ktu informatikos fakultetas 1922

T07. Susietasis sąrašas. Veiksmai

2 ak. val.

ktu informatikos fakultetas 1922

Temos klausimai

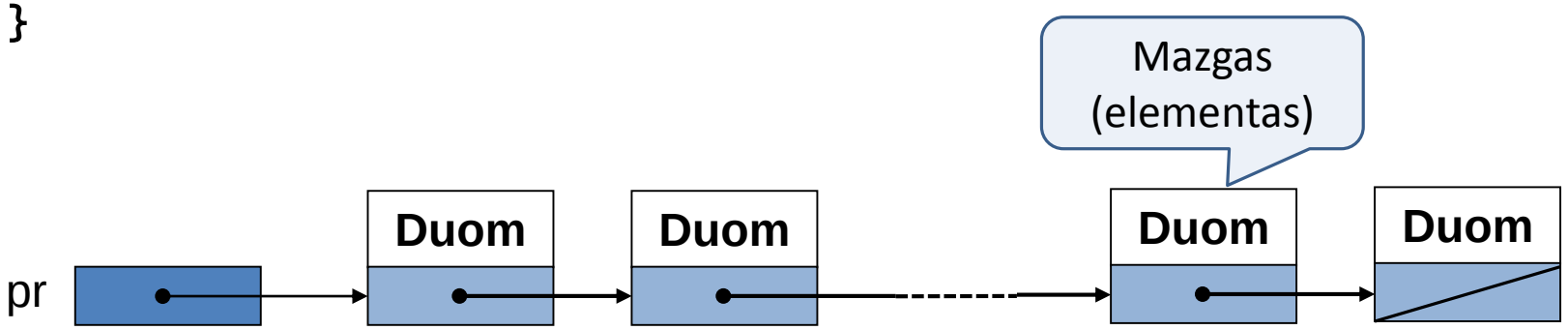
1. Reikšmių sukeitimas
2. Elementų naikinimas
3. Sąrašo naikinimas
4. Elemento įterpimas
5. Sąrašo rikiavimas
6. Sąrašas su fiktyviais elementais

Veiksmai su vienkrypčiu sąrašu

- formavimas,
- peržiūra,
- elementų šalinimas,
- elementų įterpimas,
- paieška,
- rikiavimas.

Sąrašo mazgo (elemento) klasė

```
public sealed class Mazgas
{
    public int Duomenys { get; private set; }
    public Mazgas Kitas { get; set; }
    public Mazgas() { }
    public Mazgas(int duomenys, Mazgas adresas)
    {
        this.Duomenys = duomenys;
        this.Kitas = adresas;
    }
}
```

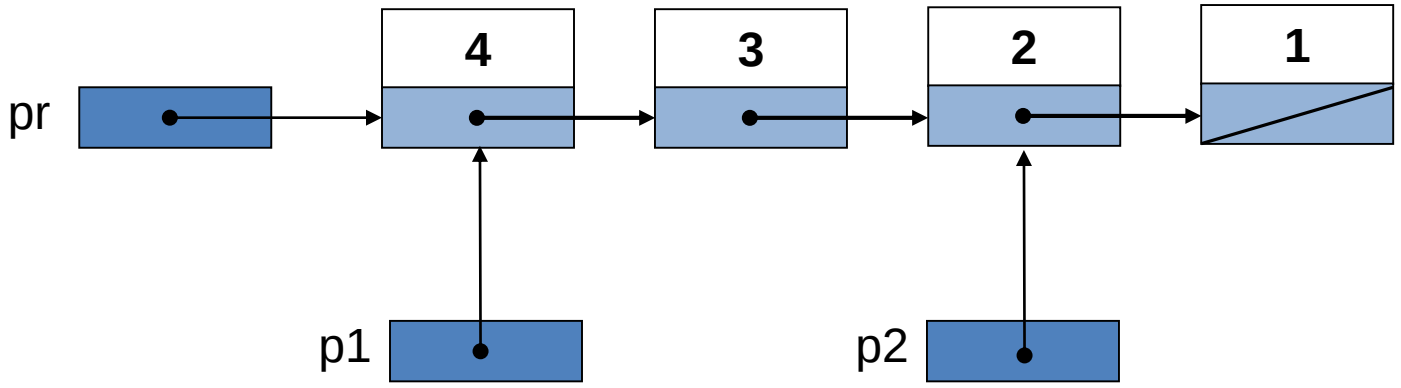


Dviejų elementų (mazgų) duomenų sukeitimas vietomis

Galimi variantai:

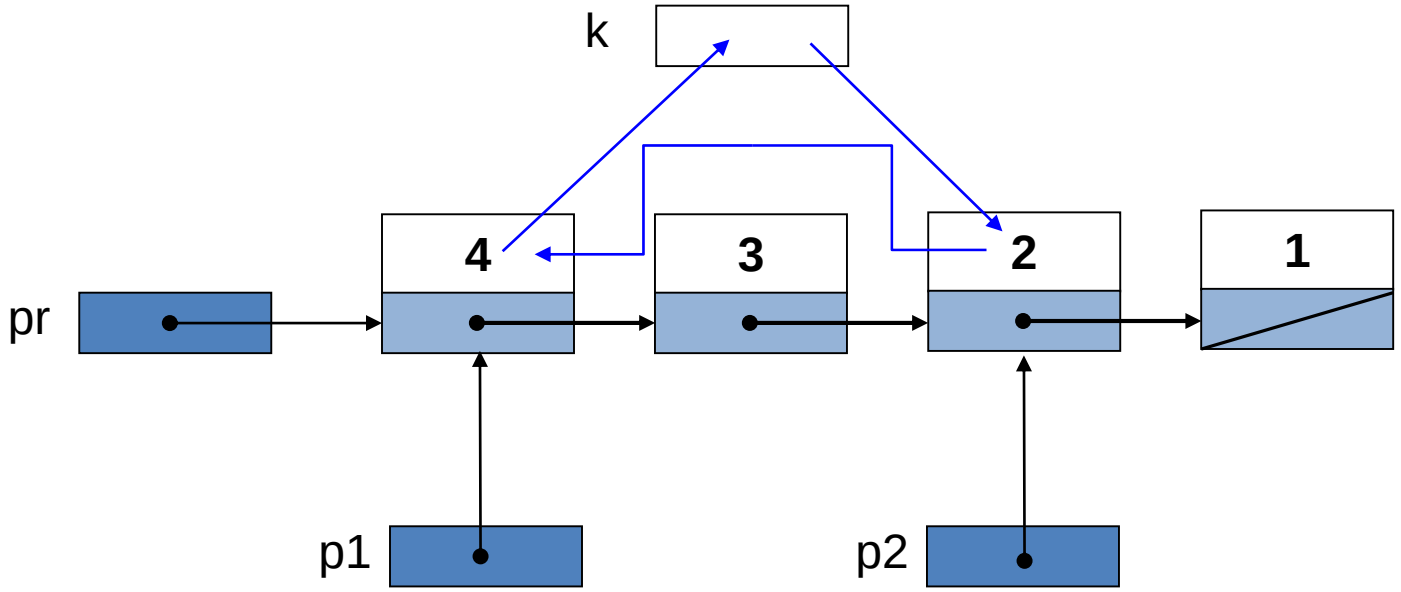
sukeisti informacines (duomenų) dalis (*lengva*);

„perkabinti” rodykles (*sudėtinga*).



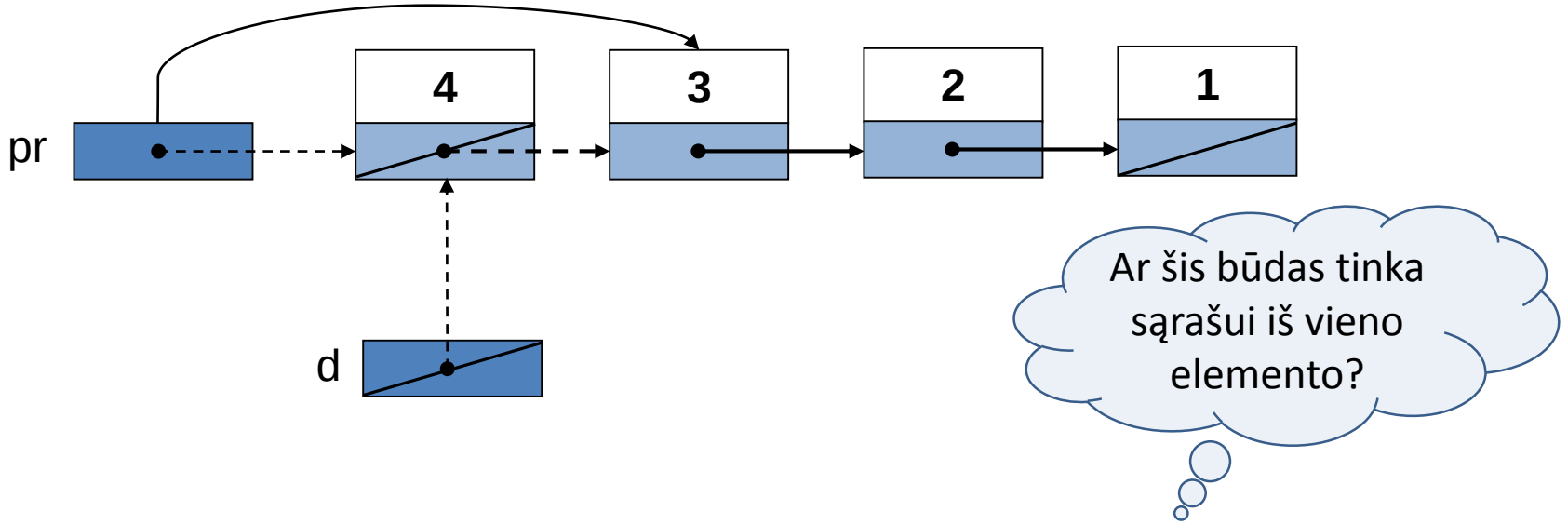
Dviejų mazgų (elementų) duomenų sukeitimas vietomis

```
// Jeigu duomenims nėra nurodyta private set!
int k = p1.Duomenys; // pagalbinis kintamasis
p1.Duomenys = p2.Duomenys;
p2.Duomenys = k;
```



