Szabad := Elem[SzabadJ.Köv; (a listaelem a szabad listában a Köv-re mutasson)

Lefoglal := Igaz;

Különben

Lefoglal := Hamis;

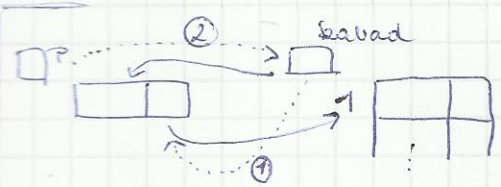
H vége;

FGV vége.

Eljárás Felszabadít (Hely : MutatóTípus);

Algoritmus

Ha Hely <> Végjel Akkor (megmutatja, mit kell bekötni)



(Ha a szabad lista fejében végjel van, elvesztjük a szabad lista elemeit.)

Elem[Hely].Köv := Szabad;

Szabad := Hely;

H vége;

E vége;

Műveletek láncolt listával:

Eljárás Listakezd (Fej : MutatóTípus);

(inicializáláskor a listafej tartalmazzon végjelet.)

Algoritmus

Fej := Végjel;

E vége;

Eljárás ListaFeldolg (Fej: MutatóTípus);

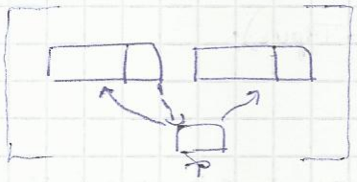
Változó P: MutatóTípus; (mutasson a köv elemre)

Algoritmus

P:=Fej; (kezdőérték a fej → a lista első elemére mutat először)

Ciklus míg P<> Végjel

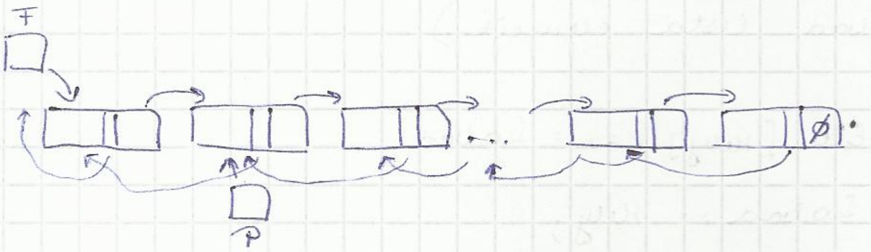
→Feldolgozó_eljárás (Elem [P]. Érték);



P:= Elem [P]. Köv;

C vége;

E vége;



- Visszafelé nem lehet mászkálni a listában
- Bevezetünk még 1 mutatót: KÉTIRÁNYÚ LÁNCOLT LISTA
- A listát annyiféle képpen tudjuk bejárni, ahány mutató tartozik hozzá. (az elemekhez)

- Ha a listaelem utolsó mutatója az első elemre mutat, akkor lesz vége, ha P értéke = a fej értékével: CIKLIKUSAN LÁNCOLT LISTA

Van 1 kitüntetett elem, amelyre a fej mutat ⇒ Van első elem.

GYŰRŰ: Ha a fej nem az első elemre mutat.

algo II. EA.

3. előadás

Függvény Listakeresés (Fej, Hely: Mutató Típus; Adat: ÉrtékTípus): logikai

Fej: egyértelműen azonosítja a listát

Hely: Megmutatja, hol találta meg a keresett elemet.

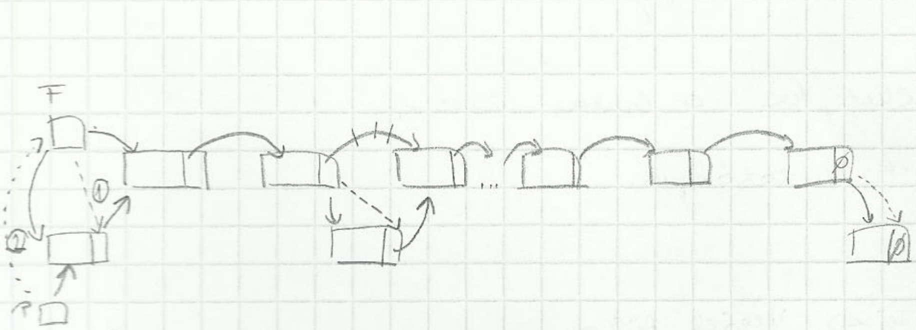
változó P: Mutató Típus;

