ktu informatikos fakultetas 1922

T09. Bendrinės klasės

2 ak. val.

ktu informatikos fakultetas 1922

Temos klausimai

1. Bendriniai metodai.
2. Bendrinės klasės.
3. Bendrinių klasių kolekcijos.

ktu informatikos fakultetas 1922

Bendrinės klasės, bendriniai metodai

Bendrinės metodai ir bendrinės klasės – vienos galingiausių C# 2.0 priemonių. Jos leidžia dirbti su duomenų struktūromis, nenurodant konkretaus duomenų tipo.

Bendrinės metodai ir bendrinės klasės yra panašu į C++ šablonus. Bet tuo pačiu ir smarkiai skiriasi.

ktu informatikos fakultetas 1922

Bendriniai metodai

Skirtingo tipo duomenų masyvų spausdinimas 1/6

```
// Sveikų skaičių masyvo spausdinimas
static void Spausdinti(int[] sk, string antraste)
{
    Console.WriteLine(antraste);
    foreach (int el in sk)
        Console.Write(el + "  ");
    Console.WriteLine();
}
```

Skirtingo tipo duomenų masyvų spausdinimas 2/6

```
// Realių skaičių masyvo spausdinimas
static void Spausdinti(double[] sk, string antraste)
{
    Console.WriteLine(antraste);
    foreach (double el in sk)
        Console.Write(el + "  ");
    Console.WriteLine();
}
```

Skirtingo tipo duomenų masyvų spausdinimas 3/6

```
// Simbolių masyvo spausdinimas
static void Spausdinti(char[] sk, string antraste)
{
    Console.WriteLine(antraste);
    foreach (char el in sk)
        Console.Write(el + "  ");
    Console.WriteLine();
}
```

Skirtingo tipo duomenų masyvų spausdinimas 4/6

```
// Eilučių masyvo spausdinimas
static void Spausdinti(string[] sk, string antraste)
{
    Console.WriteLine(antraste);
    foreach (string el in sk)
        Console.Write(el + "  ");
    Console.WriteLine();
}
```

Skirtingo tipo duomenų masyvų spausdinimas 5/6

```
static void Main(string[] args)
{
    int[] Sveiki = { 1, 2, 3, 4, 5, 6, 7 };
    double[] Realus = { 10.1, -11, 12.8, -14, -775.62, 0, 9, -666.66 };
    char[] Simboliai = { 'P', 'a', 's', 'k', 'a', 'i', 't', 'a'};
    string[] Eilutės = { "Pirmadienis", "Antradienis", "Trečiadienis",
                         "Ketvirtadienis", "Penktadienis" };
    Spausdinti(Sveiki, "Sveikieji skaičiai");
    Console.WriteLine();
    Spausdinti(Realus, "Realieji skaičiai");
    Console.WriteLine();
    Spausdinti(Simboliai, "Simboliai");
    Console.WriteLine();
    Spausdinti(Eilutės, "Eilutės");
}
```

Skirtingo tipo duomenų masyvų spausdinimas 6/6

```
Sveikieji skaičiai
1  2  3  4  5  6  7

Realieji skaičiai
10,1  -11  12,8  -14  -775,62  0  9  -666,66

Simboliai
P  a  s  k  a  i  t  a

Eilutės
Pirmadienis  Antradienis  Trečiadienis  Ketvirtadienis  Penktadienis
Press any key to continue . . .
```

Bendrinis metodas

Bendrinio metodo užrašymai:

[<modifikatoriai>]<atsakymo_tipas><metodo_vardas><T>(<parametrai>) {}

[<modifikatoriai>]<metodo_vardas><T>(<parametrai>) {}

<T> - tipas arba tipų sąrašas. Visiems jiems turi būti galimos kamiene atliekamos operacijos. Tipas T gali būti naudojamas ir parametrų sąraše.

