

T08. Dvikryptis sąrašas

2 ak. val.

Temos klausimai

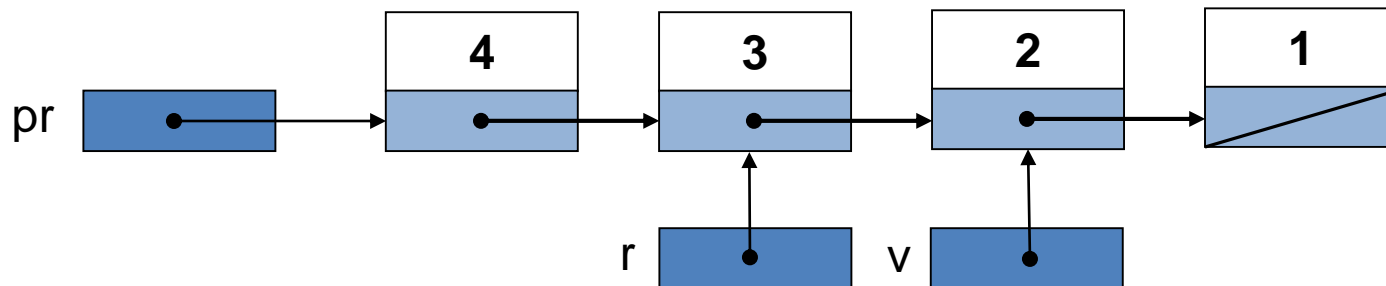
1. Aprašymas, formavimas, peržiūra, veiksmai (šalinimas, įterpimas, paieška).
2. Sąrašo klasė.
3. Sąrašas su fiktyviais elementais.

Prisiminkime

Vienkrypčiame sąrašė galima judėti tik viena kryptimi (link sąrašo pabaigos).

Norint pašalinti vienkrypčio sąrašo elementą, ar įterpti naują elementą prieš turimą, reikia žinoti elemento, esančio prieš jį, adresą.

Elemento, esančio prieš **v**, radimui reikia ciklo.

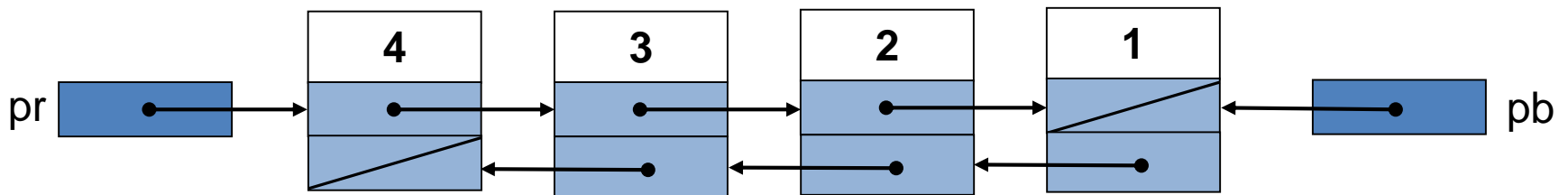


Dvikryptis sąrašas

Nuo vienkrypčio sąrašo skiriasi tuo, kad kiekviename elemente (mazge) yra papildoma nuoroda, rodanti į prieš jį esantį elementą.

Dvikrypčiame sąrašė galima judėti dviem kryptim, t.y. tiek link pabaigos, tiek link pradžios.

Tikslinga saugoti ne tik pirmojo, bet ir paskutiniojo elemento adresą, t.y. reikia turėti dvi nuorodas: sąrašo pradžiai (**pr**) ir sąrašo pabaigai (**pb**).



Elemento (Mazgo) klasė

```
public sealed class Mazgas
```

```
{
```

```
    public int Duomenys { get; set; }
```

```
    public Mazgas Desine { get; set; }
```

```
    public Mazgas Kaire { get; set; }
```

```
    public Mazgas()
```

```
{
```

```
}
```

```
    public Mazgas(int duomenys,  
                  Mazgas adresasD, Mazgas adresasK)
```

```
{
```

```
        this.Duomenys = duomenys;
```

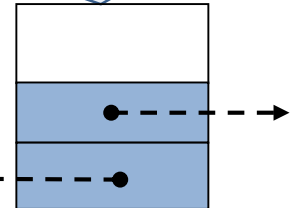
```
        this.Desine = adresasD;
```

```
        this.Kaire = adresasK;
```

```
}
```

```
}
```

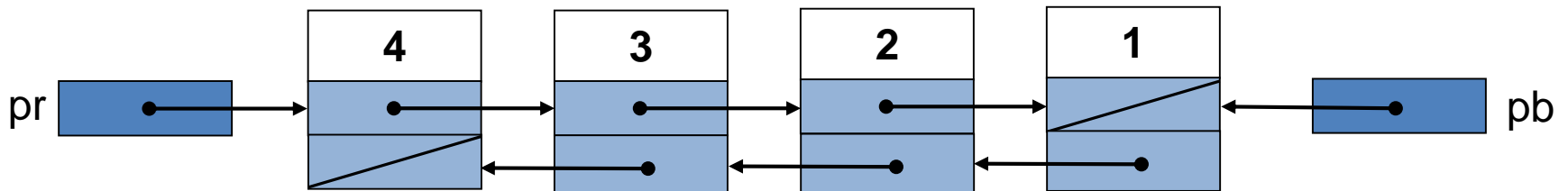
Mazgo (elemento) grafinis vaizdavimas



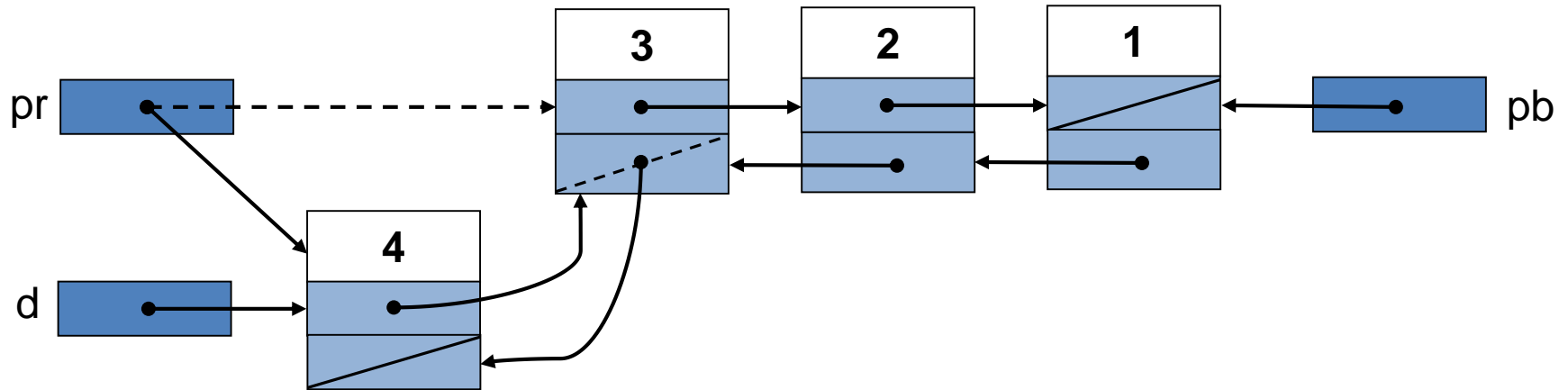
Dvi nuorodos elementų tarpusavio
susiejimui

Veiksmai su sąrašo elementais

- **Veiksmai atliekami kitaip**, nei su vienkrypčiu sąrašu:
 - formavimas,
 - perrinkimas (galimas abiem kryptimis),
 - įterpimas,
 - šalinimas.
- **Veiksmai atliekami taip pat**, kaip ir su vienkrypčiu sąrašu:
 - rikiavimas (apkeičiant tik informacinę dalį),
 - paieška,
 - naikinimas.