Pirmo elemento šalinimas

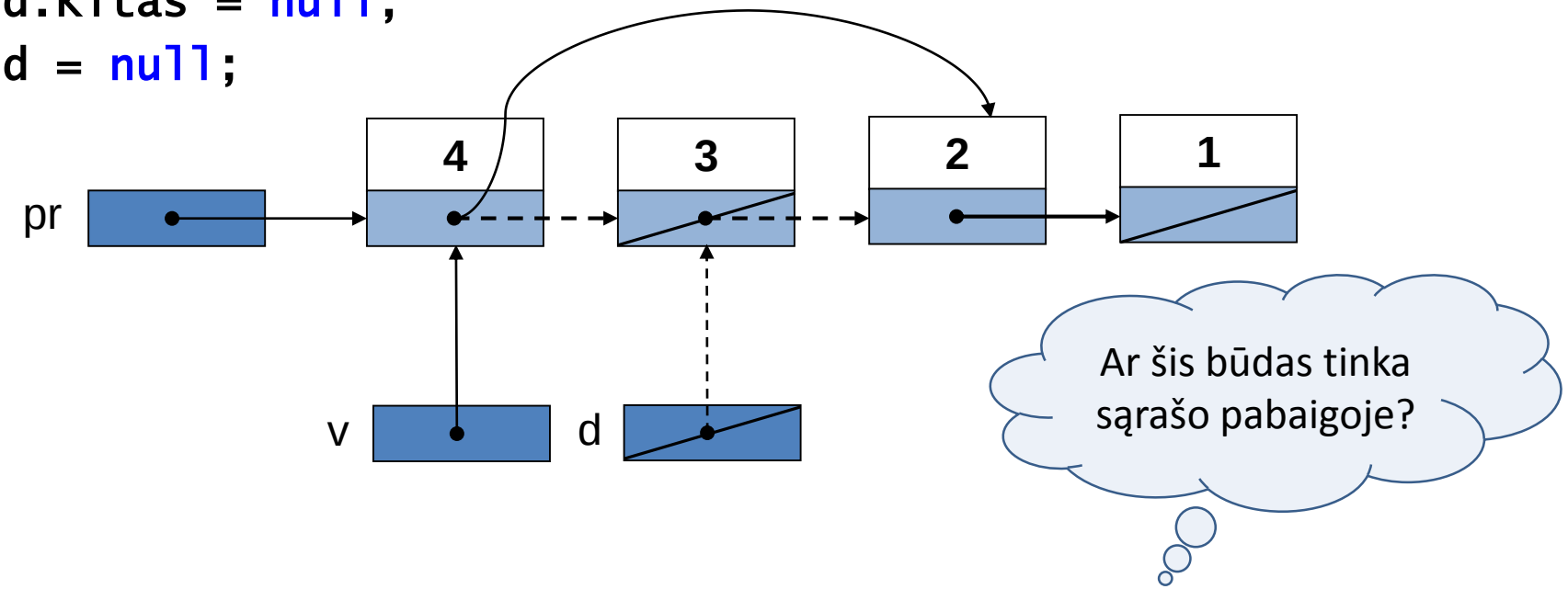
```
Mazgas d = pr;
pr = pr.Kitas;
d.Kitas = null;
d = null;
```



Vidinio elemento šalinimas

Randamas šalinamo elemento adresas **d** ir prieš jį esančio elemento adresas **v**; po to:

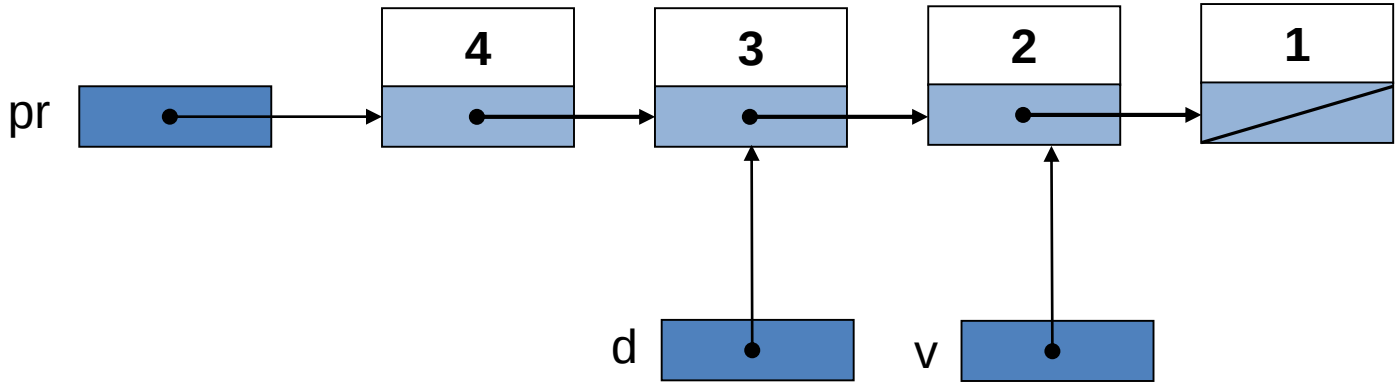
```
Mazgas d; // šalinamas elementas
Mazgas v; // elementas, esantis prieš šalinamąjį
...
v.Kitas = d.Kitas;
d.Kitas = null;
d = null;
```



Elemento šalinimas (kitas būdas) 1/3

Randamas šalinamo elemento adresas **d**, po to už jo esančio elemento adresas **v**:

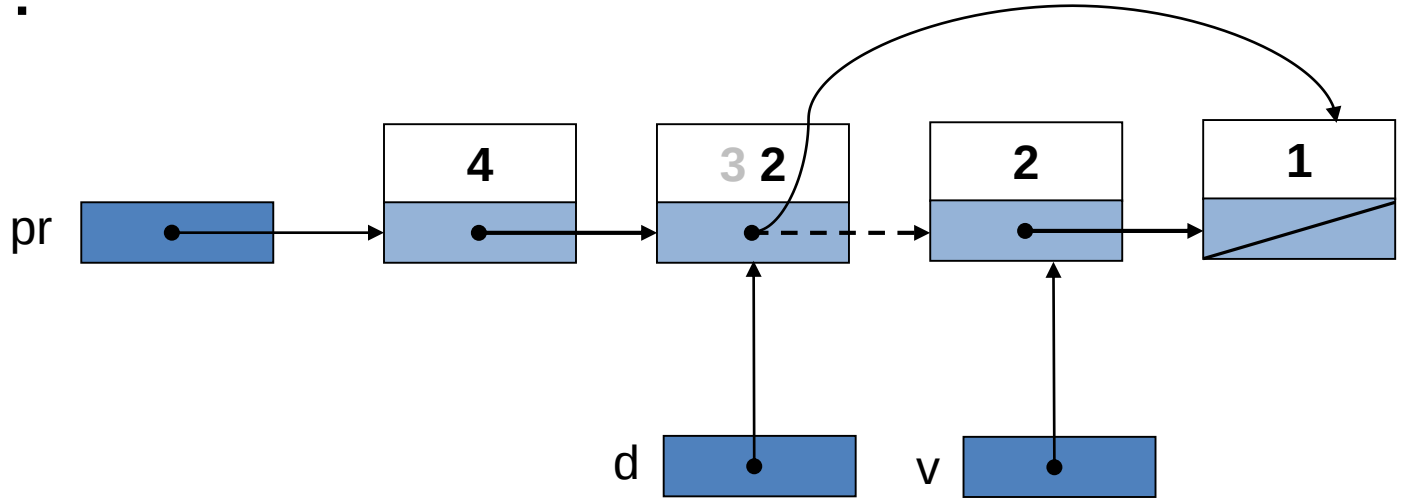
```
Mazgas d;  // šalinamas elementas
Mazgas v;  // elementas, esantis už šalinamojo
...
v = d.Kitas;
...
```



Elemento šalinimas (kitas būdas) 2/3

Į šalinamą elementą **d** perrašyti elemento **v** duomenis ir nuorodos reikšmę:

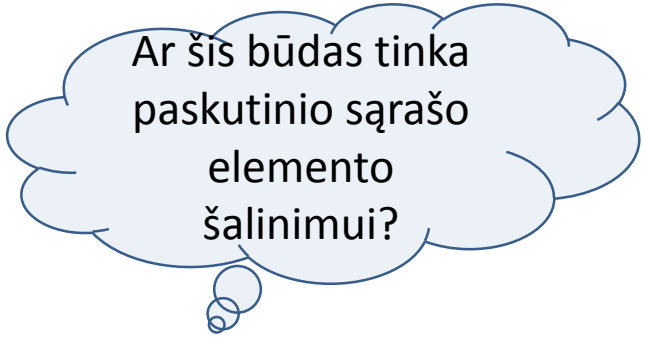
```
...
d.Duomenys = v.Duomenys;
d.Kitas = v.Kitas;
...
```