Būtina rašyti **<T>**

Bendrinio metodo panaudojimas 1/3

```
// Pasirinkto tipo masyvo spausdinimas
static void Spausdinti<Tipas>(Tipas[] sk, string antraste)
{
    Console.WriteLine(antraste);
    foreach (Tipas el in sk)
        Console.Write(el + "  ");
    Console.WriteLine();
}
```

Bendrinio metodo panaudojimas 2/3

```
static void Main(string[] args)
{
    int[] Sveiki = { 1, 2, 3, 4, 5, 6, 7 };
    double[] Realus = { 10.1, -11, 12.8, -14, -775.62, 0, 9, -666.66 };
    char[] Simboliai = { 'P', 'a', 's', 'k', 'a', 'i', 't', 'a'};
    string[] Eilutės = { "Pirmadienis", "Antradienis", "Trečiadienis",
                         "Ketvirtadienis", "Penktadienis" };
    Spausdinti<int>(Sveiki, "Sveikieji skaičiai");
    Console.WriteLine();
    Spausdinti<double>(Realus, "Realieji skaičiai");
    Console.WriteLine();
    Spausdinti<char>(Simboliai, "Simboliai");
    Console.WriteLine();
    Spausdinti<string>(Eilutės, "Eilutės");
}
```

Bendrinio metodo panaudojimas 3/3

```
Sveikieji skaičiai
1  2  3  4  5  6  7

Realieji skaičiai
10,1  -11  12,8  -14  -775,62  0  9  -666,66

Simboliai
P  a  s  k  a  i  t  a

Eilutės
Pirmadienis  Antradienis  Trečiadienis  Ketvirtadienis  Penktadienis
Press any key to continue . . .
```

Tinka ir pradinis pagrindinio metodo **`Main`** variantas.

ktu informatikos fakultetas 1922

Tipų sąrašas 1/3

```
// spausdina dviejų masyvų elementus
static void Spausdinti<Tipas1, Tipas2>(int n, Tipas1[] sk,
                        Tipas2 [] sk1)
{
    for (int i = 0; i < n; i++)
        Console.WriteLine("{0} elementai: {1}   {2}", i + 1,
                            sk[i], sk1[i]);
    Console.WriteLine();
}
```

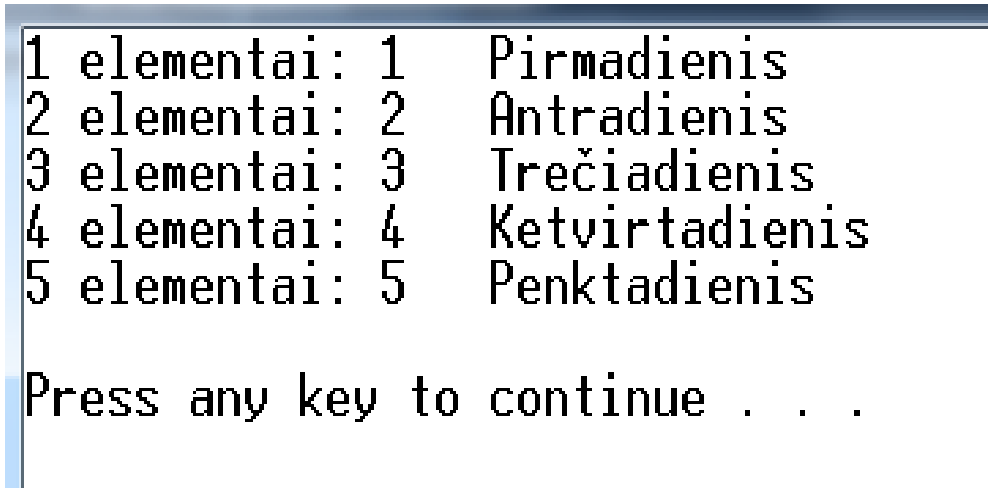
ktu informatikos fakultetas 1922

Tipų sąrašas 2/3

```
static void Main(string[] args)
{
    int[] Numeriai = { 1, 2, 3, 4, 5};
    string[] Dienos = { "Pirmadienis", "Antradienis", "Trečiadienis",
                        "Ketvirtadienis", "Penktadienis" };
    Spausdinti<int, string>(Numeriai.Count(), Numeriai, Dienos);
}
```

ktu informatikos fakultetas 1922

Tipų sąrašas 3/3



Bendrinio tipo apribojimai 1/4

Apribojimas	Aprašymas
where T:<sąsajos vardas>	Tipas privalo būti ar įgyvendinti nurodytą sąsają
where T:<bazinės klasės vardas>	Tipas privalo būti ar būti išvestinis iš nurodytos klasės
where T:U	Tipas privalo būti ar būti išvestinis iš nurodyto argumento tipo U
where T:new()	Tipas privalo turėti atvirą numatytąjį konstruktorių
where T:struct	Tipas turi būti reikšmės tipas
where T:class	Tipas privalo būti nuorodos tipas