Algoritmus

i:=1;	P:=Fej
ciklus míg (i ≤ n) ∧ (A[i] ≠ adat)	ciklus míg (P<> végjel) és (Elem[P].érték <> Adat)
i:=i+1;	P:=Elem[P].köv;
c vége;	c vége
KERES:=(i ≤ n);	Hely:=P;
	listakeresés := (P<> Végjel);

c vége

- lehet keresni a listában lineárisan, ha rendezett, de [(E+U) DIV 2 -> nem dinamikus adatszerkezet] nem éri meg



Az új elembe beírjuk annak a mutatóját, ami után befűzzük.

Függvény ListaFűzElöl(Fej: Mutató típus; Adat: Elem Típus): logikai;

változó P: Mutató típus

Algoritmus

Nem tudjuk bárikai adatmezőnek, ha nem tudunk lefoglalni hozzá helyet.

Ha Lefoglal(P) akkor

ha az érték igaz, akkor a P a lefoglalt elemre mutat. ha a visszatérési érték hamis, nem elég lefűzni.

Elem[P].Érték := Adat;

Elem[P].Köv := Fej;

Fej := P;

ListaFűzElöl := Igaz;

különben

ListaFűzElöl := Hamis;

Ha vége

Függvény ListaTörölElöl(Fej: Mutatótípus): logikai;

↳ cím szerinti átadás, mert meg fog változni

Nem sikerülhet, ha a lista üres.

változó P: Mutatótípus;

Algoritmus

Ha Fej <> Végjel akkor

P := Fej

Fej := Elem[Fej].Köv;

Felszabadít(P);

algo II.
EA

Lista TörölElől := Igaz;

különben

Lista TörölElől := Hamis;

H vége ;

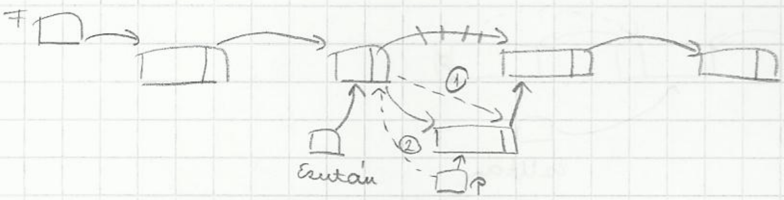
F vége;

Függvény ListaFűzElemUtán (EzUtán : Mutatótípus; Adat : ÉrtékTípus) : logikai;

változó P : Mutatótípus; // a szabadlista Eseléset segíti elő

Algoritmus

Ha (EzUtán <> Végjel) és Lefoglal (P) Akkor



Nem lehet végrehajtani, ha az ezután nem mutat sehová, v. a szabadlista üres.

Elem [P]. Érték := Adat;

Elem [P]. Köv := Elem [EzUtán]. Köv ; } nem felcserélhető!

Elem [EzUtán]. Köv := P;

ListaFűzElemUtán := Igaz;

különben

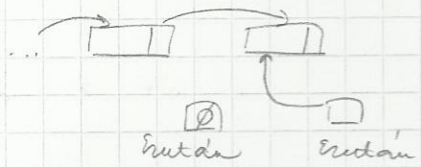
ListaFűzElemUtán := Hamis;

h vége ;

f vége ;

Függvény ListaTörölElemUtán (EzUtán : Mutatótípus): logikai

EzUtán = végjel, vagy utolsó elem ⇒ nem lehet törölni



változó P: Mutatótípus;

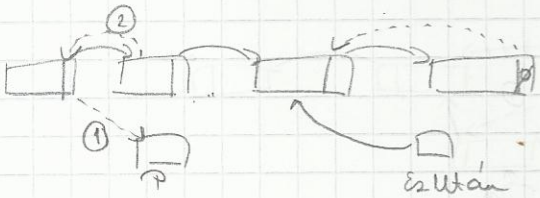
Algoritmus:

Ha (EzUtán <> végjel) és

(Elem[EzUtán].köv <> végjel) akkor

P := Elem[EzUtán].köv;

Elem[EzUtán].köv := Elem[P].köv;



Felszabadít(P);

ListaTörölElemUtán := Igaz;

különben

ListaTörölElemUtán := Hamis;

Ha vége;

f vége;

algo II.
EA

Függvény ListaFűzVég (Fej: MutatóTípus;
Adat: ÉrtékTípus): logikai;

változó P: mutató típus

Üres listában a végére való befűzés az elejére való befűzés.