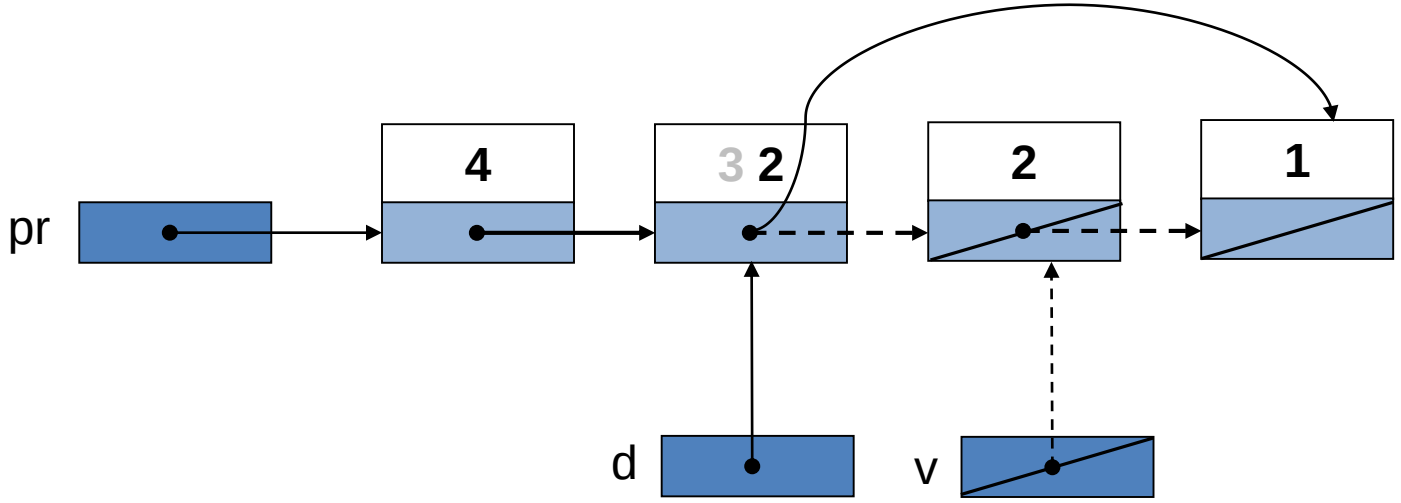


Elemento šalinimas (kitas būdas) 3/3

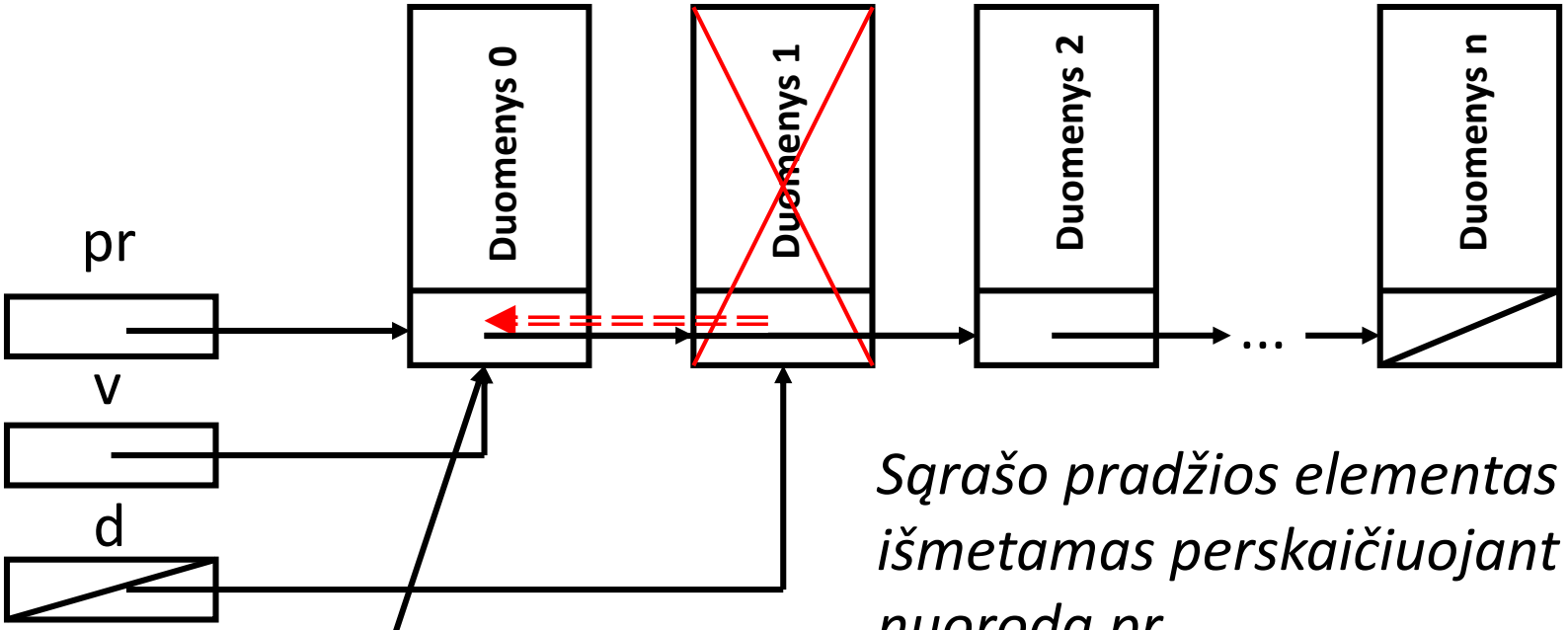
Elemento v „atjungimas“:

```
...
v.Kitas = null;
v = null;
```





Elemento pašalinimas iš sąrašo (su animacija)



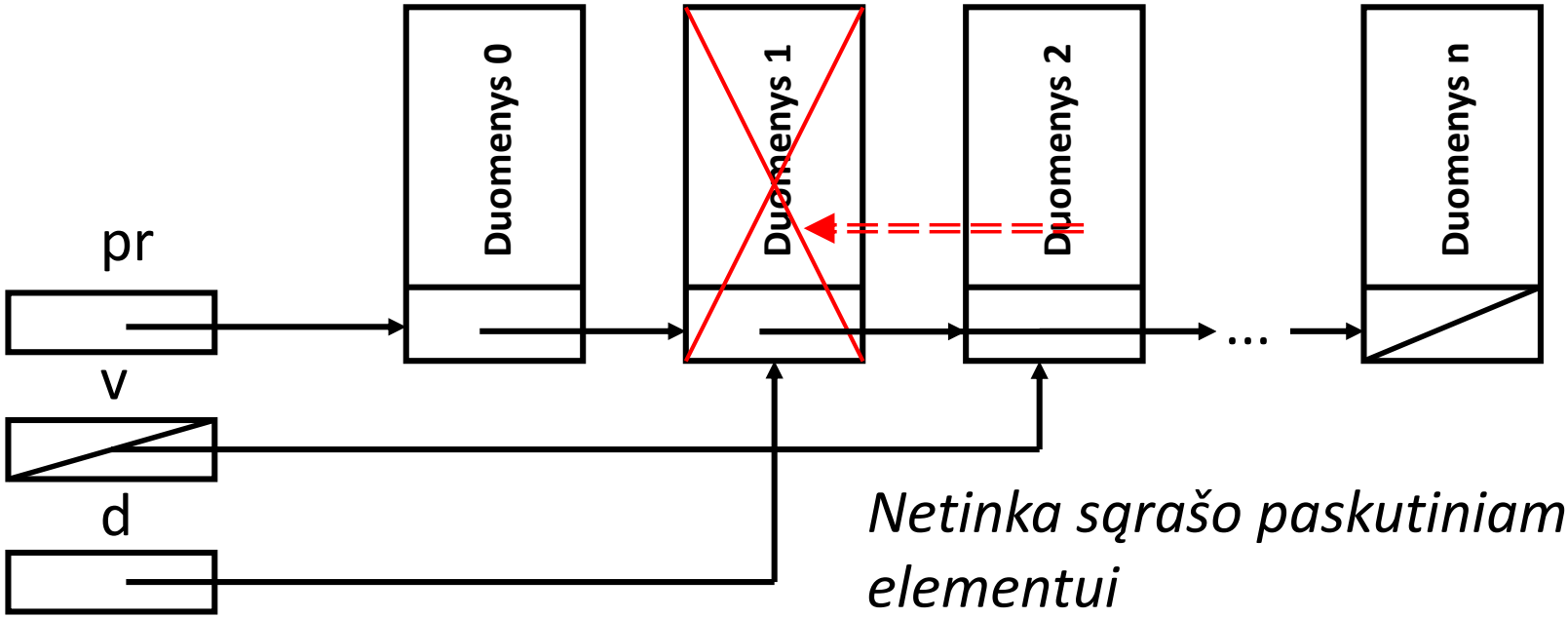
Sąrašo pradžios elementas išmetamas perskaičiuojant nuorodą pr.

Reikia rasti prieš tai esančio elemento adresą

Išmesti iš sąrašo = perrašyti adresą

Dabar galima atsisakyti elemento

Elemento pašalinimas iš sąrašo (reikšmių persiuntimas su animacija)



Netinka sąrašo paskutiniam elementui

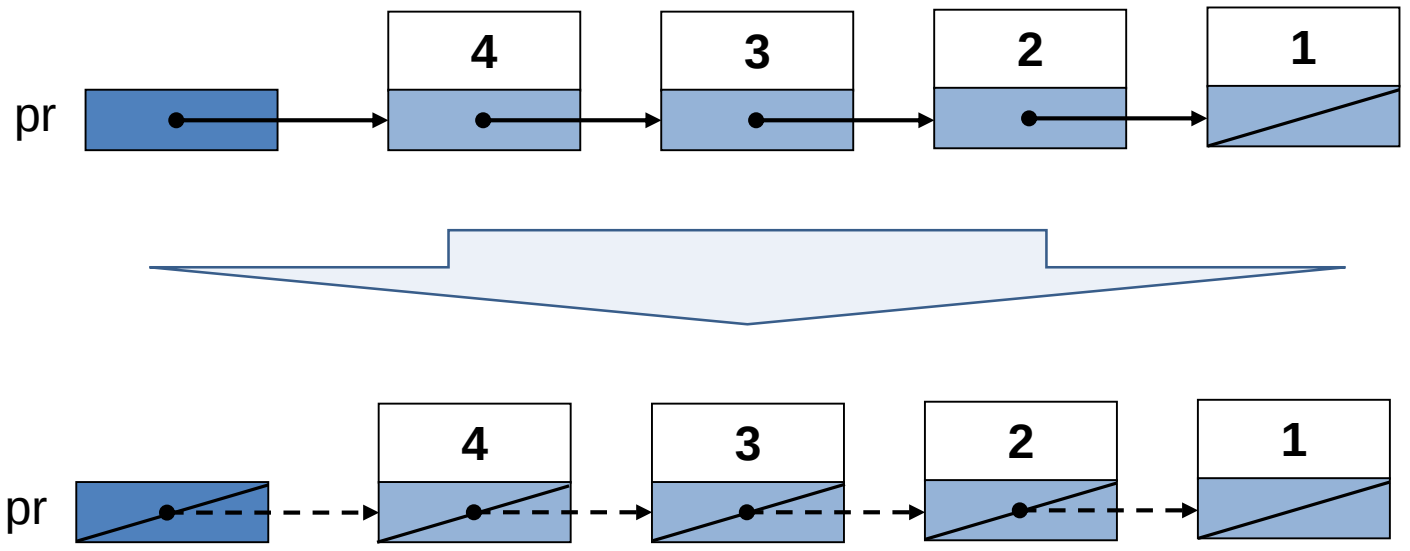
Įsidėmėkime sekančio elemento adresą

Išmesti iš sąrašo = perrašyti reikšmę

Dabar galima atsisakyti elemento (sekančio)

Sąrašo naikinimas 1/6

Sąrašo naikinimu suprantama **sąrašo struktūros sunaikinimas**, t.y. kiekvieno sąrašo elemento (mazgo) ryšio daliai priskiriant reikšmę `null`.



