybes pagal gyventojų skaičių ir valstybių pavadinimus.

U5–18. Disko metimas

Pirmo failo pirmoje failo eilutėje nurodytas sportininkų skaičius ir metimų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie sportininkus: pavardė ir vardas, valstybės pavadinimo, sportininko amžius ir sportininko masė. Kitame faile pateikta informacija apie disko nuskrietus atstumus metrais: sportininkai (eilutės), atstumai (6 stulpeliai – bandymai). Suraskite tris sportininkus, užėmusius tris pirmas vietas. Suraskite, kurią vietą užėmė vyriausias sportininkas. Surikiuokite sportininkus pagal pirmojo bandymo rezultatus ir sportininkų pavardes.

U5–19. Pražangos

Pirmo failo pirmoje failo eilutėje nurodytas komandos pavadinimas ir valstybė, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: vardas ir pavardė, gimimo metai, ūgis, žaidimo pozicija. Kitame faile atitinkamai krepšininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) gautos pražangos. Suraskite du mažiausiai ir du daugiausiai kartų prasižengusius krepšininkus. Kuriose rungtynėse krepšininkai surinko mažiausiai pražangų. Suraskite, kurios pozicijos žaidėjai surinko daugiausiai pražangų.

U5–20. Kasininkai

Pirmo failo pirmoje failo eilutėje nurodytas prekybos tinklo kasininkų/-ių skaičius ir dirbtų dienų skaičius. Tolesnėse eilutėse pateikta informacija apie kasininką/-ę: pavardė ir vardas, gimimo metai ir kasos, kurioje dirba, numeris. Kitame faile pateikta, kiek kiekvieną dieną (eilutės) kasininkas (stulpeliai) surinko pinigų aptarnaudamas pirkėjus. Suskaičiuokite kiekvienam kasininkui/-ei jo/-s atlyginimą, jeigu jis yra 1 % nuo kasoje surinktų pinigų. Suskaičiuokite, kokį atlygį gaus jauniausias ir vyriausias kasininkas/-ė. Suraskite, kurią dieną vidutiniškai buvo didžiausi prekių pardavimai.

U5–21. Būsimieji pirmakursiai

Pirmo failo pirmoje failo eilutėje nurodytas fakultetų skaičius ir dokumentų pateikimo į universitetą dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie fakultetus: fakulteto pavadinimas, planuojamas priimti studentų skaičius, dėstytojų skaičius. Kitame faile pateikta informacija apie studentų dokumentų pateikimo skaičius atitinkamomis dienomis: fakultetai (stulpeliai), dienos (eilutės). Nustatykite, kuris pirmasis fakultetas ir kurią dieną įvykdė studentų planuojamo priėmimo planą. Kurie fakultetai neįvykdė studentų priėmimo plano. Kurio fakulteto dėstytojams bus daugiausiai darbo – vienam dėstytojui teks vidutiniškai daugiausiai studentų. Surikiuokite fakultetus pagal priimtų dokumentų skaičius.

U5–22. Bankų indėlių palūkanos

Pirmo failo pirmoje failo eilutėje nurodytas Lietuvoje veikiančių bankų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie bankus: pavadinimas, valstybė, veikiančių skyrių skaičius Lietuvoje. Kitame faile pateikta informacija apie bankų (eilutės) palūkanų dydžius litais 1 mėn., 2 mėn., 3 mėn., 6 mėn., 1 metams, 2 metams ir 3 metams. Nustatykite, kuriuose bankuose geriausiai laikyti savo indėlius kiekvienam laikotarpiui. Kelių valstybių bankai veikia Lietuvoje? Surikiuokite bankus pagal palūkanas 6 mėnesiams ir bankų pavadinimus.