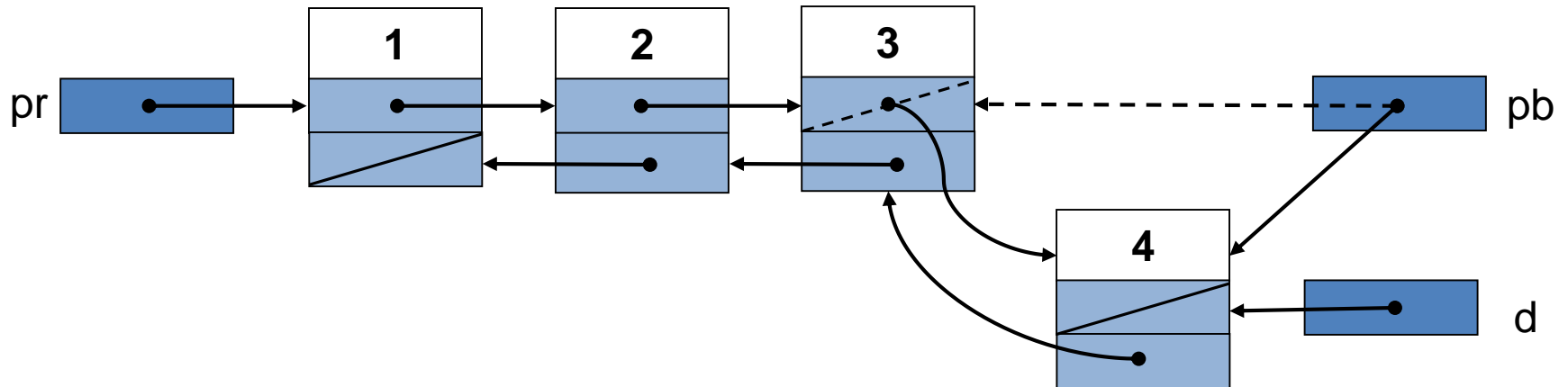
Naujo elemento įterpimas sąrašo pradžioje



```
int skaicius; // įvedamas skaičius
```

```
...
Mazgas d = new Mazgas(skaicius, pr, null);
if (pr != null)
    pr.Kaire = d;
else
    pb = d;
pr = d;
```

Naujo elemento įterpimas sąrašo pabaigoje



```
int skaicius; // įvedamas skaičius
...
Mazgas d = new Mazgas(skaicius, null, pb);
if (pb != null)
    pb.Desine = d;
else
    pr = d;
pb = d;
```

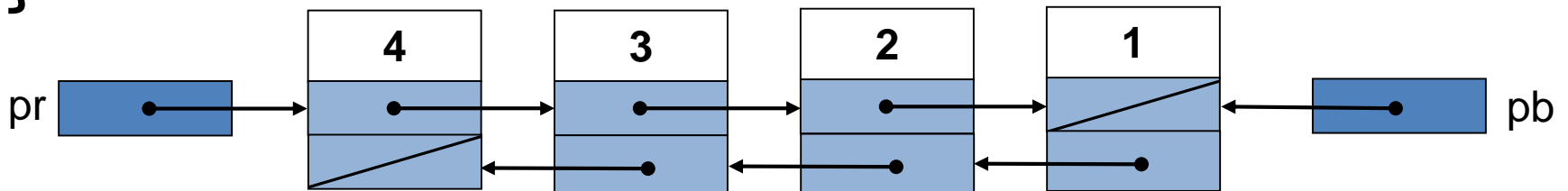

Sąrašo peržiūra 1/2

- Nuo pradžios:

```
for (Mazgas d = pr; d != null; d = d.Desine)  
{  
    // Veiksmas su d.Duomenys  
}
```

- Nuo pabaigos:

```
for (Mazgas d = pb; d != null; d = d.Kaire)  
{  
    // Veiksmas su d.Duomenys  
}
```

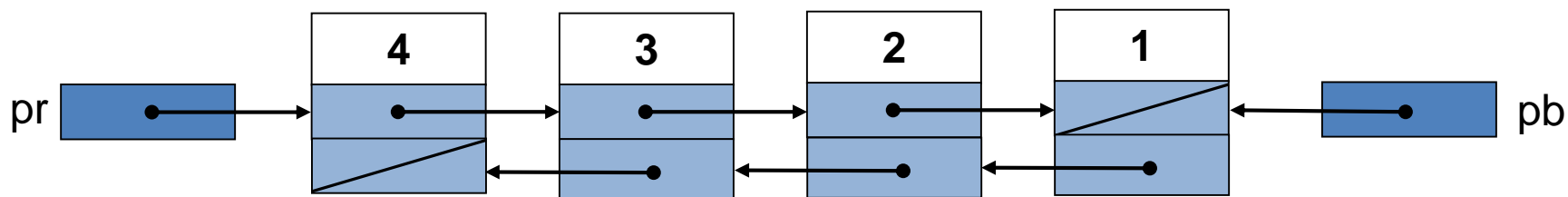


Sąrašo peržiūra 2/2

Duomenys faile:

1 2 3 4

Suformuotas sąrašas:



Rezultatai faile:

Sąrašas nuo pradžios:

4 3 2 1

Sąrašas nuo pabaigos:

1 2 3 4

Elemento įterpimas sąrašo viduje

```
int skaicius = 2; // įvedamas skaičius
```

```
...
```

```
Mazgas r = ...; // elementas, už kurio įterpiama
```

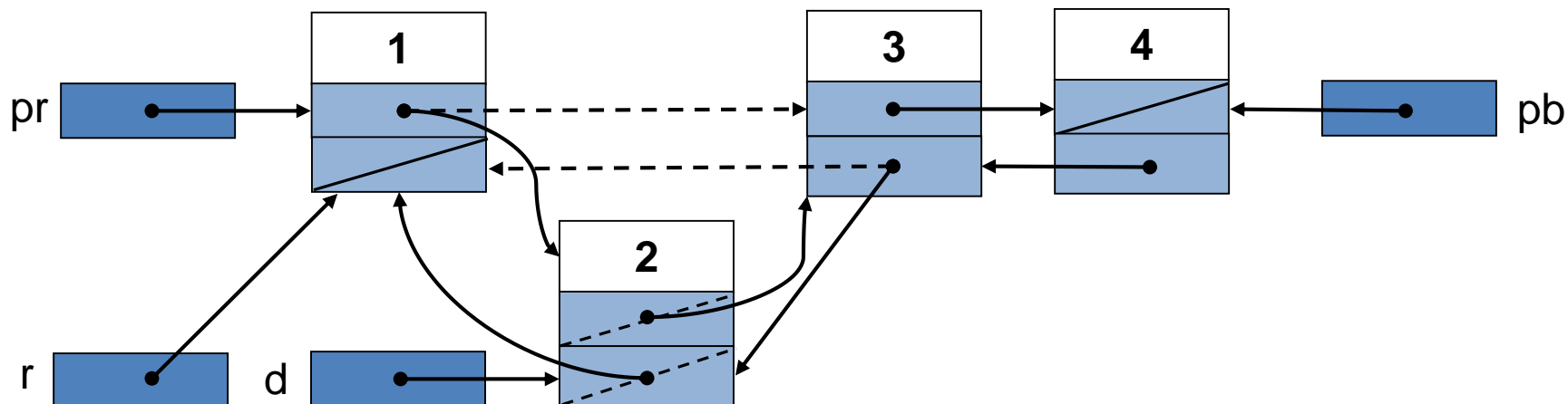
```
Mazgas d = new Mazgas(skaicius, null, null);
```

```
d.Desine = r.Desine;
```

```
d.Kaire = r;
```

```
r.Desine.Kaire = d;
```

```
r.Desine = d;
```



Elemento įterpimas už nurodyto elemento

```
int skaicius = 2; // įvedamas skaičius
```

```
...
```

```
Mazgas r = ...; // elementas, už kurio įterpiama
```

```
Mazgas d = new Mazgas(skaicius, r.Desine, r);
```

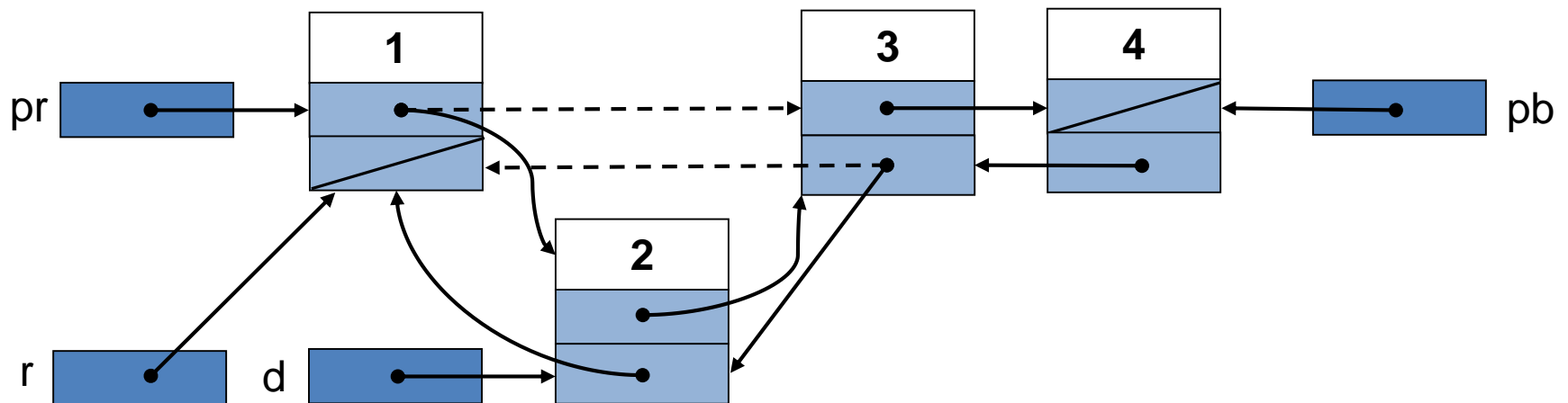
```
if (pb == r)
```

```
    pb = d; // įterpti gale (už paskutinio)
```

```
else
```