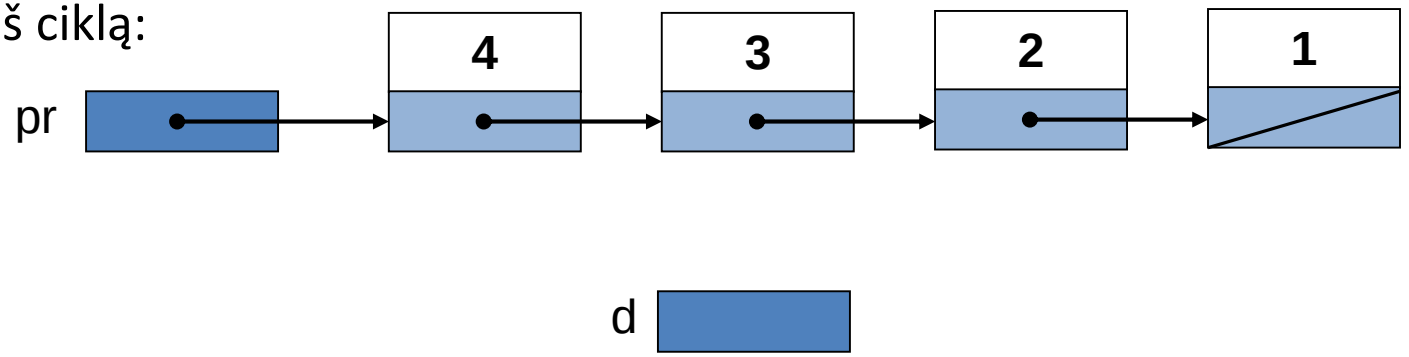
Pastaba: jeigu sąrašo elemento duomenų dalyje yra nuoroda į objektą, ten taip pat reikėtų įrašyti reikšmę `null`.

Sąrašo naikinimas 2/6

Kartoti pirmojo sąrašo elemento ryšio dalies sunaikinimą (`null`), kol sąrašo pradžios nuoroda įgaus reikšmę `null`:

```
Mazgas d;
while (pr != null)
{
    d = pr;
    pr = pr.Kitas;
    d.Kitas = null;
}
```

Prieš ciklą:

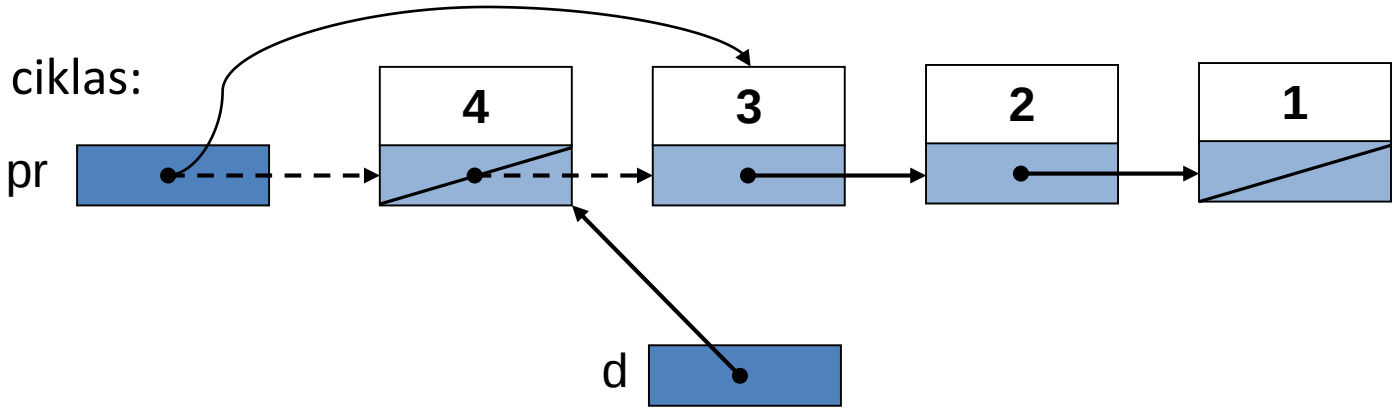


Sąrašo naikinimas 3/6

Kartoti pirmojo sąrašo elemento ryšio dalies sunaikinimą (`null`), kol sąrašo pradžios nuoroda įgaus reikšmę `null`:

```
Mazgas d;
while (pr != null)
{
    d = pr;
    pr = pr.Kitas;
    d.Kitas = null;
}
```

1-as ciklas:

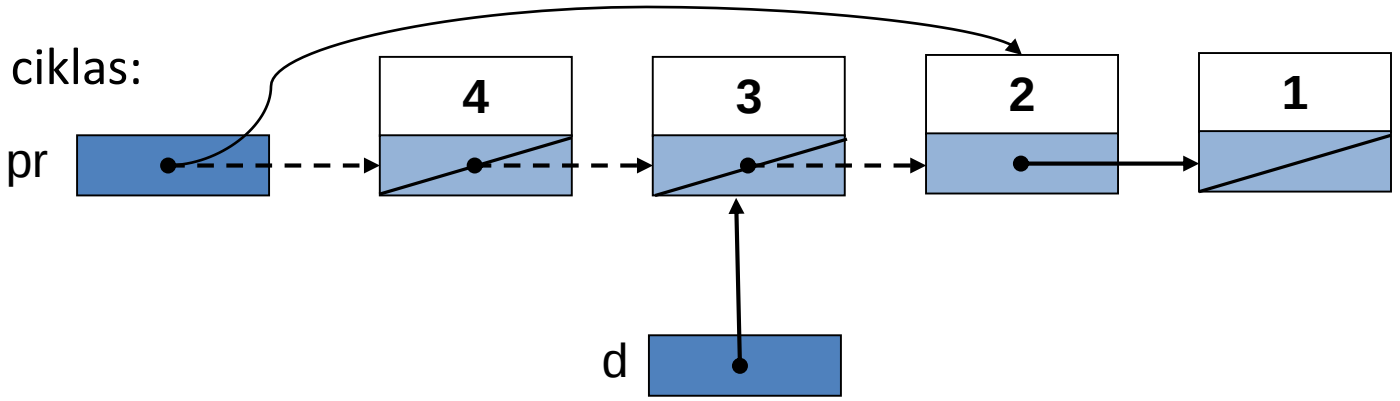


Sąrašo naikinimas 4/6

Kartoti pirmojo sąrašo elemento ryšio dalies sunaikinimą (`null`), kol sąrašo pradžios nuoroda įgaus reikšmę `null`:

```
Mazgas d;
while (pr != null)
{
    d = pr;
    pr = pr.Kitas;
    d.Kitas = null;
}
```

2-as ciklas:

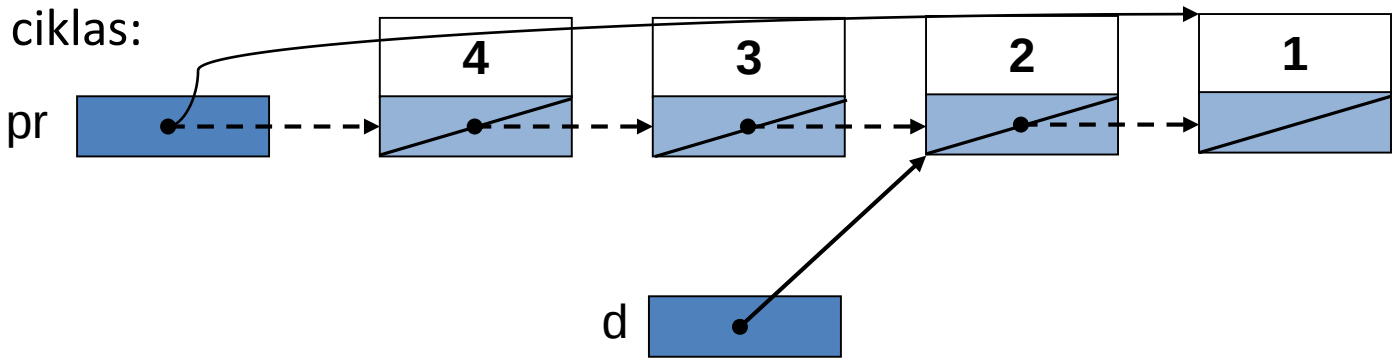


Sąrašo naikinimas 5/6

Kartoti pirmojo sąrašo elemento ryšio dalies sunaikinimą (`null`), kol sąrašo pradžios nuoroda įgaus reikšmę `null`:

```
Mazgas d;
while (pr != null)
{
    d = pr;
    pr = pr.Kitas;
    d.Kitas = null;
}
```

3-as ciklas:

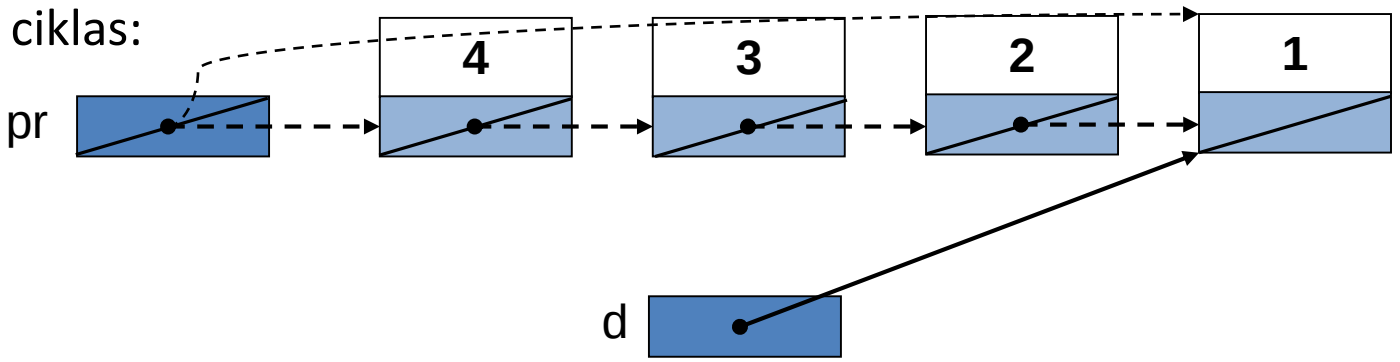


Sąrašo naikinimas 6/6

Kartoti pirmojo sąrašo elemento ryšio dalies sunaikinimą (`null`), kol sąrašo pradžios nuoroda įgaus reikšmę `null`:

```
Mazgas d;
while (pr != null)
{
    d = pr;
    pr = pr.Kitas;
    d.Kitas = null;
}
```

4-as ciklas:



Naujo elemento įterpimas 1/2

Reikia žinoti naujo elemento įterpimo sąraše vietą:

- sąrašo **pradžioje**;
- sąrašo **viduje**;
- sąrašo **pabaigoje**.