Bendrinio tipo apribojimai 2/4

```
// randa didžiausią iš trijų elementų
static tipas Maximum<tipas>(tipas x, tipas y, tipas z)
       where tipas: IComparable<tipas>
{
    tipas max = x;
    // įvertina current culture
    if (y.CompareTo(max) > 0) max = y;
    if (z.CompareTo(max) > 0) max = z;
    return max;
}
```

Bendrinio tipo apribojimai 3/4

```
static void Main(string[] args)
{
    Console.WriteLine(" Maximum iš sveikųjų {0}, {1} ir {2} yra {3}\n",
                      -1, 2, 3, Maximum(-1, 2, 3));
    Console.WriteLine(" Maximum iš realiųjų {0}, {1} ir {2} yra {3}\n",
                       1.1, 2.2, -3.3, Maximum(1.1, 2.2, -3.3));
    Console.WriteLine(" Maximum iš žodžių {0}, {1} ir {2} yra {3}\n",
                      "katė", "šuo", "pelėda",
                       Maximum("katė", "šuo", "pelėda"));
}
```

Bendrinio tipo apribojimai 4/4

```
Maximum iš sveikųjų -1, 2 ir 3 yra 3

Maximum iš realiųjų 1,1, 2,2 ir -3,3 yra 2,2

Maximum iš žodžių katė, šuo ir pelėda yra šuo

Press any key to continue . . .
```

ktu informatikos fakultetas 1922

Keli apribojimai

Jie atskiriami kableliais.

1. Jei turite klasės, reikšmės ar nuorodos apribojimą, tai galima naudoti tik vieną iš jų, ir jis įrašomas pirmasis.
2. Po to eina sąsajos apribojimai.
3. Gale užrašomas apribojimas konstruktoriui.

Galima užkloti bendrinį metodą kitu bendriniu metodu arba nebendriniu.

Bendrinės klasės

Bendrinė klasė

Bendrinės klasės užrašas:

[<modifikatoriai>] `klasė<klasės_vardas><T>`

{

}

<T> - tipas arba tipų sąrašas. Būtina rašyti **<T>**

ktu informatikos fakultetas 1922

Pavyzdys 1/16

Užduotis.

Tekstiniuose failuose "`Darius.txt`" ir "`Mikas.txt`" yra duomenys apie dviejų studentų turimus mobiliosios elektronikos įrenginius. Pirmoje eilutėje yra studento vardas ir pavardė. Kitose eilutėse yra informacija apie įrenginius: modelis, tipas, baterijos veikimo trukmė. Parašykite programą, kuri suformuotų pasirinkto tipo įrenginių sąrašą.

Kiekvieną sąrašą surikiuokite pagal veikimo trukmę ir modelį abėcėliškai.

Pavyzdys 2/16

Duomenų failo `Darius.txt` **pavyzdys**

```
Darius Elektronikas
Creative Zen;MP3 grotuvas;                  40
Motorola Razor V3;Mobilusis telefonas;     400
Apple iPod Shuffle;MP3 grotuvas;            40
Sony Ericsson K610i;Mobilusis telefonas;   480
```

Duomenų failo `Mikas.txt` **pavyzdys**

```
Mikas Humanitaras
Garmin Oregon 300;Navigacine sistema;   16
Motorola W156;Mobilusis telefonas;     465
Samsung B520;Mobilusis telefonas;      350
Sony Walkman NWZ-A826B;MP3 grotuvas;    45
Sony Walkman NWZ-E436FB;MP3 grotuvas;   45
Nokia 1200;Mobilusis telefonas;        390
```

ktu informatikos fakultetas 1922

Pavyzdys 3/16

Mikas Humanitaras

Modelis	Tipas	Veik. trukmė
Nokia 1200	Mobilusis telefonas	390,00
Sony Walkman NWZ-E436FB	MP3 grotuvas	45,00
Sony Walkman NWZ-A826B	MP3 grotuvas	45,00
Samsung B520	Mobilusis telefonas	350,00
Motorola W156	Mobilusis telefonas	465,00
Garmin Oregon 300	Navigacine sistema	16,00

Darius Elektronikas

Modelis	Tipas	Veik. trukmė
Sony Ericsson K610i	Mobilusis telefonas	480,00
Apple iPod Shuffle	MP3 grotuvas	40,00
Motorola Razor V3	Mobilusis telefonas	400,00
Creative Zen	MP3 grotuvas	40,00