U5–23. Darbo birža

Pirmo failo pirmoje failo eilutėje nurodytas miestų skaičius ir mėnesių skaičius. Tolesnėse failo eilutėse nurodyta informacija apie miestus: miesto pavadinimas, gyventojų skaičius, jaunimo nuo 19 iki 25 metų skaičius. Kitame faile pateikta informacija apie jaunimo nuo 19 iki 25 metų nedarbą miestuose: miestai (eilutės), kiek bedarbių registruota kiekvieną mėnesį (stulpeliai). Nustatykite, kurį mėnesį buvo didžiausias nedarbas jaunimo tarpe. Suraskite, kurį mėnesį ir kuriame mieste santykinis nedarbo lygis buvo mažiausias. Surikiuokite miestus pagal jaunimo skaičių ir gyventojų skaičių.

U5–24. Oficiali emigracija

Pirmo failo pirmoje failo eilutėje nurodytas savivaldybių skaičius ir mėnesių skaičius. Tolesnėse failo eilutėse nurodyta informacija apie savivaldybes: savivaldybės pavadinimas, meras, gyventojų skaičius tūkstančiais. Kitame faile pateikta informacija apie gyventojų emigraciją – kiek gyventojų emigravo atitinkamais mėnesiais: savivaldybės (eilutės), mėnesiai (stulpeliai). Raskite, kiek gyventojų emigravo kiekvieną mėnesį. Nustatykite, kurioje savivaldybėse ir kurį mėnesį emigravo daugiausiai gyventojų. Surikiuokite savivaldybes pagal santykinį emigracijos dydį, įvertinant savivaldybių gyventojų skaičių ir savivaldybių pavadinimus.

U5–25. Universitetas

Pirmo failo pirmoje failo eilutėje nurodytas studentų kiekis ir mėnesio dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, gimimo metai, fakultetas, grupė. Kitame faile pateikta informacija apie studentų paskaitų lankomumą: studentai (stulpeliai), praleista paskaitų per dieną (eilutės). Nustatykite, ar yra dienų, kai paskaitose dalyvavo visi studentai, jei taip, kurios tai dienos. Surikiuokite studentus pagal grupes ir pavardes alfabetiškai. Suskaičiuokite, kiek kiekvieno fakulteto studentai praleido paskaitų. Nustatykite, ar yra studentų, kuriuos reikia šalinti (nelankė paskaitų daugiau nei pusę mėnesio dienų). Pašalinkite tokius studentus.

U5–26. Čempionų lyga

Pirmo failo pirmoje failo eilutėje nurodytas futbolo komandų skaičius. Tolesnėse eilutėse pateikta informacija apie futbolo komandas: valstybė, pavadinimas, trenerio pavardė, vardas. Kitame faile pateikta rezultatų lentelė, išreikšta pelnytais įvarčiais. Suskaičiuokite kiekvienos komandos surinktų taškų skaičių, jei už pergalę skiriami 3 taškai, o už lygiąsias – 1 taškas. Sudarykite komandų turnyrinę lentelę – surikiuokite surinktų taškų mažėjimo tvarka. Jei komandos surinko taškų vienodai, aukščiau ta komanda, kuri turi daugiau pergalių. Suraskite daugiausiai pergalių iškovojusią komandą(as). Suraskite komandas, kurios daugiausiai rungtynių nepraleido įvarčių.

U5–27. Parduotuvė

Pirmo failo pirmoje failo eilutėje nurodytas parduotuvės kasininkų/-ių skaičius ir dirbtų dienų skaičius. Tolesnėse eilutėse pateikta informacija apie kasininką/-ę: pavardė ir vardas, gimimo metai ir kasos, kurioje dirba, numeris. Kitame faile pateikta, kiek kiekvieną dieną (stulpelis) kasininkas (eilutė) surinko pinigų aptarnaudamas pirkėjus. Suskaičiuokite kiekvienam kasininkui/-ei jo/-s atlyginimą, jeigu jis yra 1 % nuo kasoje surinktų pinigų. Suskaičiuokite, kokį atlygį gaus jauniausias ir vyriausias kasininkas/- ė (gali būti keli). Suraskite, kurią dieną/as buvo didžiausi prekių pardavimai.