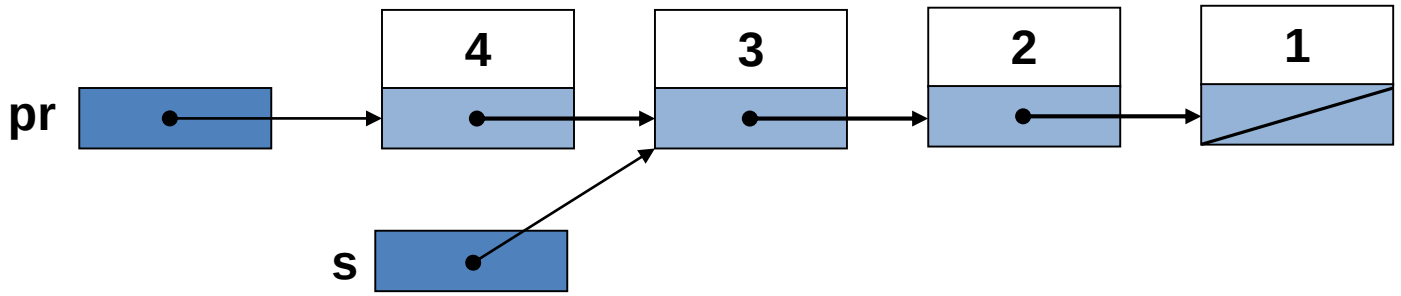
Pavyzdžiui:

- prieš didžiausią reikšmę (savybę) turintį elementą;
- už didžiausią reikšmę (savybę) turinčio elemento;
- nepažeidžiant rikiavimo tvarkos;

. . .

Naujo elemento įterpimas 2/2

Dirbant su vienkrypčiu sąrašu reikia surasti vietą, kur įterpti (**s**). Įterpimą **prieš**, reikia keisti įterpimu **po**.

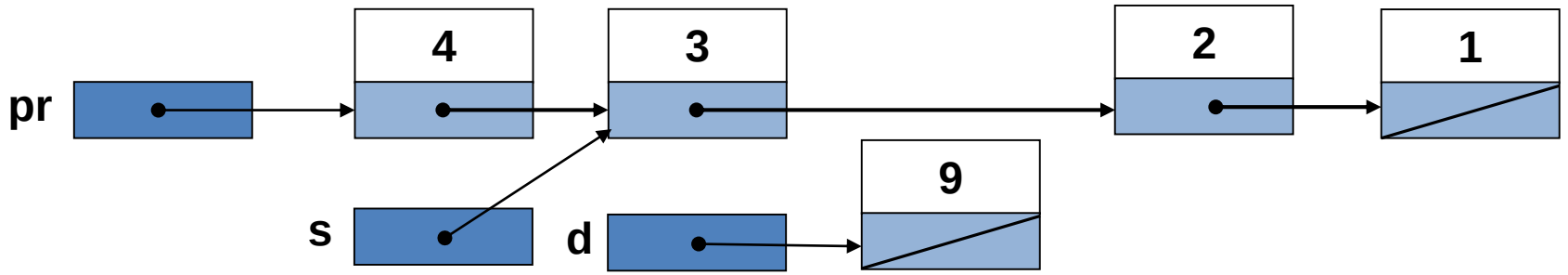


Įterpimas:

- sąrašo **pradžioje** atitinka sąrašo **formavimo atvirkštine seka** atvejį, kai naujas elementas jungiamas sąrašo pradžioje.
- sąrašo **pabaigoje** atitinka sąrašo **formavimo tiesiogine seka** atvejį, kai naujas elementas jungiamas sąrašo pabaigoje.

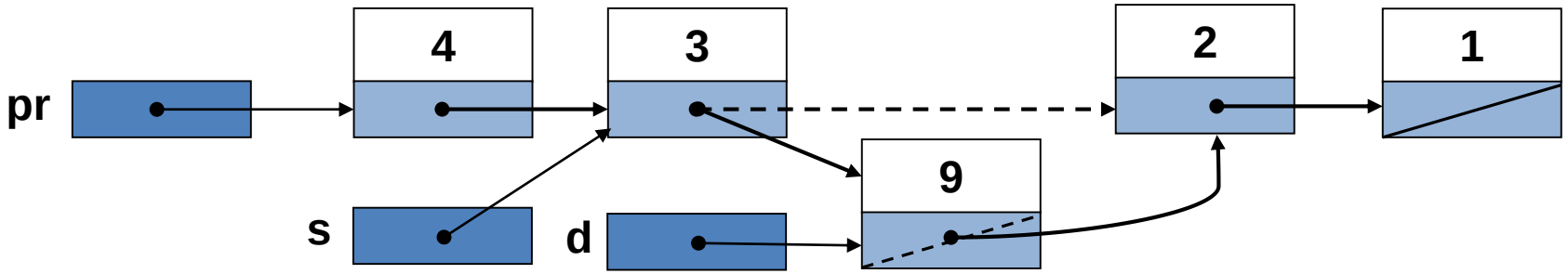
Naujo elemento įterpimas sąrašo viduje 1/2

```
Mazgas s;       // nuoroda, už kurios įterpiamas naujas elementas
int duom = 9; // įvedama arba skaičiuojama
...
Mazgas d = new Mazgas();
d.Duomenys = duom;
d.Kitas = null;
// arba Mazgas d = new Mazgas(duom, null);
...
```



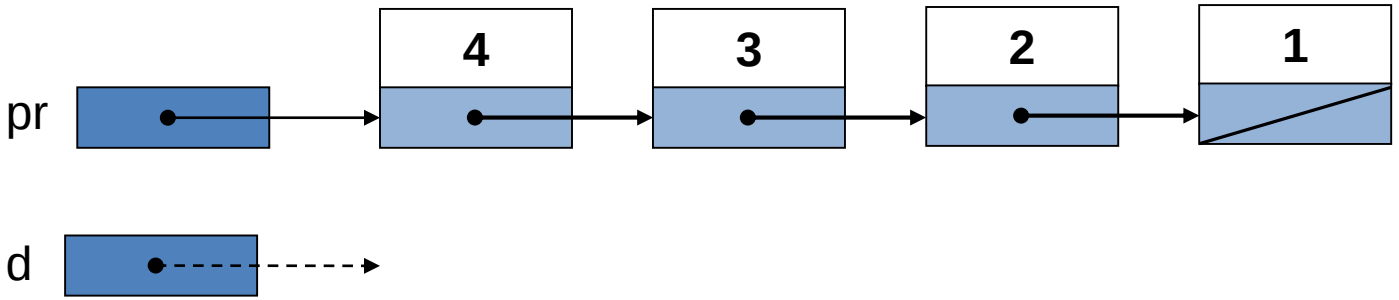
Naujo elemento įterpimas sąrašo viduje 2/2

```
...
d.Kitas = s.Kitas;
s.Kitas = d;
```



Paieška

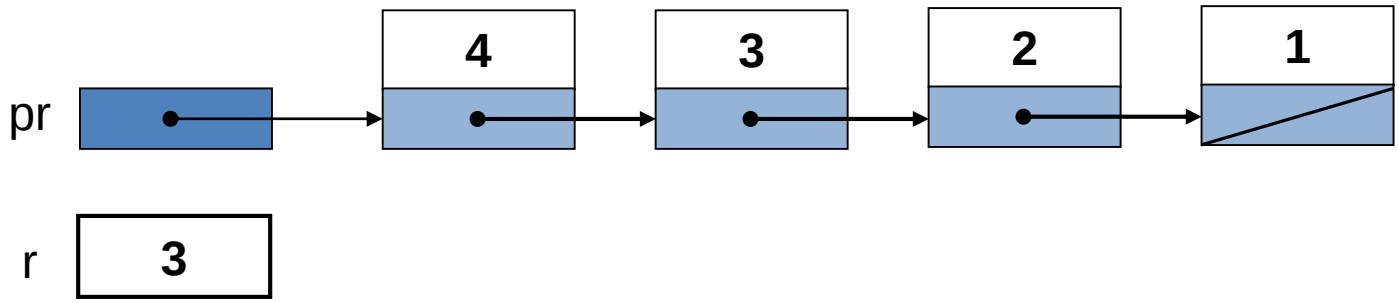
- Mažiausios (didžiausios) reikšmės paieška.
- Ieškoma konkrečios reikšmės sąraše (grąžinamas atsakymas **yra** arba **nėra**; arba gražinamas surastos reikšmės adresas, neradus – `null`).
- Ieškoma, kiek yra elementų, turinčių nurodytą požymį.
- . . .



Tinka visi nuoseklios peržiūros algoritmai, naudoti darbui su masyvais.

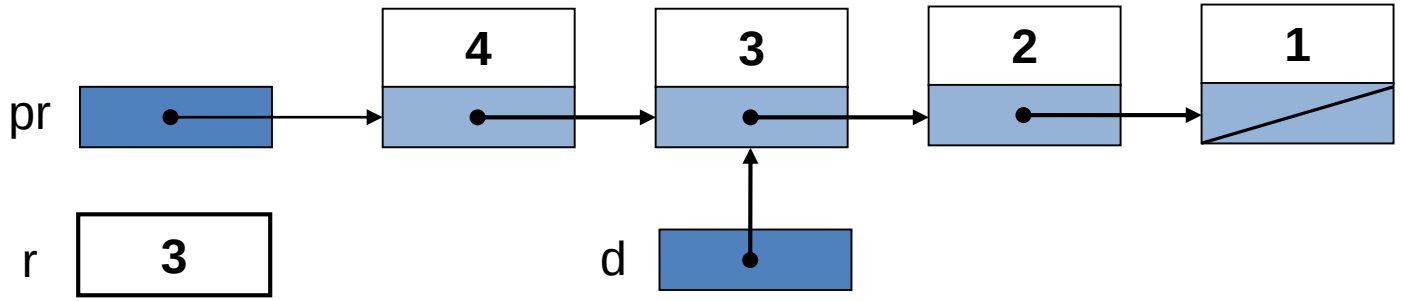
Reikšmės paieška – požymio grąžinimas

```
// Reikšmės paieška
// pr - sąrašo pradžia
// r - ieškoma reikšmė
// Grąžina true – jei reikšmė yra; false – jei nėra
static bool Yra(Mazgas pr, int r)
{
    for (Mazgas d = pr; d != null; d = d.Kitas)
        if (d.Duomenys == r)
            return true;
        return false;
}
```