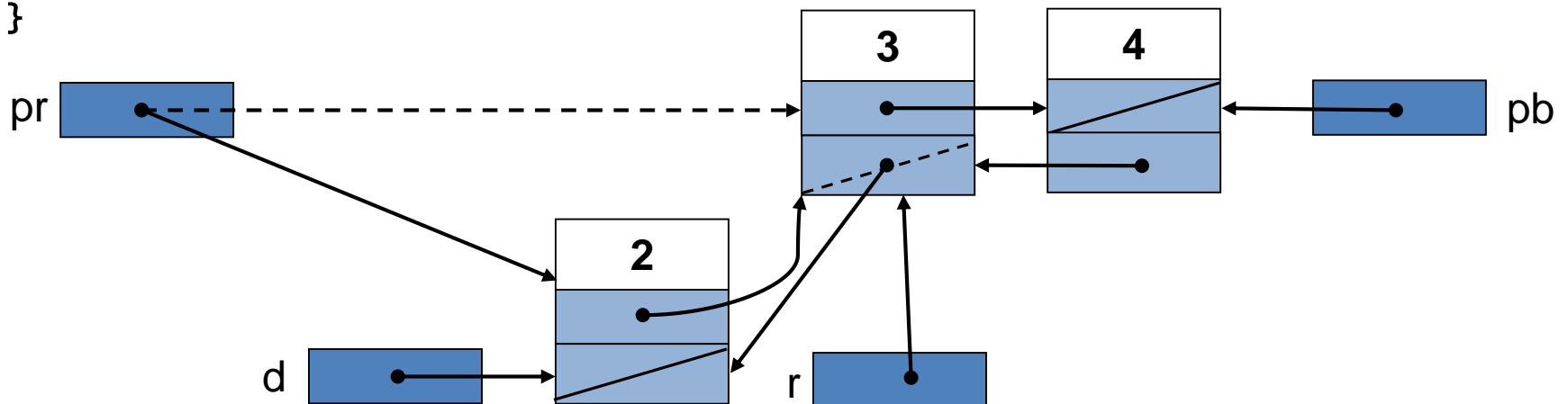
```
    d.Desine.Kaire = d;
```

```
    r.Desine = d;
```



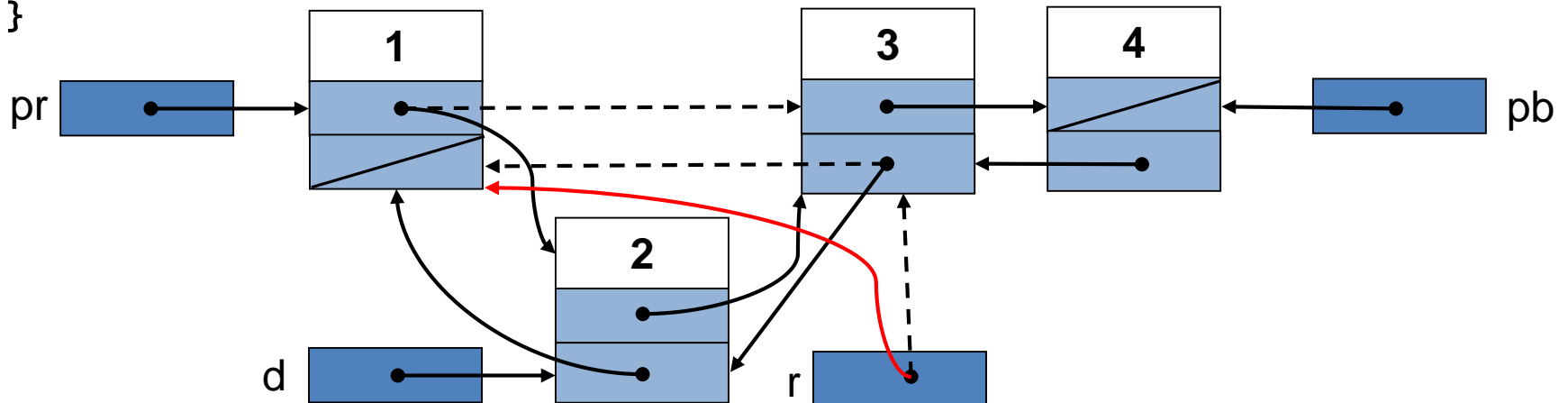
Elemento įterpimas prieš nurodytą elementą 1/2

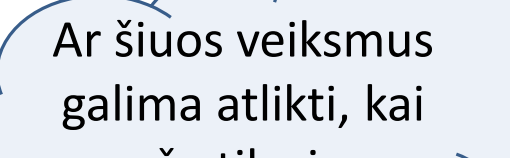
```
int skaicius = 2; // įvedamas skaičius
...
Mazgas r = ...; // elementas, prieš kurį įterpiama
if (r == pr)    // įterpti prieš pirmąjį
{
    Mazgas d = new Mazgas(skaicius, pr, null);
    pr.Kaire = d;
    pr = d;
}
else
{
    r = r.Kaire;
    // kiti veiksmai analogiškai iterpimui UŽ NURODYTO ELEMENTO
}
}
```



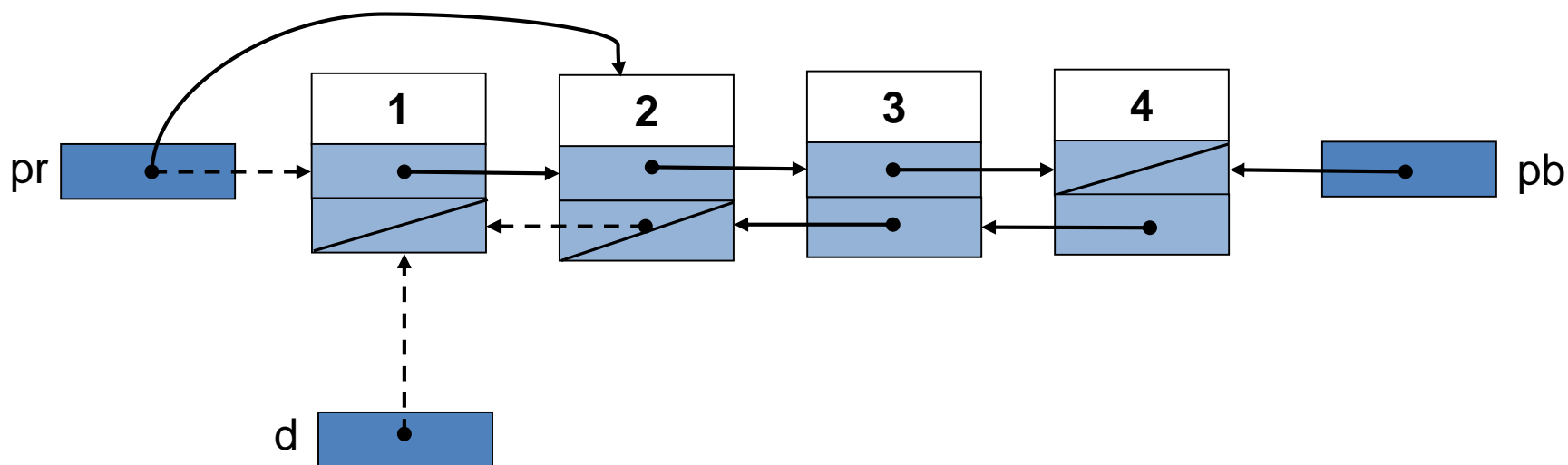
Elemento įterpimas prieš nurodytą elementą 2/2

```
int skaicius = 2; // įvedamas skaičius
...
Mazgas r = ...; // elementas, prieš kurį įterpiama
if (r == pr)    // įterpti prieš pirmąjį
{
    Mazgas d = new Mazgas(skaicius, pr, null);
    pr.Kaire = d;
    pr = d;
}
else
{
    r = r.Kaire; // schemaje pažymėta raudona spalva
    // kiti veiksmai analogiškai įterpimui UŽ NURODYTO ELEMENTO
}
```