Pirmas surikiuotas

Modelis	Tipas	Veik. trukmė
Garmin Oregon 300	Navigacine sistema	16,00
Sony Walkman NWZ-A826B	MP3 grotuvas	45,00
Sony Walkman NWZ-E436FB	MP3 grotuvas	45,00
Samsung B520	Mobilusis telefonas	350,00
Nokia 1200	Mobilusis telefonas	390,00
Motorola W156	Mobilusis telefonas	465,00

Antras surikiuotas

Modelis	Tipas	Veik. trukmė
Apple iPod Shuffle	MP3 grotuvas	40,00
Creative Zen	MP3 grotuvas	40,00
Motorola Razor V3	Mobilusis telefonas	400,00
Sony Ericsson K610i	Mobilusis telefonas	480,00

Pavyzdys 4/16

Atrinkti nerikiuoti

Modelis	Tipas	Veik. trukmė
Sony Ericsson K610i	Mobilusis telefonas	480,00
Motorola Razor V3	Mobilusis telefonas	400,00
Motorola W156	Mobilusis telefonas	465,00
Nokia 1200	Mobilusis telefonas	390,00
Samsung B520	Mobilusis telefonas	350,00

Atrinkti surikiuoti

Modelis	Tipas	Veik. trukmė
Samsung B520	Mobilusis telefonas	350,00
Nokia 1200	Mobilusis telefonas	390,00
Motorola Razor V3	Mobilusis telefonas	400,00
Motorola W156	Mobilusis telefonas	465,00
Sony Ericsson K610i	Mobilusis telefonas	480,00

Atrenkamų įrenginių tipas – Mobilusis telefonas.

ktu informatikos fakultetas 1922

Pavyzdys 5/16

```
// mazgo klasė
public sealed class Mazgas<Tipas>
    where Tipas:IComparable<Tipas>
{
    public Tipas Duomenys { get; set; }
    public Mazgas<Tipas> Kitas { get; set; }
    public Mazgas() { }
    public Mazgas(Tipas duomenys, Mazgas<Tipas> adresas)
    {
        this.Duomenys = duomenys;
        this.Kitas = adresas;
    }
}
```

ktu informatikos fakultetas 1922

Pavyzdys 6/16

```
// Mobiliojo įrenginio duomenų saugojimo klasė
public class Mobilus:IComparable<Mobilus>
{   // mobiliosios elektronikos įrenginys
    public string modelis { get; set; }  // modelio pavadinimas
    public string tipas { get; set; }    // įrenginio tipas
    public int baterija { get; set; }    // baterijos veikimo trukmė
    //  konstruktorius
    public Mobilus(string modelis = "", string tipas = "", int baterija = 0)
    {
        this.modelis = modelis;
        this.tipas = tipas;
        this.baterija = baterija;
    }
    //  objekto naujos reikšmės
    // a – modelio pavadinimas
    // b – įrenginio tipas
    // c – baterijos veikimo trukmė
    public void Dėti(string a, string b, int c)
    {
        modelis = a;
        tipas = b;
        baterija = c;
    }

              . . .
}
```

Pavyzdys 7/16

```
// Mobiliojo įrenginio duomenų saugojimo klasė
public class Mobilus:IComparable<Mobilus>
{   // tęsinys
    public override string ToString()
    {
        string eilute;
        eilute = string.Format("|{0, -30}| {1, -20} |   {2, 8:f}   |",
                        modelis, tipas, baterija);
        return eilute;
    }
    public override bool Equals(object objektas)
    {
        Mobilus telef = objektas as Mobilus;
        return telef.tipas == tipas && telef.modelis == modelis && telef.baterija ==
                    baterija;
    }
    public override int GetHashCode()
    {
        return base.GetHashCode();
    }
    . . .
}
```

ktu informatikos fakultetas 1922

Pavyzdys 8/16

```
// Mobiliojo įrenginio duomenų saugojimo klasė
public class Mobilus:IComparable<Mobilus>
{   // tęsinys1

    public int CompareTo(Mobilus kitas)
    {
        if (kitas == null) return 1;
        if (baterija.CompareTo(kitas.baterija) != 0)
            return baterija.CompareTo(kitas.baterija);
        else return modelis.CompareTo(kitas.modelis);
    }
}
```

ktu informatikos fakultetas

Pavyzdys 9/16

```
                // Mobiliųjų įrenginių vienkryptis sąrašas
public sealed class Sąrašas<Tipas>  where Tipas : IComparable<Tipas>
{
    Mazgas<Tipas> pr;      // sąrašo pradžios nuoroda
    Mazgas<Tipas> d;       // sąsajos nuoroda
    public Sąrašas()
    {
        this.pr = null;
        this.d = null;
    }
    // Sąsajai priskiriama sąrašo pradžia
    public void Pradžia()
    {
        d = pr;
    }
    // Sąsajai priskiriamas sąrašo sekantis elementas
    public void Kitas()
    {
        d = d.Kitas;
    }
    // Grąžina true, jeigu sąsaja netuščia; false - priešingu atveju
    public bool Yra()
    {
        return d != null;
    }
...
}
```