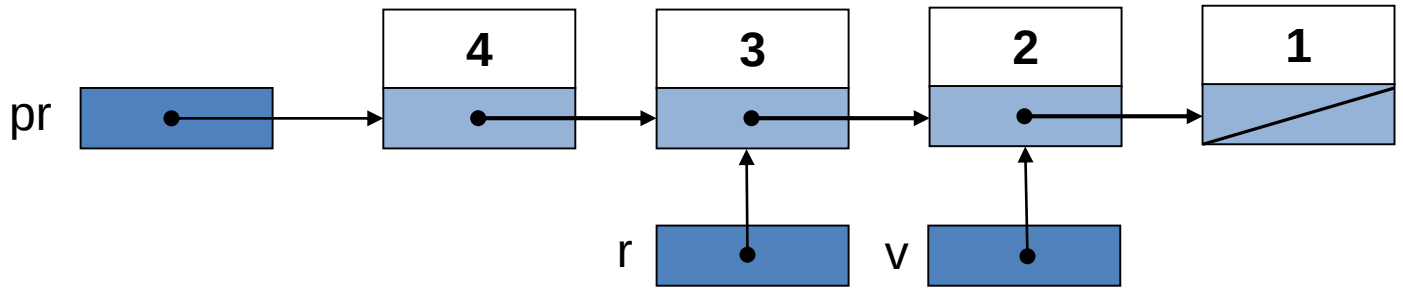
Reikšmės paieška – adreso grąžinimas

```
// Reikšmės paieška - adreso grąžinimas
// pr - sąrašo pradžia
// r - ieškoma reikšmė
// Grąžina surastos reikšmė elemento adresą, jei neranda - null
static Mazgas Vieta(Mazgas pr, int r)
{
    for (Mazgas d = pr; d != null; d = d.Kitas)
        if (d.Duomenys == r)
            return d;
        return null;
}
```



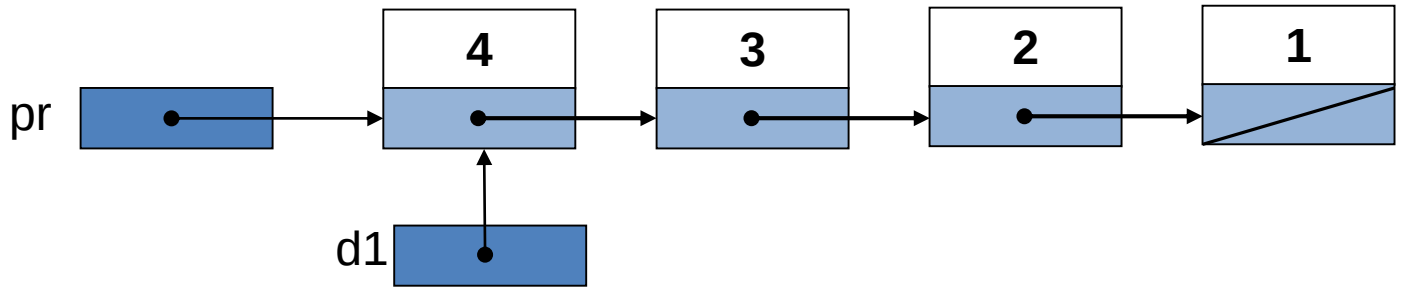
Elemento, esančio prieš žinomą, adreso radimas

```
// pr – sąrašo pradžia
// v – žinomas elementas
// Grąžina elemento esančio prieš v adresą, jeigu elemento nėra - null
static Mazgas VietaPries(Mazgas pr, Mazgas v)
{
    if (v != pr)
    {
        Mazgas r;
        for (r = pr; r.Kitas != v; r = r.Kitas)
            ;
        return r;
    }
    else
        return null;
}
```



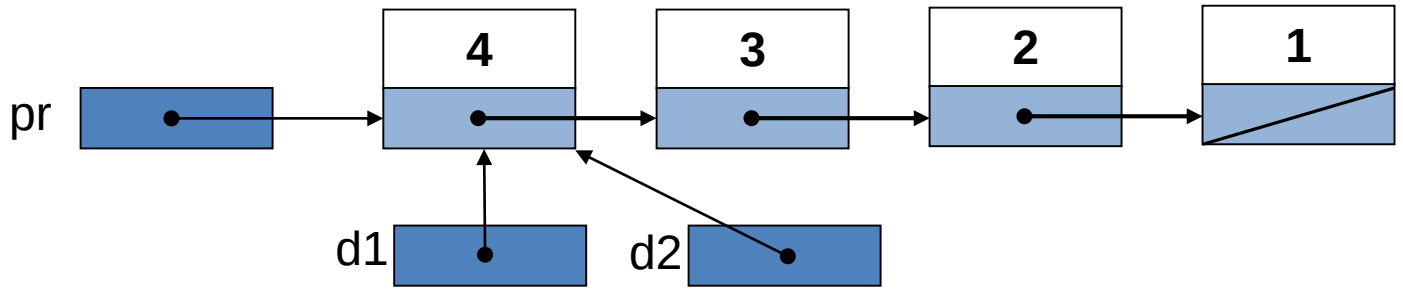
Sąrašo rikiavimas išrinkimo būdu (Minmax) 1/4

```
for (Mazgas d1 = pr; d1 != null; d1 = d1.Kitas)
{
  /* Nuoroda d1 rodo didžiausios reikšmės
     paieškos intervalo pradžią.
     Rasta didžiausia reikšmė bus sukeičiama su d1 elemento reikšme:
                  duomenų (informacinių) dalių sukeitimas vietomis.
  */
    ...
}
```



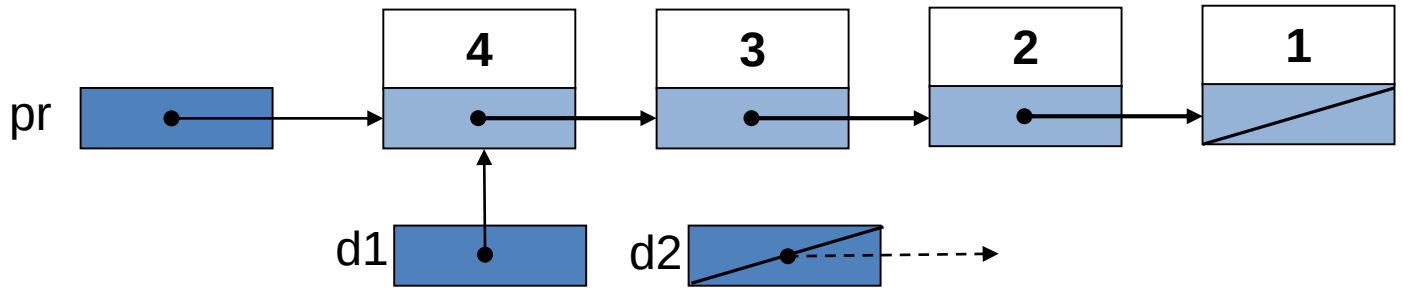
Sąrašo rikiavimas išrinkimo būdu (Minmax) 2/4

```
for (Mazgas d1 = pr; d1 != null; d1 = d1.Kitas)
{
    // Didžiausios reikšmės paieška intervale
    Mazgas maxv = d1;
    for (Mazgas d2 = d1; d2 != null; d2 = d2.Kitas)
        if (d2.Duomenys > maxv.Duomenys)
            maxv = d2;
    ...
}
```



Sąrašo rikiavimas išrinkimo būdu (Minmax) 3/4

```
for (Mazgas d1 = pr; d1 != null; d1 = d1.Kitas)
{
    ...
    // Duomenų (informacinių) dalių sukeitimas vietomis
    int k = d1.Duomenys;
    d1.Duomenys = maxv.Duomenys;
    maxv.Duomenys = k;
}
```



Sąrašo rikiavimas išrinkimo būdu (Minmax) 4/4

ktu informatikos fakultetas 1922

```
// Sąrašo rikiavimas MAŽĖJIMO tvarka
public void Minmax()              // Metodo vieta: sąrašo klasė
{
    for (Mazgas d1 = pr; d1 != null; d1 = d1.Kitas)
    {
        // Didžiausios reikšmės paieška intervale
        Mazgas maxv = d1;
        for (Mazgas d2 = d1; d2 != null; d2 = d2.Kitas)
            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.

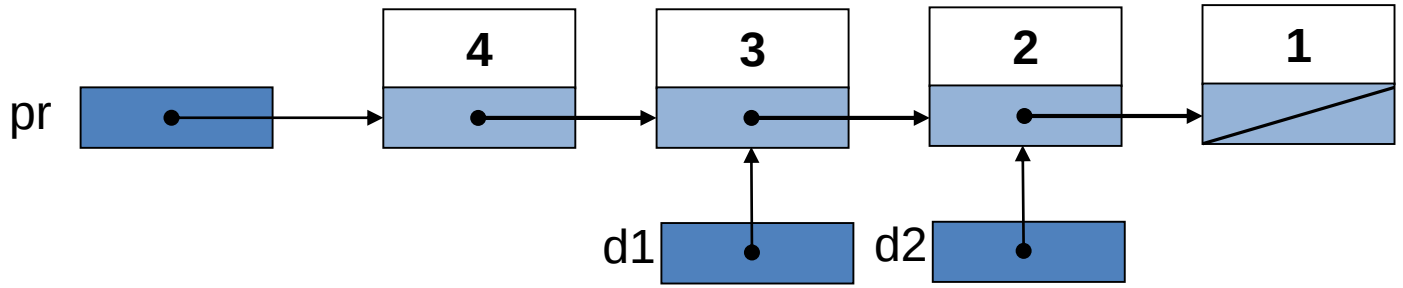
Sąrašo rikiavimas burbuliuko būdu

Nagrinėjami du gretimi elementai **d1** ir **d2** (pora): jie sukeičiami vietomis arba ne.

Sąrašas peržiūrimas pakartotinai daug kartų.

Rikiavimas baigiamas, kai eilinės peržiūros metu nebuvo sukeistas nė vienas elementas.

Sąrašas rikiuojamas, jeigu jame yra bent vienas elementas.

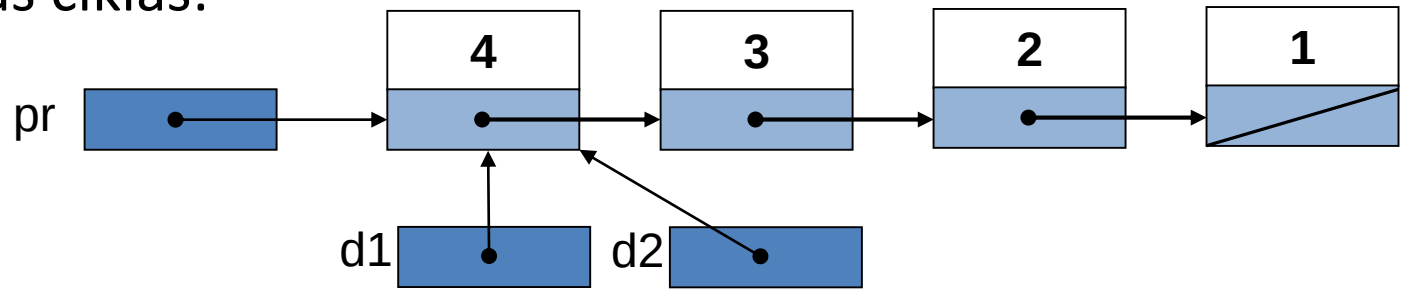


Sąrašo peržiūros ciklas imant gretimus elementus 1/5