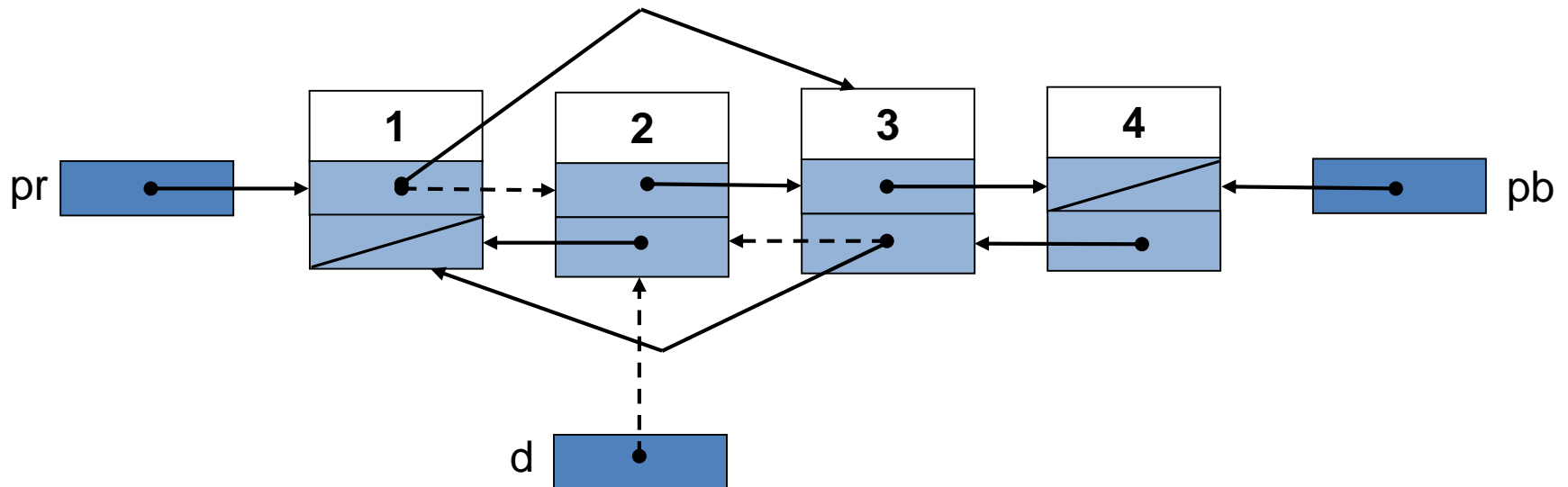
Ar šiuos veiksmus galima atlikti, kai sąraše tik vienas elementas?



Vidurinio elemento šalinimas

```
Mazgas d = ...;  
d.Kaire.Desine = d.Desine;  
d.Desine.Kaire = d.Kaire;  
d = null;
```

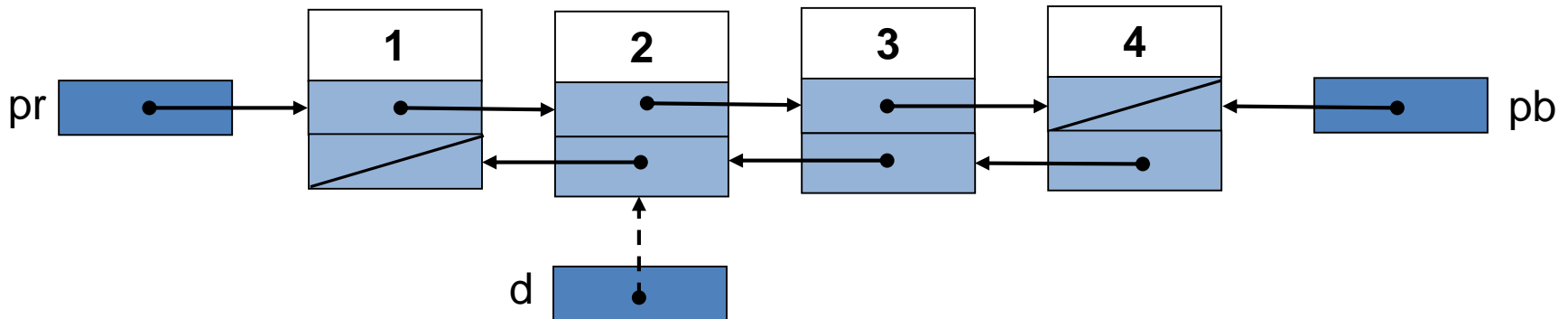
Ar šiuos veiksmus
galima atlikti su
paskutiniu sąrašo
elementu?



Elemento šalinimas (bendras atvejis)

```

Mazgas d = ...; // pirmas, vidurinis, paskutinis
if (d == pr)
    pr = pr.Desine;
if (d == pb)
    pb = pb.Kaire;
if (d.Kaire != null)
    d.Kaire.Desine = d.Desine;
if (d.Desine != null)
    d.Desine.Kaire = d.Kaire;
d = null;
    
```



Sąrašo klasė

```
public sealed class Sąrašas
{
    private Mazgas pr;    // sąrašo pradžia
    private Mazgas pb;    // sąrašo pabaiga
    private Mazgas ss;    // sąrašo sąsaja
    // Konstruktorius: suteikiamos pradinės reikšmės
    public Sąrašas()
    {
        this.pr = null;
        this.pb = null;
        this.ss = null;
    }
    // Sąsajos metodai
    ...
}
```

Sąrašo klasės sąsajos metodai 1/6

```
public sealed class Sąrašas
{
```

```
    ...
```

```
    // Sąsajos metodai
```

```
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PRADŽIOS
```

```
    // skaicius – naujo elemento reikšmė (duomenys)
```

```
    public void DėtiDuomenįA(int skaicius)
```

```
    {
```

```
        Mazgas d = new Mazgas(skaicius, pr, null);
```

```
        if (pr != null)
```

```
            pr.Kaire = d;
```

```
        else
```

```
            pb = d;
```

```
        pr = d;
```

```
    }
```

```
    ...
```

Sąrašo klasės sąsajos metodai 2/6

```
public sealed class Sąrašas
{
    ...
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PABAIGOS
    // skaicius – naujo elemento reikšmė (duomenys)
    public void DėtiDuomenįT(int skaicius)
    {
        Mazgas d = new Mazgas(skaicius, pr, null);
        if (pb != null)
            pb.Desinė = d;
        else
            pr = d;
        pb = d;
    }
    ...
}
```

Sąrašo klasės sąsajos metodai 3/6

```
public sealed class Sąrašas
{
    ...
    // Grąžina sąrašo sąsajos elemento reikšmę
    public int ImtiDuomenis()
    {
        return ss.Duomenys;
    }
    ...
}
```

Sąrašo klasės sąsajos metodai 4/6

```
public sealed class Sąrašas
```

```
{
```

```
    . . .
```

```
    // Sąsajai priskiriama sąrašo pradžia
```

```
    public void Pradžia()
```

```
    {
```

```
        ss = pr;
```

```
    }
```

```
    // Sąsajai priskiriama sąrašo pabaiga
```

```
    public void Pabaiga()
```

```
    {
```

```
        ss = pb;
```

```
    }
```

```
    // Gražina true, jeigu sąsaja netuščia; false - priešingu atveju
```

```
    public bool Yra()
```

```
    {
```

```
        return ss != null;
```

```
    }
```

```
    . . .
```

```
}
```

Sąrašo klasės sąsajos metodai 5/6

```
public sealed class Sąrašas
```

```
{  
    ...  
    // Sąsajai priskiriamas sąrašo elementas, esantis dešiniau  
    public void Desinė()  
    {  
        ss = ss.Desinė;  
    }  
    // Sąsajai priskiriamas sąrašo elementas, esantis kairiau  
    public void Kairen()  
    {  
        ss = ss.Kaire;  
    }  
    ...  
}
```

