**Susietasis sąrašas**

**Vienkryptis sąrašas**

*Duomenų struktūros aprašas:*

public sealed class Mazgas

{

public int Duom { get; private set; }

public Mazgas Kitas { get; set; }

public Mazgas() { }

public Mazgas(int reikme, Mazgas adr)

{

Duom = reikme;

Kitas = adr;

}

}

*Klasės aprašo pradžia:*

public sealed class Sąrašas

{

private Mazgas pr;

private Mazgas pb;

private Mazgas d;

public Sąrašas()

{

this.pr = null;

this.pb = null;

this.d = null;

}

}

*Sąsajos metodas:*

public Studentas ImtiDuomenis()

{

return d.Duom;

}

*Sąrašo paveikslas:*

Mazgas p1, p2;

p1 = new Mazgas;

p2 = new Mazgas;

p1.Kitas = p2;

*Sąrašo formavimas tiesiogine tvarka:*

Naujas elementas jungiamas į sąrašo pabaigą. Rodyklė d – nebūtina.

public void DėtiDuomenisT(Studentas naujas)

{

var dd = new Mazgas(naujas, null);

if (pr != null) {

pb.Kitas = dd;

pb = dd;

}

else {

pr = dd;

pb = dd;

}

}

*Sąrašo formavimas atvirkštine* tvarka*:*

public void DėtiDuomenisA(Studentas naujas)

{

pr = new Mazgas(naujas, pr);

}

*Sąrašo spausdinimas (elementų peržiūra):*

Sąrašo peržiūra (spausdinimas, informacijos paieška) gali būti realizuojama tokiu ciklu:

for (Mazgas d = pr; d != null; d = d.Kitas) {

// Čia galima naudoti elemento d informacinę dalį d.Duom;

}

*Sąrašo rikiavimas:*

public void Rikiuoti()

{

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

{

Mazgas maxv = d1;

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

if (d2.Duom < maxv.Duom)

maxv = d2;

// Informacinių dalių sukeitimas vietomis

Studentas St = d1.Duom;

d1.Duom = maxv.Duom;

maxv.Duom = St;

}

}

*Sąrašo rikiavimas burbuliuko metodu:*

public void RikiuotiB() {

if (pr == null) { return; }

bool keista = true;

while (keista) {

keista = false;

var pra = pr;

while (pra.Kitas != null) {

if (pra.Duom > pra.Kitas.Duom) {

Studentas St = pra.Duom;

pra.Duom = pra.Kitas.Duom;

pra.Kitas.Duom = St;

keita = true;

}

pra = pra.Kitas;

}

}

}

*Elementų paieška tvarkingame ir netvarkingame sąraše:*

bool Yra(int rr)

{

for (Mazgas d = pr; d != null; d = d.Kitas)

if (d.Duom == rr)

return true;

return false;

}

Mazgas Vieta(int rr)

{

for (Mazgas d = pr; d != null; d = d.Kitas)

if (d.Duom == rr)

return d;

return null;

}

|  |  |
| --- | --- |
| *Įterpimas:*  // Turi būti žinomas s – adresas elemento, po kurio įterpiama  d.Kitas = s.Kitas;  s.Kitas = d; | *Šalinimas:*  v.Kitas = d.Kitas;  d = null; |
| *Viso sąrašo naikinimas:*  while (pr != null) {  Mazgas d = pr;  pr = pr.Kitas;  d = null;  } | *Informacinių dalių sukeitimas:*  int k = s1.Duom;  s1.Duom = s2.Duom;  s2.Duom = k; |

**Dvikryptis sąrašas**

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

*Duomenų struktūros aprašas:*

public sealed class Mazgas

{

public Studentas Duom { get; set; }

public Mazgas Kaire { get; set; }

public Mazgas Desine { get; set; }

public Mazgas(Studentas reikme, Mazgas adrk, Mazgas adrd)

{

Duom = reikme;

Kaire = adrk;

Desine = adrd;

}

}

*Sąrašo paveikslas:*

*Sąrašo formavimas tiesiogine tvarka:*

public void DėtiDuomenisT(Studentas naujas)

{

var dd = new Mazgas(naujas, pb, null);

if (pr != null)

pb.Desine = dd;

else

pr = dd;

pb = dd;

}

*Sąrašo formavimas atvirkštine tvarka:*

public void DėtiDuomenisA(Studentas naujas)

{

var dd = new Mazgas(naujas, null, pr);

if (pr != null)

pr.Kaire = dd;

else

pb = dd;

pr = dd;

}

*Sąrašo spausdinimas (elementų peržiūra):*

// Nuo pradžios:

for (Mazgas d = **pr**; d != null; d = d.**Desine**)

// Veiksmai su d.Duom;

// Nuo pabaigos:

for (Mazgas d = **pb**; d != null; d = d.**Kaire**)

// Veiksmai su d.Duom;

*Elemento įterpimas už nurodyto elemento:*

void ĮterptiUž(Mazgas rr, int kk) {

Mazgas dd = new Mazgas(kk, rr, rr.Desine);

if (pb == rr)

pb = dd;

else

dd.Desine.Kaire = dd;

rr.Desine = dd;

}

*Elemento įterpimas prieš nurodytą elementą:*

void ĮterptiPrieš(Mazgas rr, int kk)

{

if (rr == pr) {

Mazgas dd = new Mazgas(kk, null, pr);

pr.Kaire = dd;

pr = dd;

} else {

rr = rr.Kaire;

ĮterptiUž(rr, kk);

}

}

*Šalinimas iš dvikrypčio sąrašo:*

void Šalinti(Mazgas dd)

{

if (dd == pr) pr = pr.Desine;

if (dd == pb) pb = pb.Kaire;

if (dd.Kaire != null)

dd.Kaire.Desine = dd.Desine;

if (dd.Desine != null)

dd.Desine.Kaire = dd.Kaire;

dd = null;

}

**Sąrašas su fiktyviais elementais**

Pirmasis ir paskutinis elementai nenaudojami duomenims saugoti;

Sąrašas niekuomet nebus tuščias;

Elementų įterpimas ir šalinimas vyksta tik sąrašo viduje (mažiau tikrinimo veiksmų).

private Mazgas pr; // sąrašo pradžia

private Mazgas pb; // sąrašo pabaiga

private Mazgas pre; // eilei

private Mazgas pab; // dėklui

private Mazgas d; // sąrašo sąsaja

public Sąrašas()

{

pr = Mazgas(new Studentas(), null, null);

pb = Mazgas(new Strudentas(), pr, null);

pre = pr;

pr.Desine = pb;

pab = pb;

d = null;

}

|  |  |
| --- | --- |
| public void DėtiDuomenisA(Studentas naujas)  {  pab.Kaire = new Mazgas(naujas, pr, pab);  pb = pab.Kaire;  pr.Desine = pab;  } | public void DėtiDuomenisT(Studentas naujas)  {  pre.Desine = new Mazgas(naujas, pre, pb);  pre = pre.Desine;  pb.Kaire = pre;  } |

public void Pradžia() { d = pr.Desine; }

public void Kitas() { d = d.Desine; }

public bool Yra() { return d.Desine != null; }

|  |  |
| --- | --- |
| void Šalinti(Mazgas dd)  {  dd.Desine.Kaire = dd.Kaire;  dd.Kaire.Desine = dd.Desine;  dd = null;  } | void ĮterptiUž(Mazgas dd, int kk)  {  Mazgas rr = new Mazgas(kk, dd, dd.Desine);  dd.Desine = rr;  rr.Desine.Kaire = rr;  } |