U5–28. Krepšinio varžybos

Pirmo failo pirmoje failo eilutėje nurodytas komandos pavadinimas, valstybė, krepšininkų skaičius ir žaistų rungtynių skaičius. Tolesnėse eilutėse pateikta informacija apie krepšininkus: vardas ir pavardė, gimimo metai, ūgis, žaidimo pozicija. Kitame faile atitinkamai krepšininkų (eilutės) išdėstymo tvarka pateikta kiekvienose rungtynėse (stulpeliai) gautos pražangos. Suraskite jauniausią ir vyriausią daugiausiai kartų prasižengusius krepšininkus (gali būti po kelis). Kuriose rungtynėse krepšininkai surinko mažiausiai pražangų. Suraskite, kurios pozicijos/ų žaidėjai surinko daugiausiai pražangų.

U5–29. Ieties metimas

Pirmo failo pirmoje failo eilutėje nurodytas sportininkų skaičius ir metimų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie sportininkus: pavardė ir vardas, valstybės pavadinimo, sportininko amžius. Kitame faile pateikta informacija apie ieties nuskrietus atstumus metrais: sportininkai (eilutės), atstumai (6 stulpeliai – bandymai). Suraskite tris sportininkus, užėmusius tris pirmas vietas. Jei yra keli, pirmas tas, kurio bandymo numeris mažesnis. Suraskite, kurią vietą užėmė vyriausias sportininkas/ai. Surikiuokite sportininkus pagal pirmojo bandymo rezultatus ir sportininkų pavardes.

U5–30. Semestro kontroliniai

Pirmo failo pirmoje failo eilutėje nurodytas studentų skaičius ir kontrolinių darbų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė, vardas, grupė, specialybė. Kitame faile pateikta informacija apie studentų kontrolinių darbų įvertinimus: studentai (eilutės), įvertinimai (stulpeliai). Nustatykite, ar yra tokių kontrolinių darbų, kurių visi įvertinimai teigiami (>= 5), jei taip, tai kurie. Surikiuokite studentus pagal vidurkius. Suskaičiuokite kiekvienos grupės kiekvieno kontrolinio darbo įvertinimų vidurkį. Nustatykite, ar yra studentų, kuriuos reikia šalinti (vidurkis < 3). Pašalinkite tokius studentus.

U5–31 Lenktynės

Pirmo failo pirmoje failo eilutėje nurodytas lenktynėse dalyvaujančių pilotų skaičius ir trasų skaičius. Tolesnėse eilutėse pateikta informacija apie pilotus: pavardė ir vardas, komanda. Kitame faile pateikta sezono lenktynių atskirose trasose rezultatų lentelė, išreikšta pilotų užimtomis vietomis. Sudarykite pilotų turnyrinę lentelę. Taškai skiriami: 1 vieta – 10 taškų, 2 – 8 t., 3 – 7 t., 4 – 6 t., 5 – 5 t., 6 – 4 t., 7 – 3 t., 8 – 2 t., 9 – 1 t. Surikiuokite lentelę surinktų taškų mažėjimo tvarka. Suraskite, kuris pilotas/ai daugiausiai kartų lenktynėse buvo antras/i. Suraskite, kiek ir kokios komandos dalyvavo sezono varžybose.

U5–32. Paskaitų lankomumas

Pirmo failo pirmoje failo eilutėje nurodytas studentų kiekis ir mėnesio dienų skaičius. Tolesnėse failo eilutėse nurodyta informacija apie studentus: studento pavardė ir vardas, fakultetas, grupė. Kitame faile pateikta informacija apie studentų paskaitų lankomumą: studentai (eilutės), praleista paskaitų per dieną (stulpeliai). Nustatykite, ar yra dienų, kai paskaitose dalyvavo visi studentai, jei taip, kurios tai dienos. Surikiuokite studentus pagal grupes ir pavardes vardus alfabetiškai. Suskaičiuokite kiekvienai grupei praleistų valandų kiekį. Nustatykite, ar yra studentų, kuriuos reikia šalinti (nelankė paskaitų daugiau nei pusę mėnesio dienų). Pašalinkite tokius studentus.