Sąrašo klasės sąsajos metodai 6/6

```
public sealed class Sąrašas
{
    ...
    // sunaikinamas sąrašas
    public void Naikinti()
    {
        while (pr != null)
        {
            ss = pr;
            pr = pr.Desine;
            ss.Desine = null;
            ss.Kaire = null;
        }
        pb = ss = pr;    // pb = ss = null;
    }
}
```


Sąrašo rikiavimas išrinkimo būdu (Minmax)

```
// Sąrašo rikiavimas MAŽĖJIMO tvarka
```

```
public void Minmax()
```

```
{
    for (Mazgas d1 = pr; d1 != null; d1 = d1.Desine)
    {
```

```
        // Didžiausios reikšmės paieška intervale
```

```
        Mazgas maxv = d1;
```

```
        for (Mazgas d2 = d1; d2 != null; d2 = d2.Desine)
```

```
            if (d2.Duomenys > maxv.Duomenys)
```

```
                maxv = d2;
```

```
        // Duomenų (informacinių) dalių sukeitimas vietomis
```

```
        int k = d1.Duomenys;
```

```
        d1.Duomenys = maxv.Duomenys;
```

```
        maxv.Duomenys = k;
```

```
    }
```

```
}
```

Pastaba: kai sąrašo duomenų dalyje yra objektai, palyginimo sakinyje (**if**) reikia naudoti atitinkamą objektų palyginimo užklotą operatorių.

Sąrašo rikiavimas burbuliuko būdu 1/2

```
public void Burbulas()    // Sąrašo rikiavimas MAŽĖJIMO tvarka
{
    bool keista = true;
    Mazgas d1, d2;
    while (keista)
    {
        keista = false;
        d1 = d2 = pr;
        while (d2 != null)
        {
            if (d2.Duomenys > d1.Duomenys)
            {
                int k = d1.Duomenys;
                d1.Duomenys = d2.Duomenys;
                d2.Duomenys = k;
                keista = true;
            }
            d1 = d2;    d2 = d2.Desine;
        }
    }
}
```

Sąrašo rikiavimas burbuliuko būdu 2/2

```
public void Burbulas() // Sąrašo rikiavimas MAŽĖJIMO tvarka
{
    if (pr == null) { return; }
    bool keista = true;
    while (keista)
    {
        keista = false;
        Mazgas d = pr;
        while (d.Desine != null)
        {
            if (d.Desine.Duomenys > d.Duomenys)
            {
                int k = d.Duomenys;
                d.Duomenys = d.Desine.Duomenys;
                d.Desine.Duomenys = k;
                keista = true;
            }
            d = d.Desine;
        }
    }
}
```

Kodėl reikalingas
šis tikrinimas?

Sąrašo klasės sąrašo sudarymas 1/2

// Skaitomi skaičiai iš failo ir sudedami į sąrašą ATVIRKŠTINE tvarka

// fv – duomenų failo vardas

```
static Sąrašas skaitytiAtv(string fv)
{
    var A = new Sąrašas();
    using (var failas = new StreamReader(fv))
    {
        int skaicius;
        string eilute;
        while ((eilute = failas.ReadLine()) != null)
        {
            skaicius = Convert.ToInt32(eilute);
            A.DėtiDuomenisA(skaicius);
        }
    }
    return A;
}
```

Sąrašo klasės sąrašo sudarymas 2/2

// Skaitomi skaičiai iš failo ir sudedami į sąrašą TIESIOGINE tvarka

// fv – duomenų failo vardas

```
static Sąrašas skaitytiTiesiog(string fv)
{
    var A = new Sąrašas();
    using (var failas = new StreamReader(fv))
    {
        int skaicius;
        string eilute;
        while ((eilute = failas.ReadLine()) != null)
        {
            skaicius = Convert.ToInt32(eilute);
            A.DėtiDuomenįT(skaicius);
        }
    }
    return A;
}
```

Sąrašo klasės sąrašo spausdinimas 1/2

```
// Sąrašo duomenys spausdinami faile nuo PRADŽIOS
// fv – duomenų failo vardas
// A - sąrašo objekto nuoroda
// koment - komentaras
static void Spausdinti1(string fv, Sąrašas A, string koment)
{
    using (var failas = new StreamWriter(fv, true))
    {
        failas.WriteLine(koment);
        // Sąrašo peržiūra, panaudojant sąsajos metodus
        for (A.Pradžia(); A.Yra(); A.Desinen())
        {
            failas.WriteLine("{0, 3:d}", A.ImtiDuomenis());
        }
        failas.WriteLine();
    }
}
```

Sąrašo klasės sąrašo spausdinimas 2/2

```
// Sąrašo duomenys spausdinami faile nuo PABAIGOS
// fv – duomenų failo vardas
// A - sąrašo objekto nuoroda
// koment - komentaras
static void Spausdinti2(string fv, Sąrašas A, string koment)
{
    using (var failas = new StreamWriter(fv, true))
    {
        failas.WriteLine(koment);
        // Sąrašo peržiūra, panaudojant sąsajos metodus
        for (A.Pabaiga(); A.Yra(); A.Kairen())
        {
            failas.WriteLine("{0, 3:d}", A.ImtiDuomenis());
        }
        failas.WriteLine();
    }
}
```

Sąrašo klasės objektas 1/2

```
. . .  
const string CFd = @"..\..\Duomenys.txt";  
const string CFr = @"..\..\Rezultatai.txt";  
  
. . .  
Sąrašas Advikr = SkaitytiAtv(CFd);  
Spausdinti1(CFr, Advikr, "Atvirkštinis dvikryptis sąrašas");  
  
Sąrašas TDvikr = SkaitytiTiesiog(CFd);  
Spausdinti2(CFr, TDvikr, "Tiesioginis dvikryptis sąrašas");  
  
. . .
```


Sąrašo klasės objektas 2/2

Duomenų failas:

1 2 3 4

Rezultatų failas:

Atvirkštinis dvikryptis sąrašas

4

3

2

1

Tiesioginis dvikryptis sąrašas

1

2

3

4

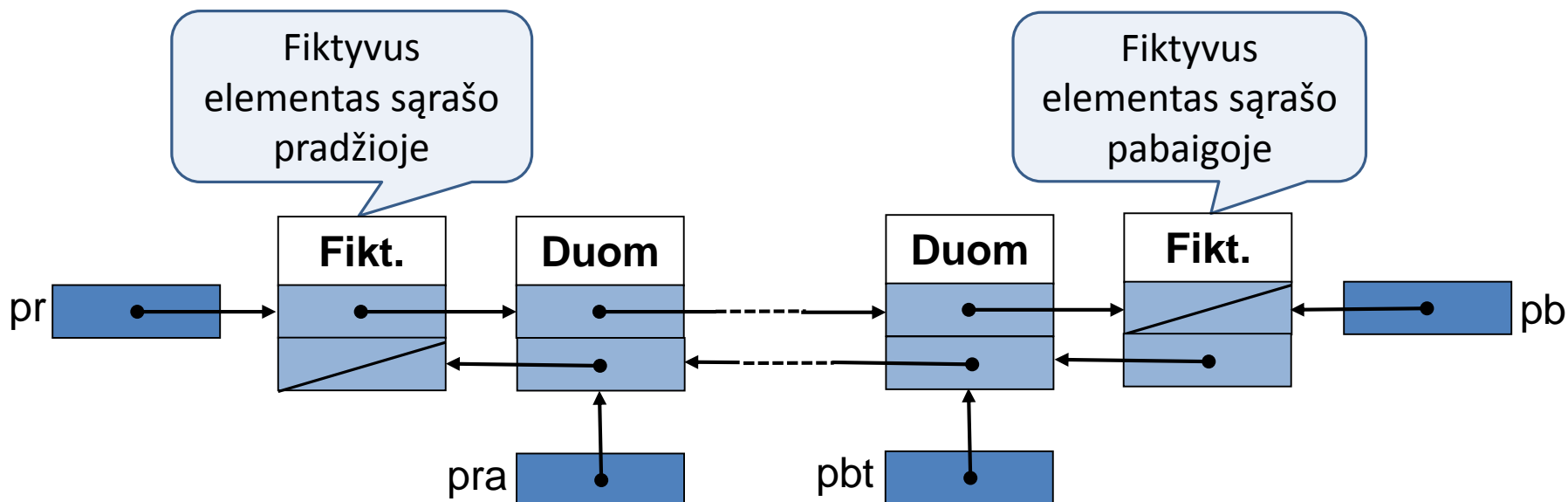
Sąrašas su fiktyviais elementais

Pirmas ir paskutinis elementai nenaudojami informacijai saugoti.