Algoritmus

Ha Fej = Végjel akkor

ListaFűzVég := ListaFűzElől (Fej, Adat)

Különben

P := Fej;

Ciklus Míg Elem[P].Köv <> Végjel

P := Elem[P].Köv; → P utolsó elemre mutat

Ciklus vége;

ListaFűzVég := ListaFűzElemUtán (P, Adat);

Ha vége;

F vége;

— o —

Függvény ListaTörölVég (Fej: MutatóTípus): Logikai;

Nem tudunk törölni, ha üres a lista.

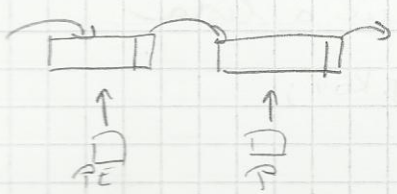
Nincsenek átadás. Ha a törlendő elem az első.

változó P, PE: MutatóTípus;

Elemek a megelőző elem

Mindig igaz:

P = Elem[PE].KÖV



```
Algoritmus
Ha Fej <> Végjel Akkor
  Ha Elem[Fej].Köv = Végjel Akkor
      ListaTörölVég := ListaTörölElső(Fej)
  különben
    PE := Fej;
    P := Elem[Fej].Köv;
    Ciklus míg Elem[P].Köv <> Végjel
      PE := P; P := Elem[P].Köv;
    Ciklus vége
    ListaTörölVég := ListaTörölElemUtán(PE);
  Ha vége
Különben
  ListaTörölVég := Hamis;
Ha vége;
F vége;
```

ha a listának legalább 1 eleme van.

```
Függvény ListaFilzRead(Fej : Mutatótípus; Adat: Értéktíp): logikai;
Nem ismert, ha nem tudunk helyet foglalni.
Változó PE,P : Mutatótípus;
Algoritmus
  Ha (Fej = Végjel) vagy (Adat < Elem[Fej].Érték) Akkor
                    elején bővítjük
    ListaFilzRead := ListafilzElöl(Fej,Adat);
  Különben legalább 1 elemű a lista
    PE := Fej; P := Elem[Fej].Köv;
```

algo V. EA.

ciklus míg (P<> Végjel) És (Elem[P].Érték < Adat)

PE := P; P := Elem[P].Köv;

C vége

Ezt az elemet találjuk meg, amelyik elé kell fűzni, de csak úgy tudjuk, hogy MÖGÉ.
Ezért kell a PE.

ListaFűzRend := ListaFűzElemUtán(PE, Adat);

H vége;

F vége;

— o —

Függvény ListaTörlRend (Fej: Mutatótípus; Adat: Érték típus): logikai;

Változó PE, P: Mutatótípus;

Algoritmus

Ha (Fej <> Végjel) Akkor

Ha Elem[Fej].Érték = Adat Akkor

ListaTörlRend := ListaTörlElőle(Fej);

Különben

PE := Fej; P := Elem[Fej].Köv;

ciklus míg (P <> végjel) És (Elem[P].Érték < Adat)

PE := P; P := Elem[P].Köv;

C vége;

Ha (P <> Végjel) És (Elem[P].Érték = Adat) Akkor

ListaTörlRend := ListaTörlElemUtán(PE);

Különben

ListaTörölRend := Hamis;

Hvége ;

hvége ;

különben

ListaTörölRend := hamis;

hvége;

Fvége;

X.8.

4. előadás

F1 F2

listák => dinamikus és homogén

az elemek sorozatot alkotnak.

Karaktersorozat, verem, tömb, lista => alkalmas sorozatok tárolására

A1 A2

ciklus i := 1..n

c := A1[i]; A1[i] := A2[i]; A2[i] := c;

(c → elemtípus)

ciklus vége

Lehet: c := A1, A1 := A2, A2 := C

(c → vektortípus) (A tömböket mozgatjuk)

hogy lehet 2 lista tartalmát felcserélni?

c := F_1; F1 := F2; F2 := C;

3×4byte-os értékadás (csak a mutatók)