Pavyzdys 10/16

```
                // Mobiliųjų įrenginių vienkryptis sąrašas
public sealed class Sąrašas<Tipas>  where Tipas : IComparable<Tipas>
{   //  tęsinys
    // Grąžina pagalbinės rodyklės rodomo elemento reikšmę
    public Tipas ImtiDuomenis() { return d.Duomenys; }
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PRADŽIOS
    // inf – naujo elemento reikšmė (duomenys)
    public void DėtiDuomenisA(Tipas inf)
    {
        var d = new Mazgas<Tipas>(inf, null);
        d.Kitas = pr;
        pr = d;
        // arba
        //pr = new Mazgas(inf, pr);
    }
    // Sunaikinamas sąrašas
    public void Naikinti()
    {
        while (pr != null)
        {
            d = pr;
            pr = pr.Kitas;
            d = null;
        }
        d = pr;
    }       ...
}
```

ktu informatikos fakultetas 1922

Pavyzdys 11/16

```
                // Mobiliųjų įrenginių vienkryptis sąrašas
public sealed class Sąrašas<Tipas>  where Tipas : IComparable<Tipas>
{   //  tęsinys1
    //-------------------------------------------------------------
    // Rikiavimas išrinkimo būdu
    public void Rikiuoti()
    {
        for (Mazgas<Tipas> d1 = pr; d1.Kitas != null; d1 = d1.Kitas)
        {
            Mazgas<Tipas> maxv = d1;
            for (Mazgas<Tipas> d2 = d1; d2 != null; d2 = d2.Kitas)
                if (d2.Duomenys.CompareTo(maxv.Duomenys) < 0 )
                    maxv = d2;
            Tipas St = d1.Duomenys;
            d1.Duomenys = maxv.Duomenys;
            maxv.Duomenys = St;
        }
    }
}
```

Pavyzdys 12/16

```
class Program
{
    const string CFd1 = "..\\..\\Mikas.txt";
    const string CFd2 = "..\\..\\Darius.txt";
    const string CFr = "..\\..\\Rezultatai.txt";
    static void Main(string[] args)
    {
        Sąrašas<Mobilus> A,        // pirmojo studento duomenys
                         B;        // antrojo studento duomenys
        string [] Vardai = new string [3];
        A = SkaitytiAtv(CFd1, ref Vardai[0]);
        B = SkaitytiAtv(CFd2, ref Vardai[1]);
        if (File.Exists(CFr))
            File.Delete(CFr);
        Spausdinti(CFr, A, Vardai[0]);
        Spausdinti(CFr, B, Vardai[1]);
        A.Rikiuoti();
        Spausdinti(CFr, A, "Pirmas surikiuotas");

        B.Rikiuoti();
        Spausdinti(CFr, B, "Antras surikiuotas");
        ...
    }
}
```

Pavyzdys 13/16

```
    static void Main(string[] args)
    {   // tęsinys
        Sąrašas<Mobilus> Naujas = new Sąrašas<Mobilus>();    // atrinkti duomenys
        Console.WriteLine("Įveskite norimą įrenginio tipą:");
        string tipas = Console.ReadLine();  // Įvedamas norimas įrenginio tipas
        Atrinkti(A, tipas, Naujas);
        Atrinkti(B, tipas, Naujas);
        // --- Suformuoto sąrašo spausdinimas ir rikiavimas ---
        Naujas.Pradžia();
        if (Naujas.Yra())
        {
            Spausdinti(CFr, Naujas, "Atrinkti nerikiuoti");
            Naujas.Rikiuoti();
            Spausdinti(CFr, Naujas, "Atrinkti surikiuoti");
        }
        else
        {
            using (var failas = new StreamWriter(CFr, true))
            {
                failas.WriteLine("Naujas sąrašas nesudarytas.");
            }
        }
    }
// Program klasės metodai
}
```