Sąrašas niekada nebus tuščias.

Elementų **įterpimas** bei **šalinimas** vyksta tik sąrašo viduje (mažiau tikrinimų).

Atliekant veiksmus su sąrašo elementais reikia tai įvertinti.



Sąrašo klasė su fiktyviais elementais

```
public sealed class Sąrašas
```

```
{
```

```
    private Mazgas pr;    // sąrašo pradžia
```

```
    private Mazgas pb;    // sąrašo pabaiga
```

```
    private Mazgas pra;    // sąrašo pradžia (papildoma)
```

```
    private Mazgas pbt;    // sąrašo pabaiga (papildoma)
```

```
    private Mazgas ss;    // sąrašo sąsaja
```

```
    // Konstruktorius: sukuriami du fiktyvūs elementai
```

```
    public Sąrašas()
```

```
{
```

```
        this.pr = new Mazgas(Int32.MaxValue, null, null);
```

```
        this.pb = new Mazgas(Int32.MaxValue, null, pr);
```

```
        pra = pb;
```

```
        pr.Desine = pb;
```

```
        pbt = pr;
```

```
        this.ss = null;
```

```
}
```

```
    // Sąsajos metodai
```

```
    ...
```

```
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 1/6

```
public sealed class Sąrašas
{
```

```
    ...
```

```
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PRADŽIOS
```

```
    // skaicius – naujo elemento reikšmė (duomenys)
```

```
    public void DėtiDuomenisA(int skaicius)
```

```
    {
```

```
        pra.Kaire = new Mazgas(skaicius, pra, pr);
```

```
        pra = pra.Kaire;
```

```
        pr.Desine = pra;
```

```
    }
```

```
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PABAIGOS
```

```
    // skaicius – naujo elemento reikšmė (duomenys)
```

```
    public void DėtiDuomenisT(int skaicius)
```

```
    {
```

```
        pbt.Desine = new Mazgas(skaicius, pb, pbt);
```

```
        pbt = pbt.Desine;
```

```
        pb.Kaire = pbt;
```

```
    }
```

```
    ...
```

```
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 2/6

```
public sealed class Sąrašas
{
    ...
    // Grąžina sąrašo sąsajos elemento reikšmę
    public int ImtiDuomenis()
    {
        return ss.Duomenys;
    }
    ...
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 3/6

```
public sealed class Sąrašas
{
    ...
    // sąsajai priskiriama sąrašo pradžia
    public void Pradžia()
    {
        ss = pr.Desine;
    }
    // sąsajai priskiriama sąrašo pabaiga
    public void Pabaiga()
    {
        ss = pb.Kaire;
    }
    ...
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 4/6

```
public sealed class Sąrašas
{
    ...
    // Sąsajai priskiriamas sąrašo elementas, esantis dešiniau
    public void Desinen()
    {
        ss = ss.Desine;
    }
    // Sąsajai priskiriamas sąrašo elementas, esantis kairiau
    public void Kairen()
    {
        ss = ss.Kaire;
    }
    ...
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 5/6

```
public sealed class Sąrašas
```

```
{
```

```
    ...
```

```
    // Gražina true, jeigu sąsaja netuščia einant į DEŠINĘ  
    //           false - priešingu atveju
```

```
    public bool YraD()
```

```
    {
```

```
        return (ss.Desine != null);
```

```
    }
```

```
    // Gražina true, jeigu sąsaja netuščia einant į KAIRE  
    //           false - priešingu atveju
```

```
    public bool YraK()
```

```
    {
```

```
        return (ss.Kaire != null);
```

```
    }
```

```
    ...
```

```
}
```


Sąrašo klasės su fiktyviais elementais sąsajos metodai 6/6

```
public sealed class Sąrašas
{
```

```
    ...
```

```
    // sunaikina sąrašą
```

```
    public void Naikinti()
```

```
    {
```

```
        while (pr != null)
```

```
        {
```

```
            ss = pr;
```

```
            pr = pr.Desine;
```

```
            ss.Desine = null;
```

```
            ss.Kaire = null;
```

```
        }
```

```
        pb = ss = pra = pbt = pr;
```

```
    }
```

```
}
```

Pastaba: jeigu sąrašo elementų duomenų dalyje yra nuoroda į objektą, ten taip pat reikėtų įrašyti reikšmę **null**.

Sąrašo su fiktyviais elementais rikiavimas išrinkimo būdu (Minmax)

```
// Sąrašo rikiavimas MAŽĖJIMO tvarka
```

```
public void Minmax() // Metodo vieta: sąrašo klasė
{
    for (Mazgas d1 = pr.Desine; d1.Desine != null; d1 = d1.Desine)
    {
        // Didžiausios reikšmės paieška intervale
        Mazgas maxv = d1;
        for (Mazgas d2 = d1; d2.Desine != null; d2 = d2.Desine)
            if (d2.Duomenys > maxv.Duomenys)
                maxv = d2;
        // Duomenų (informacinių) dalių sukeitimas vietomis
        int k = d1.Duomenys;
        d1.Duomenys = maxv.Duomenys;
        maxv.Duomenys = k;
    }
}
```

Pastaba: kai sąrašo duomenų dalyje yra objektai, palyginimo sakinyje (**if**) reikia naudoti atitinkamą objektų palyginimo užklotą operatorių.

Sąrašo rikiavimas burbuliuko būdu

```
public void Burbulas()    // Sąrašo rikiavimas MAŽĖJIMO tvarka
{
    bool keista = true;
    Mazgas d1, d2;
    while (keista)
    {
        keista = false;
        d1 = d2 = pr.Desine;
        while (d2.Desine != null)
        {
            if (d2.Duomenys > d1.Duomenys)
            {
                int k = d1.Duomenys;
                d1.Duomenys = d2.Duomenys;
                d2.Duomenys = k;
                keista = true;
            }
            d1 = d2;    d2 = d2.Desine;
        }
    }
}
```

Naujo sąrašo formavimas

```
// Iš sąrašo A suformuoja sąrašą B
// pozymis - duomenų atrinkimo požymis
static void Formuoti(Sąrašas A, Sąrašas B, int pozymis)
{
    for (A.Pradžia(); A.YraD(); A.Desinen())
    {
        if (A.ImtiDuomenis() > pozymis)
            B.DėtiDuomenisT(A.ImtiDuomenis());
    }
}
```

Šalinimo metodas

// Šalinamas sąsajos nuorodos (ss) rodomas elementas

// Sąsajos nuoroda "perkeliamo" į dešinę

```
public void šalinti()
```

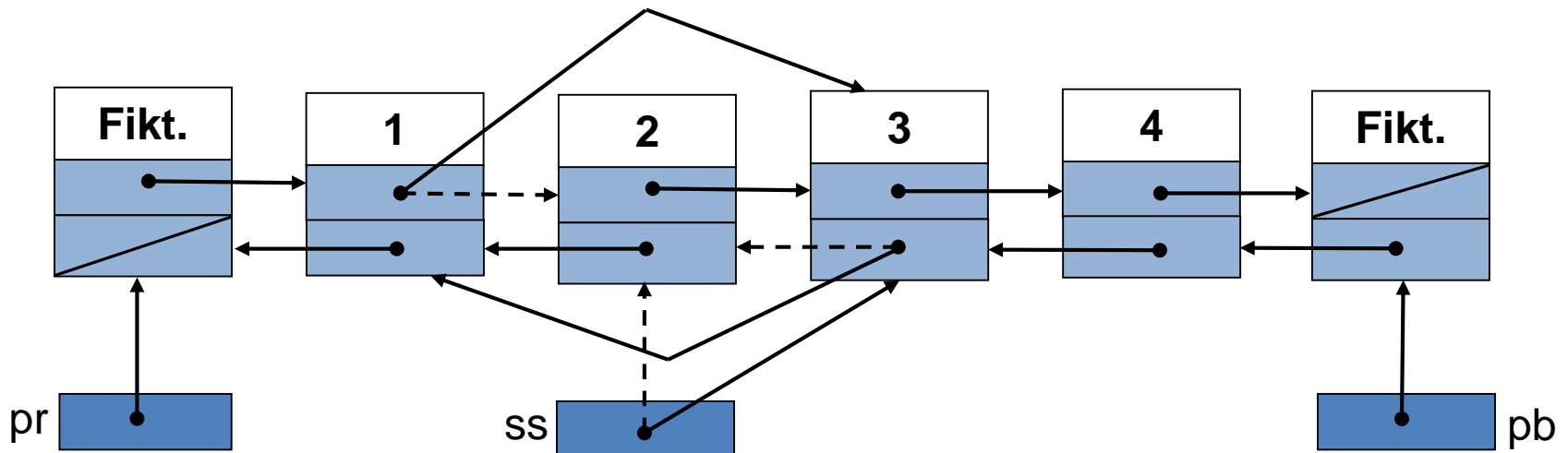
```
{
```