```
...
Mazgas d1, d2;
d1 = d2 = pr;
while (d2 != null)
{
    Console.WriteLine("{0} ir {1}", d1.Duomenys, d2.Duomenys);
    d1 = d2;
    d2 = d2.Kitas;
}
...
```

```
4 ir 4
```

1-as ciklas:

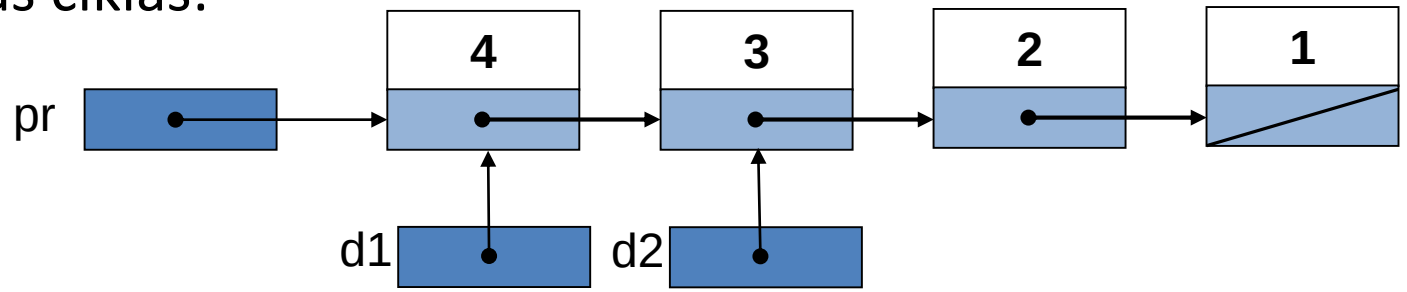


Sąrašo peržiūros ciklas imant gretimus elementus 2/5

```
...
Mazgas d1, d2;
d1 = d2 = pr;
while (d2 != null)
{
    Console.WriteLine("{0} ir {1}", d1.Duomenys, d2.Duomenys);
    d1 = d2;
    d2 = d2.Kitas;
}
...
```

```
4 ir 3
```

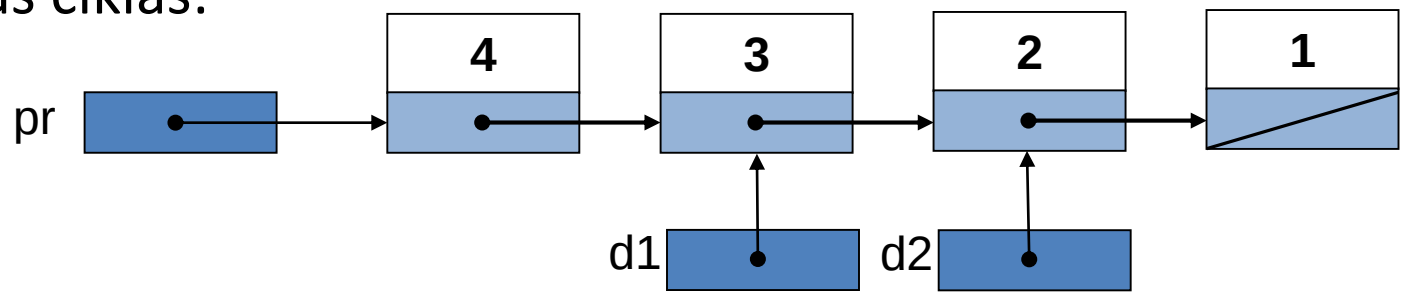
2-as ciklas:



Sąrašo peržiūros ciklas imant gretimus elementus 3/4

```
...
Mazgas d1, d2;
d1 = d2 = pr;
while (d2 != null)
{
    Console.WriteLine("{0} ir {1}", d1.Duomenys, d2.Duomenys);
    d1 = d2;
    d2 = d2.Kitas;
}
...
```

3-as ciklas:

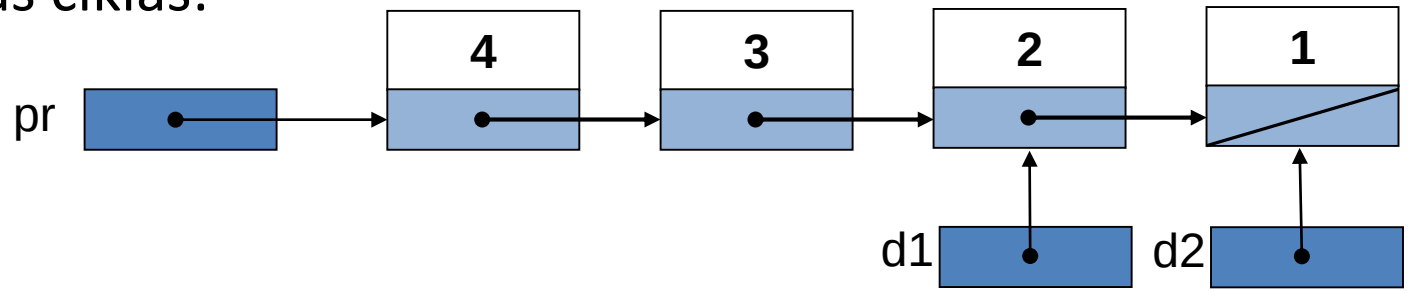


Sąrašo peržiūros ciklas imant gretimus elementus 4/4

```
...
Mazgas d1, d2;
d1 = d2 = pr;
while (d2 != null)
{
    Console.WriteLine("{0} ir {1}", d1.Duomenys, d2.Duomenys);
    d1 = d2;
    d2 = d2.Kitas;
}
...
```

2 ir 1

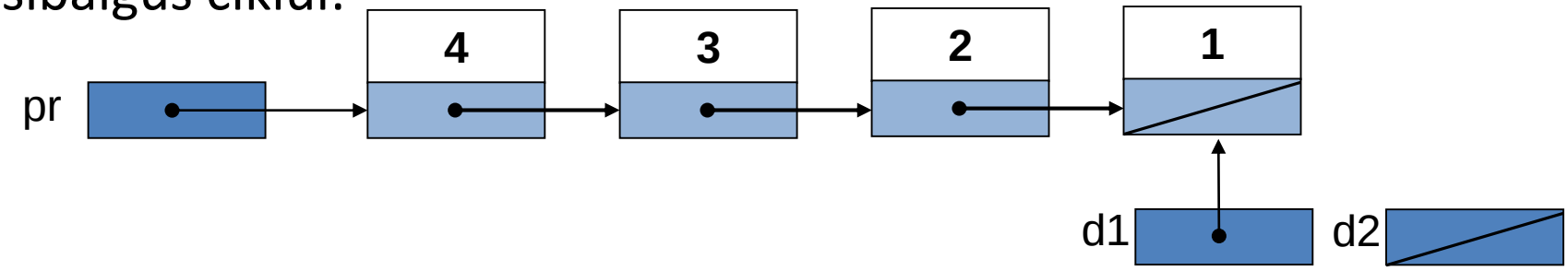
4-as ciklas:



Sąrašo peržiūros ciklas imant gretimus elementus 5/5

```
...
Mazgas d1, d2;
d1 = d2 = pr;
while (d2 != null)
{
    Console.WriteLine("{0} ir {1}", d1.Duomenys, d2.Duomenys);
    d1 = d2;
    d2 = d2.Kitas;
}
...
```

Pasibaigus ciklui:



Sąrašo rikiavimas burbuliuko būdu 1/2

```
public void Burbulas()
{
    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.Kitas;
        }
    }
```

Sąrašo rikiavimas burbuliuko būdu 2/2

ktu informatikos fakultetas 1922

```
public void Burbulas()
{
    if (pr == null) { return; }
    bool keista = true;
    while (keista)
    {
        keista = false;
        Mazgas d = pr;
        while (d.Kitas != null)
        {
            if (d.Kitas.Duomenys > d.Duomenys)
            {
                int k = d.Duomenys;
                d.Duomenys = d.Kitas.Duomenys;
                d.Kitas.Duomenys = k;
                keista = true;
            }
            d = d.Kitas;
        }
    }
}
```

Kodėl reikalingas šis tikrinimas?

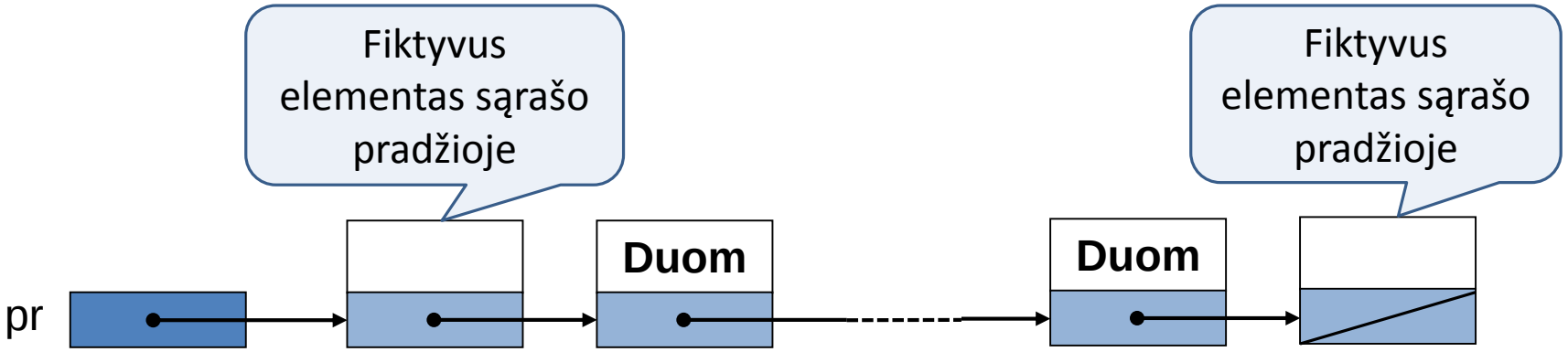
Sąrašas su fiktyviais elementais 1/2

Pirmas ir paskutinis elementai nenaudojami informacijai saugoti.

Sąrašas niekada nebus tuščias.

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

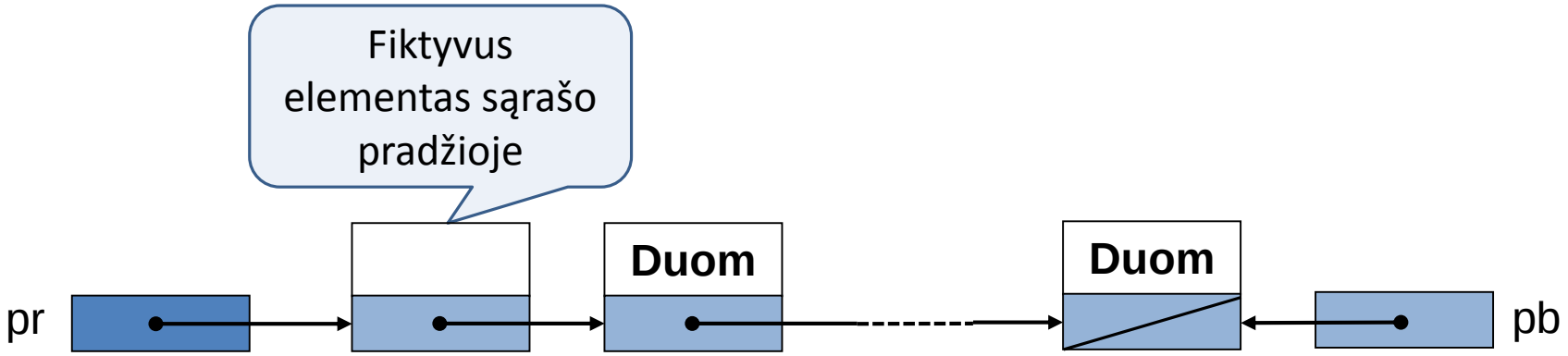
Atliekant veiksmus su sąrašo elementais reikia tai įvertinti.



Sąrašas su fiktyviais elementais 2/2

Galimi sąrašai tik su vienu fiktyviu elementu.

Patogu turėti nuorodą (**pb**) į sąrašo paskutinį elementą.



Sąrašo klasė su fiktyviais elementais

ktu informatikos fakultetas 1922