ktu informatikos fakultetas 1922

Pavyzdys 14/16

```
// Skaitomi duomenys iš failo ir sudedami į sąrašą ATVIRKŠTINE tvarka
// fv – duomenų failo vardas
static Sąrašas<Mobilus> SkaitytiAtv(string fv, ref string vardas)
{
    var A = new Sąrašas<Mobilus>();
    using (var failas = new StreamReader(fv))
    {
        string eilute;
        vardas = eilute = failas.ReadLine();
        while ((eilute = failas.ReadLine()) != null)
        {
            string[] eilDalis = eilute.Split(';');
            string modelis = eilDalis[0];
            string tipas = eilDalis[1];
            int baterija = int.Parse(eilDalis[2]);
            Mobilus elem = new Mobilus(modelis, tipas, baterija);
            A.DėtiDuomenisA(elem);
        }
    }
    return A;
}
```

Pavyzdys 15/16

```
// Sąrašo duomenys spausdinami faile
// fv – duomenų failo vardas
// A  - sąrašo objekto nuoroda
// koment  - komentaras
static void Spausdinti(string fv, Sąrašas<Mobilus> A, string koment)
{
    using (var failas = new StreamWriter(fv, true))
    {
        failas.WriteLine(koment);
        failas.WriteLine("+------------------------------+--------------" +
                         "-------+-------------+");
        failas.WriteLine("|             Modelis          |        Tipas  " +
                         "       | Veik. trukmė |");
        failas.WriteLine("+------------------------------+--------------" +
                         "-------+-------------+");
        // Sąrašo peržiūra, panaudojant sąsajos metodus
        for (A.Pradžia(); A.Yra(); A.Kitas())
        {
            failas.WriteLine("{0}", A.ImtiDuomenis().ToString());
        }
        failas.WriteLine("+------------------------------+--------------" +
                         "-------+-------------+");
        failas.WriteLine();
    }
}
```

ktu informatikos fakultetas 1922

Pavyzdys 16/16

```
// Formuojamas naujas sąrašas
// senas  – turimas sąrašas
// naujas – formuojamas sąrašas
// tipas  - atrenkamų įrenginių tipas
static void Atrinkti(Sąrašas<Mobilus> senas, string tipas, Sąrašas<Mobilus> naujas)
{
    for (senas.Pradžia(); senas.Yra(); senas.Kitas())
    {
        Mobilus duom = senas.ImtiDuomenis();
        if (duom.tipas == tipas)
            naujas.DėtiDuomenisA(duom);
    }
}
```

ktu informatikos fakultetas 1922

Pavyzdys_2 1/11

Užduotis.

Tekstiniame faile „`Asmenys.txt`" yra atvykstančių žmonių registracijos duomenys: pavardė vardas, amžius, atvykimo laikas. Parašykite programą, kuri suformuotų vyresnių nei nurodyto amžiaus žmonių sąrašą.

Kiekvieną sąrašą surikiuokite abėcėliškai ir pagal amžių didėjimo tvarka. Spendimui panaudokite prieš tai spręsto uždavinio bendrinę sąrašo klasę.

Pavyzdys_2 2/11

Duomenų failo `Asmenys.txt` **pavyzdys**

```
Jonaitis Jonas;   20; 8:20:00
Petraitis Petras; 38; 7:50:00
Petraitis Petras; 18; 7:50:00
Petraitis Petras; 60; 7:50:00
Rimaitis Rimas;   41; 8:24:00
Ritaitis Jonas;   28; 6:17:00
Augaitis Augis;   20; 8:26:00
```

ktu informatikos fakultetas 1922

Pavyzdys_2 3/11

Pradinis sąrašas

Pavardė, vardas	Metai	Atvykimo laikas
Augaitis Augis	20	08:26:00
Ritaitis Jonas	28	06:17:00
Rimaitis Rimas	41	08:24:00
Petraitis Petras	60	07:50:00
Petraitis Petras	18	07:50:00
Petraitis Petras	38	07:50:00

Surikiuotas pradinis sąrašas

Pavardė, vardas	Metai	Atvykimo laikas
Augaitis Augis	20	08:26:00
Petraitis Petras	18	07:50:00
Petraitis Petras	38	07:50:00
Petraitis Petras	60	07:50:00
Rimaitis Rimas	41	08:24:00
Ritaitis Jonas	28	06:17:00

Atrinkti nerikiuoti

Pavardė, vardas	Metai	Atvykimo laikas
Ritaitis Jonas	28	06:17:00
Rimaitis Rimas	41	08:24:00
Petraitis Petras	60	07:50:00
Petraitis Petras	38	07:50:00

Atrinkti surikiuoti

Pavardė, vardas	Metai	Atvykimo laikas
Petraitis Petras	38	07:50:00
Petraitis Petras	60	07:50:00
Rimaitis Rimas	41	08:24:00
Ritaitis Jonas	28	06:17:00

Atrenkamų žmonių amžius > 20.