```
    ss.Desine.Kaire = ss.Kaire;
```

```
    ss.Kaire.Desine = ss.Desine;
```

```
    ss = ss.Desine;
```

```
}
```



Įterpimo metodas

// Įterpiamas naujas elementas už sąsajos nuorodos (ss) rodomo elemento

// Sąsajos nuoroda "perkeliamą" į dešinę, ties įterptu elementu

```
public void Įterpti(int k)
```

```
{
```

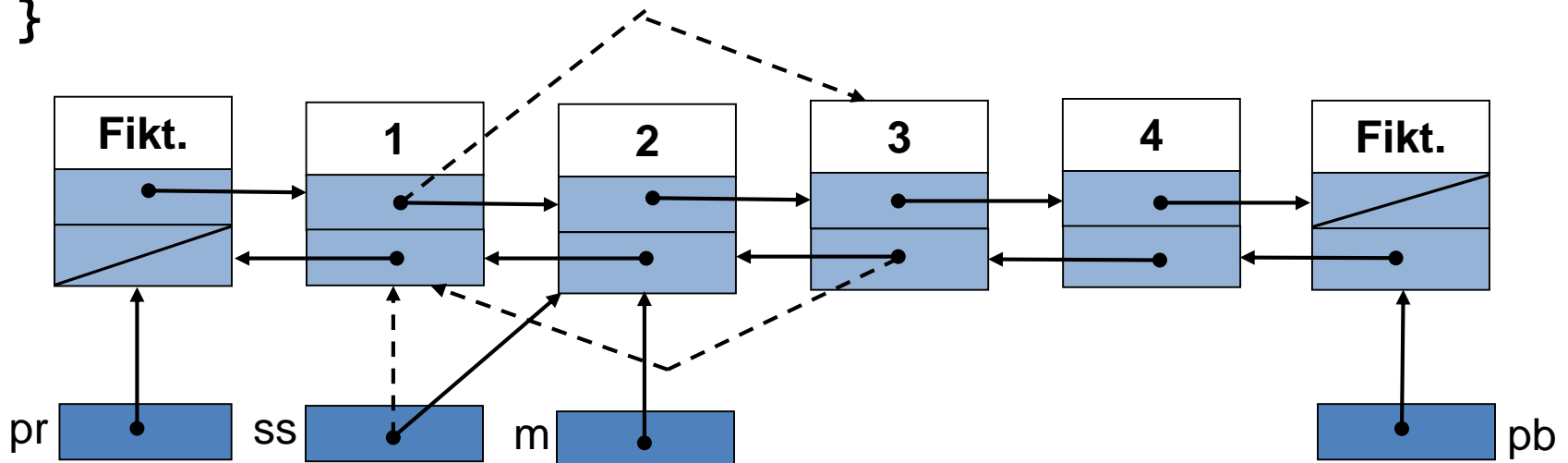
```
    Mazgas m = new Mazgas(k, ss.Desine, ss);
```

```
    ss.Desine = m;
```

```
    m.Desine.Kaire = m;
```

```
    ss = m;
```

```
}
```



Sąrašo klasės objektas 1/4

```
. . .  
const string CFd = @"..\..\Duomenys.txt";  
const string CFr = @"..\..\Rezultatai.txt";  
  
. . .  
Sąrašas AFikt = skaitytiAtv(CFd);  
Spausdinti1(CFr, AFikt, "Atvirkštinis sąrašas");  
Spausdinti2(CFr, AFikt, "Tiesioginis sąrašas");  
. . .
```

Sąrašo klasės objektas 2/4

```
. . .  
AFikt.Pradžia();  
AFikt.Desinen();  
//AFikt.Pabaiga();  
AFikt.Šalinti();  
Spausdinti1(CFr, AFikt, "Atvirkštinis sąrašas (po pašalinimo)");  
Spausdinti2(CFr, AFikt, "Tiesioginis sąrašas (po pašalinimo)");  
  
//AFikt.Pradžia();  
AFikt.Pabaiga();  
AFikt.Įterpti(11);  
AFikt.Įterpti(22);  
Spausdinti1(CFr, AFikt, "Atvirkštinis sąrašas (po įterpimo)");  
Spausdinti2(CFr, AFikt, "Tiesioginis sąrašas (po įterpimo)");  
. . .
```


Sąrašo klasės objektas 3/4

Duomenų failas:

1 2 3 4

Rezultatų failas:

Atvirkštinis sąrašas

4

3

2

1

Tiesioginis sąrašas

1

2

3

4

. . . (kitoje skaidrėje)

Sąrašo klasės objektas 4/4

Duomenų failas:

1 2 3 4

Rezultatų failas:

```
. . .  
Atvirkštinis sąrašas (po pašalinimo)  
4  
2  
1  
Tiesioginis sąrašas (po pašalinimo)  
1  
2  
4  
Atvirkštinis sąrašas (po įterpimo)  
4  
2  
1  
11  
22  
Tiesioginis sąrašas (po įterpimo)  
22  
11  
1  
2  
4
```

Apibendrinimas: sąrašų klasių panašumai ir skirtumai

Kuo **panašios** ir kuo **skiriasi** sąrašų klasės **su fiktyviais** elementais ir **be fiktyvių** elementų?

Skiriasi:

naujo elemento sukūrimo ir prijungimo metodais,
sąsajos pradinųjų reikšmių (adresų) nustatymo metodais,
sąsajos patikros (palyginimo su **null**) metodais.

Nesiskiria:

duomenų paėmimo metodais,
sąsajos “judėjimo” pirmyn ir atgal metodais.

Sudarant metodus, kurie naudoja sąsajos metodus, reikia įvertinti ar sąraše yra fiktyvūs elementai, ar jų nėra?



Pavyzdys

Užduotis

Tekstiniame faile yra duomenys apie realaus pasaulio objektus (pvz. prekes, gyventojus, gatves, planetas ir pan.):

Pavadinimas (eilutė), kiekis (sveikas skaičius)

Pvz., turime duomenis apie studentų programavimo kontrolinio darbo įvertinimus.

Reikia sudaryti dvikryptį objektų sąrašą.

Sąrašą atspausdinti faile lentele.

Pašalinti iš sąrašo studentus, kurių įvertinimas ≥ 5 , t.y. palikti sąraše tuos studentus, kurie antrą kartą galės perrašyti kontrolinį darbą.

Sąrašą surikiuoti išrinkimo metodu pagal du požymius: pažymius ir pavardes.

Pavyzdys 1/14

```
// klasė studento duomenims saugoti
public class Studentas
{
    public string pavVrd { get; set; } // savybė: studento pavardė ir vardas
    public int pazym { get; set; } // savybė: pažymys (įvertinimas)
    //Konstruktorius su numatytosiomis reikšmėmis
    public Studentas(string pavv = "", int pazym = 0)
    {
        this.pavVrd = pavv;
        this.pazym = pazym;
    }
    // užklotas metodas ToString()
    public override string ToString()
    {
        string eilute;
        eilute = string.Format("{0, -20} {1, 2}", pavVrd, pazym);
        return eilute;
    }
    . . .
}
```

Pavyzdys 2/14

```
public class Studentas
{
    . . .
    public static bool operator >=(Studentas stud1, Studentas stud2)
    {
        int poz = String.Compare(stud1.pavVrd, stud2.pavVrd,
                                StringComparison.CurrentCulture);
        return (stud1.pazym > stud2.pazym) ||
            ((stud1.pazym == stud2.pazym) && (poz < 0));
    }
    // -----
    public static bool operator <=(Studentas stud1, Studentas stud2)
    {
        int poz = String.Compare(stud1.pavVrd, stud2.pavVrd,
                                StringComparison.CurrentCulture);
        return (stud1.pazym < stud2.pazym) ||
            ((stud1.pazym == stud2.pazym) && (poz > 0));
    }
    . . .
}
```