```
public sealed class Sąrašas
{
    private Mazgas pr;    // sąrašo pradžia
    private Mazgas pb;    // sąrašo pabaiga
    private Mazgas pbt;   // sąrašo pabaiga (papildomas)
    private Mazgas ss;    // sąrašo sąsaja
    // Konstruktorius: sukuriami du fiktyvūs elementai
    public Sąrašas()
    {
        this.pb = new Mazgas(Int32.MaxValue, null);
        this.pr = new Mazgas(Int32.MaxValue, pb);
        this.pbt = pr;
        this.ss = null;
    }
    // Sąsajos metodai
    ...
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 1/4

ktu informatikos fakultetas 1922

```
public sealed class Sąrašas
{
    ...
    // Sąsajos metodai
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PRADŽIOS
    // skaicius – naujo elemento reikšmė (duomenys)
    public void DėtiDuomenisA(int skaicius)
    {
        var d = new Mazgas(skaicius, pr.Kitas);
        pr.Kitas = d;
    }
    // Sukuriamas sąrašo elementas ir prijungiamas prie sąrašo PABAIGOS
    // skaicius – naujo elemento reikšmė (duomenys)
    public void DėtiDuomenisT(int skaicius)
    {
        pbt.Kitas = new Mazgas(skaicius, pb);
        pbt = pbt.Kitas;
    }
    ...
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 2/4

ktu informatikos fakultetas 1922

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

Sąrašo klasės su fiktyviais elementais sąsajos metodai 3/4

ktu informatikos fakultetas 1922

```
public sealed class Sąrašas
{
    ...
    // Sąsajai priskiriama sąrašo pradžia
    public void Pradžia()
    {
        ss = pr.Kitas;
    }
    // Sąsajai priskiriamas sąrašo sekantis elementas
    public void Kitas()
    {
         ss = ss.Kitas;
    }
    // Grąžina true, jeigu sąsaja netuščia; false - priešingu atveju
    public bool Yra()
    {
        return ss.Kitas != null;
    }

    ...
}
```

Sąrašo klasės su fiktyviais elementais sąsajos metodai 4/4

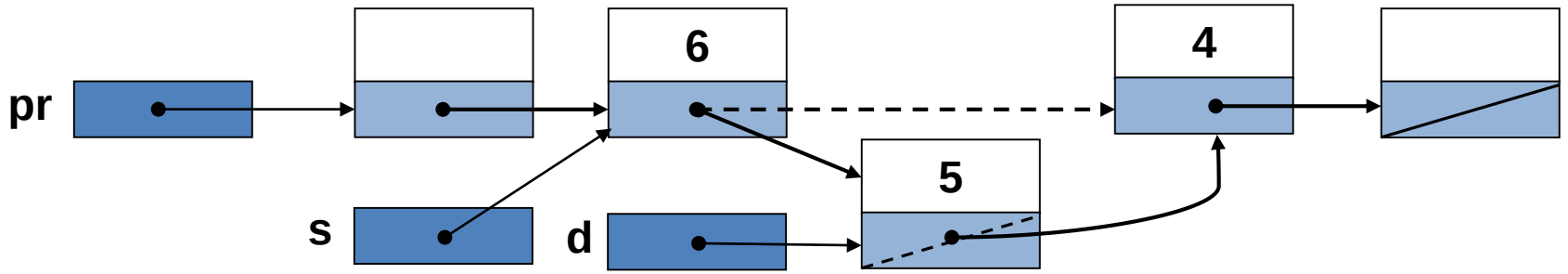
ktu informatikos fakultetas 1922

```
public sealed class Sąrašas
{
    ...
    // Sunaikinamas sąrašas
    public void Naikinti()
    {
        while (pr != null)
        {
            ss = pr;
            pr = pr.Kitas;
            ss.Kitas = null;
        }
        pb = pbt = ss = pr;   // pb = pbt = ss = null;
    }
}
```

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

Elemento įterpimas (sąrašas su fiktyviais elementais)

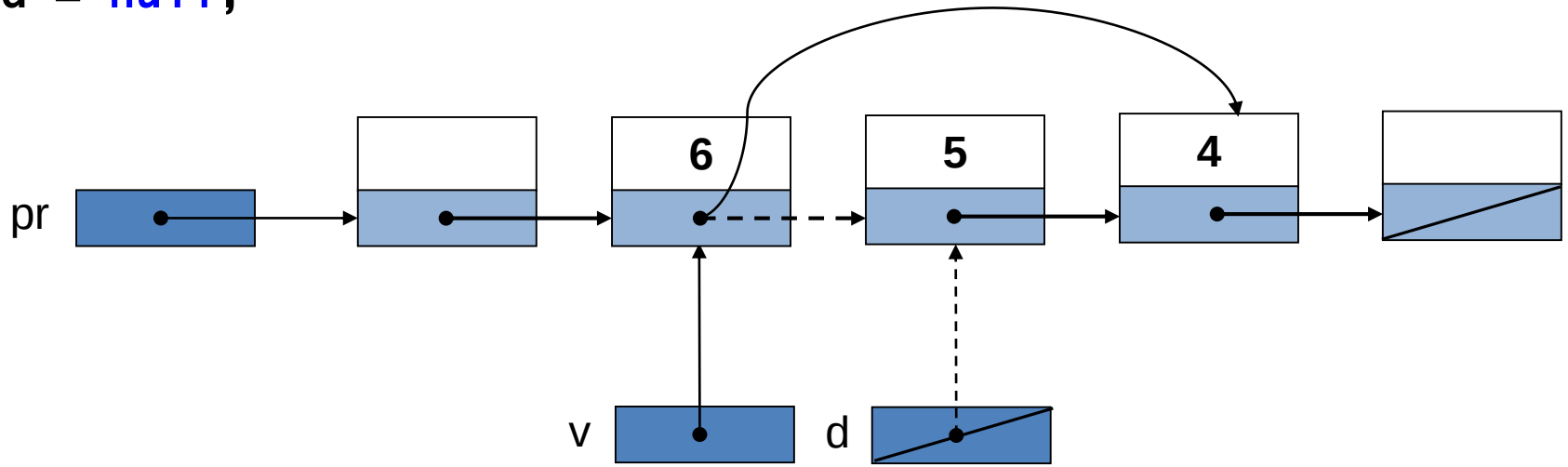
```
Mazgas s;       // nuoroda, už kurios įterpiamas naujas elementas
int duom = 5; // įvedama arba skaičiuojama
...
Mazgas d = new Mazgas();
d.Duomenys = duom;
d.Kitas = null;
// arba Mazgas d = new Mazgas(duom, null);
d.Kitas = s.Kitas;
s.Kitas = d;
...
```



Elemento šalinimas (sąrašas su fiktyviais elementais)

Randamas šalinamo elemento adresas **d** ir prieš jį esančio elemento adresas **v**; po to:

```
Mazgas d; // šalinamas elementas
Mazgas v; // elementas, esantis prieš šalinamjį
...
v.Kitas = d.Kitas;
d = null;
```



Sąrašo su fiktyviais elementais rikiavimas išrinkimo būdu (Minmax)

ktu informatikos fakultetas 1922

```
// Sąrašo rikiavimas MAŽĖJIMO tvarka
public void Minmax()              // Metodo vieta: sąrašo klasė
{
    for (Mazgas d1 = pr.Kitas; d1.Kitas != null; d1 = d1.Kitas)
    {
        // Didžiausios reikšmės paieška intervale
        Mazgas maxv = d1;
        for (Mazgas d2 = d1; d2.Kitas != null; d2 = d2.Kitas)
            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 su fiktyviais elementais rikiavimas burbuliuko būdu 1/2

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

Sąrašo su fiktyviais elementais rikiavimas burbuliuko būdu 2/2

```
public void Burbulas()
{
    if (pr.Kitas.Kitas == null) { return; }
    bool keista = true;
    while (keista)
    {
        keista = false;
        Mazgas d = pr;
        while (d.Kitas.Kitas != null)
        {
            if (d.Kitas.Duomenys > d.Duomenys)
            {
                int k = d.Duomenys;
                d.Duomenys = d.Kitas.Duomenys;
                d.Kitas.Duomenys = k;
                keista = true;
            }
            d = d.Kitas;
        }
    }
}
```

Kodėl reikalingas šis tikrinimas?

Būtinos sąlygos darbui su sąrašais

Dirbant su susietaisiais sąrašais **būtina patikrinti** metodų veikimą, kai:

- sąrašas tuščias
- sąraše yra tik vienas elementas
- apdorojamas pirmas sąrašo elementas
- apdorojamas paskutinis sąrašo elementas
- apdorojamas vidinis sąrašo elementas

ktu informatikos fakultetas 1922

Šioje temoje susipažinote:

1. Reikšmių sukeitimu
2. Elementų naikinimu
3. Sąrašo naikinimu
4. Elemento įterpimu
5. Sąrašo rikiavimu
6. Sąrašu su fiktyviais elementais

Klausimai?