Pavyzdys_2 5/11

```
// Asmens duomenų saugojimo klasė
public class Asmuo : IComparable<Asmuo>
{
    public string pavVard { get; set; }      // pavardė, vardas
    public int amžius { get; set; }          // amžius
    public TimeSpan laikas { get; set; }     // atvykimo laikas
    public Asmuo(string pavVard, int amžius, TimeSpan laikas)
    {
        this.pavVard = pavVard;
        this.amžius = amžius;
        this.laikas = laikas;
    }
    public override string ToString()
    {
        string eilute;
        eilute = string.Format("| {0, -30}    |       {1, -7}    |   {2}      |",
               pavVard, amžius, laikas);
        return eilute;
    }
                                    . . .
}
```

ktu informatikos fakultetas 1922

Pavyzdys_2 6/11

```
// Asmens duomenų saugojimo klasė
public class Asmuo : IComparable<Asmuo>
{   // tęsinys
    public int CompareTo(Asmuo kitas)
    {
        int poz = String.Compare(this.pavVard, kitas.pavVard,
                                 StringComparison.CurrentCulture);
        if (poz > 0) return 1;
        if (poz < 0) return -1;
        else
            if (this.amžius > kitas.amžius) return 1;
            else if (this.amžius < kitas.amžius) return -1;
            else return 0;
    }

    public override int GetHashCode()
    {
        return base.GetHashCode();
    }
}
```

ktu informatikos fakultetas 1922

Pavyzdys_2 7/11

```
class Program
{
    const string CFd1 = "..\\..\\Asmenys.txt";
    const string CFr = "..\\..\\Rezultatai.txt";

    static void Main(string[] args)
    {
        Sąrašas<Asmuo> A;         // asmenų sąrašas
        A = SkaitytiAtv(CFd1);

        if (File.Exists(CFr))
            File.Delete(CFr);

        Spausdinti(CFr, A, "Pradinis sąrašas");
        A.Rikiuoti();
        Spausdinti(CFr, A, "Surikiuotas pradinis sąrašas");
        ...
    }
}
```

Pavyzdys_2 8/11

```
    static void Main(string[] args)
    {   // tęsinys
        Sąrašas<Asmuo> Naujas = new Sąrašas<Asmuo>();    // atrinkti duomenys
        Console.WriteLine("Įveskite norimus metus:");
        int metai = int.Parse(Console.ReadLine());  // Įvedamas norimas ribinis amžius
        Atrinkti(A, metai, Naujas);
        Naujas.Pradžia();
        if (Naujas.Yra())
        {
            Spausdinti(CFr, Naujas, "Atrinkti nerikiuoti");
            Naujas.Rikiuoti();
            Spausdinti(CFr, Naujas, "Atrinkti surikiuoti");
        }
        else
        {
            using (var failas = new StreamWriter(CFr, true))
                failas.WriteLine("Naujas sąrašas nesudarytas.");
        }
  }
// Program klasės metodai
}
```

ktu informatikos fakultetas 1922

Pavyzdys_2 9/11

```
// Skaitomi duomenys iš failo ir sudedami į sąrašą ATVIRKŠTINE tvarka
// fv – duomenų failo vardas
static Sąrašas<Asmuo> SkaitytiAtv(string fv)
{
    var A = new Sąrašas<Asmuo>();
    using (var failas = new StreamReader(fv))
    {
        string eilute, vardas;
        vardas = eilute = failas.ReadLine();
        while ((eilute = failas.ReadLine()) != null)
        {
            string[] eilDalis = eilute.Split(';');
            string pavVard = eilDalis[0];
            int amžius = int. Parse(eilDalis[1]);
            TimeSpan laikas = TimeSpan.Parse(eilDalis[2]);
            Asmuo elem = new Asmuo(pavVard, amžius, laikas);
            A.DėtiDuomenisA(elem);
        }
    }
    return A;
}
```