```
// Mazgo klasė studento duomenims saugoti
public sealed class Mazgas
{
    public Studentas Duomenys { get; set; }
    public Mazgas Desine { get; set; }
    public Mazgas Kaire { get; set; }
    public Mazgas()
    {
    }
    public Mazgas(Studentas duomenys, Mazgas adresasD, Mazgas adresask)
    {
        this.Duomenys = duomenys;
        this.Desine = adresasD;
        this.Kaire = adresask;
    }
}
```


Pavyzdys 4/14

// sąrašo klasė studentų duomenims saugoti (SU FIKTYVIAIS ELEMENTAIS)

```
public sealed class Sąrašas
```

```
{
```

```
    private Mazgas pr;    // sąrašo pradžia
```

```
    private Mazgas pb;    // sąrašo pabaiga
```

```
    private Mazgas pra;    // sąrašo pradžia (papildoma ATVIRKŠTINIAM)
```

```
    private Mazgas pbt;    // sąrašo pabaiga (papildoma TIESIOGINIAM)
```

```
    private Mazgas ss;    // sąrašo sąsaja
```

```
    // konstruktorius
```

```
    public Sąrašas()
```

```
{
```

```
        this.pr = new Mazgas(new Studentas(), null, null);
```

```
        this.pb = new Mazgas(new Studentas(), null, pr);
```

```
        pra = pb;
```

```
        pr.Desine = pb;
```

```
        pbt = pr;
```

```
        this.ss = null;
```

```
    }
```

```
    . . .
```

```
}
```

Pavyzdys 5/14

. . .

// Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PRADŽIOS

// studentas – naujo elemento reikšmė (duomenys)

public void DėtiDuomenisA(Studentas studentas)

```
{  
    pra.Kaire = new Mazgas(studentas, pra, pr);  
    pra = pra.Kaire;  
    pr.Desine = pra;  
}
```

// Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PABAIGOS

// studentas – naujo elemento reikšmė (duomenys)

public void DėtiDuomenisT(Studentas studentas)

```
{  
    pbt.Desine = new Mazgas(studentas, pb, pbt);  
    pbt = pbt.Desine;  
    pb.Kaire = pbt;  
}
```

. . .

Pavyzdys 6/14

```
. . .  
// Gražina sąrašo sąsajos elemento reikšmę  
public Studentas ImtiDuomenis()  
{  
    return ss.Duomenys;  
}  
// Sąsajai priskiriama sąrašo pradžia  
public void Pradžia()  
{  
    ss = pr.Desine;  
}  
// Sąsajai priskiriama sąrašo pabaiga  
public void Pabaiga()  
{  
    ss = pb.Kaire;  
}  
. . .
```

Pavyzdys 7/14

```
. . .  
// Sąsajai priskiriamas sąrašo elementas, esantis dešiniau  
public void Desinen()  
{  
    ss = ss.Desine;  
}  
// Sąsajai priskiriamas sąrašo elementas, esantis kairiau  
public void Kairen()  
{  
    ss = ss.Kaire;  
}  
// Gražina true, jeigu sąsaja netuščia, einant į DEŠINĘ  
// false - priešingu atveju  
public bool YraD()  
{  
    return (ss.Desine != null);  
}  
// Gražina true, jeigu sąsaja netuščia, einant į KAIRĘ  
// false - priešingu atveju  
public bool YraK()  
{  
    return (ss.Kaire != null);  
}  
. . .
```

Pavyzdys 8/14

```
. . .  
// sunaikina sąrašą  
public void Naikinti()  
{  
    while (pr != null)  
    {  
        ss = pr;  
        pr = pr.Desine;  
        ss.Duomenys = null;  
        ss.Desine = null;  
        ss.Kaire = null;  
    }  
    pb = ss = pra = pbt = pr;  
}  
. . .
```

Pavyzdys 9/14

. . .

// Rikiavimas mažėjimo tvarka

public void Minmax()

{

for (Mazgas d1 = pr.Desine; d1 != null; d1 = d1.Desine)

{

// Didžiausios reikšmės paieška intervale

Mazgas maxv = d1;

for (Mazgas d2 = d1; d2.Desine != null; d2 = d2.Desine)

if (d2.Duomenys >= maxv.Duomenys)

maxv = d2;

// Duomenų (informacinių) dalių sukeitimas vietomis

Studentas stud = d1.Duomenys;

d1.Duomenys = maxv.Duomenys;

maxv.Duomenys = stud;

}

}

. . .

```
. . .  
// Šalinamas sąsajos nuorodos rodomas elementas  
// Sąsajos nuoroda "perkeliamas" į dešinę  
public void Šalinti()  
{  
    ss.Desine.Kaire = ss.Kaire;  
    ss.Kaire.Desine = ss.Desine;  
    ss = ss.Desine;  
}  
//-----  
// Sukuriamas ir įterpiamas naujas elementas už sąsajos nuorodos rodomo elemento  
// Sąsajos nuoroda "perkeliamas" į dešinę, ties įterptu elementu  
public void Įterpti(Studentas stud)  
{  
    Mazgas m = new Mazgas(stud, ss.Desine, ss);  
    ss.Desine = m;  
    m.Desine.Kaire = m;  
    ss = m;  
}  
}
```

Pavyzdys 11/14

// Iš sąrašo A pašalina elementus pagal nurodytą požymį

// pozymis - pašalinimo požymis

```
static void Šalinti(Sąrašas A, int pozymis)
```

```
{
```

```
    for (A.Pradžia(); A.YraD(); )
```

```
    {
```

```
        if (A.ImtiDuomenis().pazym >= pozymis)
```

```
            A.Šalinti();
```

```
        else
```

```
            A.Desinen();
```

```
    }
```

```
}
```


Pavyzdys 12/14

```
. . .  
const string CFd = @"..\..\Studentai.txt";  
const string CFr = @"..\..\Rezultatai.txt";  
.  
.  
.  
Sąrašas StudentaiFiktA = SkaitytiAtv(CFd);  
Spausdinti1(CFr, StudentaiFiktA, "Studentų atvirkštinis sąrašas");  
Spausdinti2(CFr, StudentaiFiktA, "Studentų tiesioginis sąrašas");  
Šalinti(StudentaiFiktA, 5);  
StudentaiFiktA.Pradžia();  
if (!StudentaiFiktA.YraD())  
    Console.WriteLine("Visi studentai teigiami");  
else  
{  
    StudentaiFiktA.Minmax();  
    Spausdinti1(CFr, StudentaiFiktA, "Studentų sąrašas (neigiami)");  
    Spausdinti2(CFr, StudentaiFiktA, "Studentų sąrašas (Neigiami)");  
}  
StudentaiFiktA.Naikinti();  
.  
.  
.
```

Pavyzdys 13/14

Jonaitis Jonas;	3;
Petraitis Petras;	7;
Antanaitis Antanas;	10;
Giedraitis Giedrius;	4;
Onaitytė Ona;	8;
Juozaitis Juozas;	4;
Ramunaitė Ramunė;	2;

Studentų atvirkštinis sąrašas

Pavardė ir vardas	Pažymys
-------------------	---------

Ramunaitė Ramunė	2
Juozaitis Juozas	4
Onaitytė Ona	8
Giedraitis Giedrius	4
Antanaitis Antanas	10
Petraitis Petras	7
Jonaitis Jonas	3

Studentų tiesioginis sąrašas

Pavardė ir vardas	Pažymys
-------------------	---------

Jonaitis Jonas	3
Petraitis Petras	7
Antanaitis Antanas	10
Giedraitis Giedrius	4
Onaitytė Ona	8
Juozaitis Juozas	4
Ramunaitė Ramunė	2

. . .

Pavyzdys 14/14

Jonaitis Jonas;	3;
Petraitis Petras;	7;
Antanaitis Antanas;	10;
Giedraitis Giedrius;	4;
Onaitytė Ona;	8;
Juozaitis Juozas;	4;
Ramunaitė Ramunė;	2;

. . .

Studentų sąrašas (Neigiami)

Pavardė ir vardas	Pažymys
Giedraitis Giedrius	4
Juozaitis Juozas	4
Jonaitis Jonas	3
Ramunaitė Ramunė	2

Studentų sąrašas (Neigiami)

Pavardė ir vardas	Pažymys
Ramunaitė Ramunė	2
Jonaitis Jonas	3
Juozaitis Juozas	4
Giedraitis Giedrius	4

Šioje temoje susipažinote:

1. Dvikrypčio sąrašo aprašymu, formavimu, peržiūra, veiksmams (šalinimu, įterpimu, paieška).
2. Dvikrypčio sąrašo klase.
3. Dvikrypčiu sąrašu su fiktyviais elementais.



Klausimai?