Pavyzdys_2 10/11

```
// Sąrašo duomenys spausdinami faile
// fv – duomenų failo vardas
// A  - sąrašo objekto nuoroda
// koment  - komentaras
static void Spausdinti(string fv, Sąrašas<Asmuo> A, string koment)
{
    using (var failas = new StreamWriter(fv, true))
    {
        failas.WriteLine(koment);
        failas.WriteLine("+-----------------------------------+--------------" +
                         "---+-----------------+");
        failas.WriteLine("|             Pavardė, vardas       |     Metai     " +
                         "   | Atvykimo laikas |");
        failas.WriteLine("+-----------------------------------+--------------" +
                         "---+-----------------+");
        // Sąrašo peržiūra, panaudojant sąsajos metodus
        for (A.Pradžia(); A.Yra(); A.Kitas())
        {
            failas.WriteLine("{0}", A.ImtiDuomenis().ToString());
        }
        failas.WriteLine("+-----------------------------------+--------------" +
                         "---+-----------------+");
        failas.WriteLine();
    }
}
```

Pavyzdys_2 11/11

```
// Formuojamas naujas sąrašas
// senas  – turimas sąrašas
// naujas – formuojamas sąrašas
// metai  - amžiaus metai, už kurį turi būti didesni
static void Atrinkti(Sąrašas<Asmuo> senas, int metai, Sąrašas<Asmuo> naujas)
{
    for (senas.Pradžia(); senas.Yra(); senas.Kitas())
    {
        Asmuo duom = senas.ImtiDuomenis();
        if (duom.amžius > metai)
            naujas.DėtiDuomenisA(duom);
    }
}
```

ktu informatikos fakultetas 1922

Bendrinių klasių kolekcijos

ktu informatikos fakultetas 1922

Bendrinės masyvų ir sąrašų klasės

Klasė	Aprašymas
List<T>	Bendros paskirties kolekcija
LinkedList<T>	Susietasis dvikryptis sąrašas
LinkedListNode<T>	Dvikrypčio susietojo sąrašo mazgas
Queue<T>	Konteineris, įgyvendinantis eilės principą
Stack<T>	Konteineris, įgyvendinantis dėklo principą
Dictionary<TKey, TValue>	Bendros paskirties asociatyvus masyvas - žodynas
SortedDictionary<TKey, TValue>	Žodynas, surikiuotas pagal raktą. Naudoja daugiau atminties, nei SortedList. Efektyvesnė įterpimui ir šalinimui
SortedList<TKey, TValue>	Bendros paskirties rikiuota pagal raktą kolekcija

ICollection<T> sąsaja

Sąsaja apibrėžia funkcionalumą, būdingą visom bendrinėms kolekcijų klasėms.
Dėl šios sąsajos būtina įgyvendinti IEnumerable<T> ir IEnumerable.

Metodas	Aprašymas
Add	Papildo T tipo nariu kolekciją
Clear	Išvalo kolekciją
Contains	Nustato, ar nurodyta reikšmė yra kolekcijoje
CopyTo	Kopijuoja kolekcijos narius į masyvą
Count	Suskaičiuoja kolekcijos narių skaičių
IsReadOnly	Parodo, ar kolekcija yra tik skaitymui
Remove	Pašalina nurodytą narį iš kolekcijos

ktu informatikos fakultetas 1922

IList<T> sąsaja

Sąsaja apibrėžia bazinį sąrašinių bendrinių klasių funkcionalumą.
Ji paveldi ICollection<T>. Leidžia kolekcijai naudoti indeksus .

Metodas arba savybė	Aprašymas
Insert	Įterpia pagal nurodytą indeksą
RemoveAt	Pašalina pagal nurodytą indeksą
IndexOf	Nustato nurodyto nario poziciją

IDictionary<TKey, TValue> sąsaja

Sąsaja apibrėžia funkcionalumą, būdingą visoms bendrinėms asociatyvių masyvų klasėms.
Paveldi iš ICollection<T> .

Metodas	Aprašymas
Add	Papildo T tipo nariu kolekciją
ContainsKey	Patikrina, ar yra rakto reikšmė
GetEnumerator	Grąžina enumeratorių poroms <TKey, TValue>
Remove	Pašalina nurodytą narį iš kolekcijos pagal raktą
TryGetValue	Bando paimti reikšmę pagal raktą. True – jei raktas yra ir reikšmė priskiriama. False – jei nėra
Item	Priskiria arba paima reikšmę. Galima naudoti myDictionary[myKey] = myValue
Keys	Grąžina raktus į ICollection<T> tipą
Values	Grąžina reikšmes į ICollection<T> tipą

Klausimai?