Binarno stablo (BinaryTree)

Binarno stablo T je konačan skup podataka istog tipa (čvorova) koji je ili prazan ili ima istaknuti čvor (korijen), a ostali čvorovi su podijeljeni u dva podskupa T_L i T_R od kojih svaki ima strukturu binarnog stabla. Stablo T_L zovemo lijevo podstablo, stablo T_R desno podstablo.

Nazivi su slični onima kod uređenih stabala, a razlika je u tome što: (i) binarno stablo može biti prazno, a uređeno ne može, (ii) u binarnom stablu kod roditelja s jednim djetetom znamo je li dijete lijevo ili desno, dok u uređenom stablu za jedino dijete ne postoji takva informacija.

a.t.p. BinaryTree

```
node                         . . .
labeltype                    . . .
BinaryTree                   . . .
BiMakeNull(&T)               . . .
BiEmpty(T)                   . . .
BiCreate(l,TL,TR,&T)         . . .
BiLeftSubtree(T,&TL)         . . .
BiRightSubtree(T,&TR)        . . .
BiInsertLeftChild(l,i,&T)    . . .
BiInsertRightChild(l,i,&T)   . . .
BiDelete(i,&T)               . . .
BiRoot(T)                    . . .
BiLeftChild(i,T)             . . .
BiRightChild(i,T)            . . .
BiParent(i,T)                . . .
BiLabel(i,T)                 . . .
BiChangeLabel(l,i,&T)        . . .
```

Zadatak za DZ

Napišite funkciju `int Visina(BinaryTree T)` koja vraća visinu binarnog stabla T: (a) koristeći se funkcijama `BiLeftSubtree` i `BiRightSubtree`, (b) ne koristeći se tim funkcijama.

Zadatak za DZ

Napišite funkciju `int Brat(BinaryTree B)` koja vraća broj onih čvorova koji: nisu listovi, njihov brat ima jedno dijete i imaju barem pet potomaka. Napišite funkciju `int Brat2(BinaryTree B)` koja vraća najveću oznaku među svim takvim čvorovima.

Zadatak 4.

Dokažite sljedeće tvrdnje.

(a) U binarnom stablu najveći broj čvorova na nivou i je 2^i.

(b) Najveći ukupan broj čvorova u binarnom stablu visine k je $2^{k+1} - 1$.

(c) Neka je n_0 broj listova, a n_2 broj čvorova s dvoje djece. Dokažite da je $n_0 = n_2 + 1$.

(d) Visina binarnog stabla s n čvorova je $\geq \lceil \log_2(n+1) \rceil - 1$.

(e) Visina binarnog stabla s n_0 listova je $\geq \lceil \log_2(n_0) \rceil$.

Rješenje.

Zadatak 5.

Binarno stablo na slici opisuje građu aritmetičkog izraza. Ispišite INORDER, PREORDER i POSTORDER obilazak tog stabla.

Rješenje.

Zadatak 6.

Dokažite da je binarno stablo u kojem svi čvorovi čuvaju različite informacije jednoznačno određeno obilascima PREORDER i INORDER.

Rješenje.

Pitanje za DZ

Što ako su zadani: (a) PREORDER i POSTORDER? (b) INORDER i POSTORDER?

Zadatak 7.

Reproducirajte binarno stablo ako je PREORDER = `ABDEFHJKLCGI` i INORDER = `EDFKJLHBAGIC`.

Rješenje.

Potpuno binarno stablo je stablo čijim se čvorovima mogu dati imena 0, 1, ..., N tako da za svaki i:

(i) lijevo dijete čvora i ima ime $2i + 1$ (osim ako je $2i + 1 > N$, onda ne postoji to dijete) i

(ii) desno dijete čvora i ima ime $2i + 2$ (osim ako je $2i + 2 > N$, onda ne postoji to dijete).

Napomena. Vrlo jednostavna implementacija potpunog binarnog stabla: u polju dimenzije $2^n - 1$ možemo prikazati sva potpuna binarna stabla s visinom $\leq n$. Ovaj prikaz možemo upotrijebiti i za obična binarna stabla, ali pritom puno mjesta može ostati neiskorišteno.

Zadatak 8.

Razradite implementaciju binarnog stabla pomoću polja i u polju prikažite stablo sa slike.

Rješenje. Stablo dopunimo lažnim čvorovima u koje napišemo oznaku '−' i prikažemo ga kao u definiciji potpunog binarnog stabla.

```
#define MAXLENGTH ...

typedef int node;

typedef struct {
    labeltype labels[MAXLENGTH];
} BinaryTree;

BiRoot(BinaryTree T)

BiLeftChild(node i, BinaryTree T)

BiRightChild(node i, BinaryTree T)

BiParent(node i, BinaryTree T)

BiLabel(node i, BinaryTree T)

BiChangeLabel(labeltype l, node i, BinaryTree* T)

BiInsertLeftChild(labeltype l, node i, BinaryTree* T)

BiInsertRightChild(labeltype l, node i, BinaryTree* T)

BiDelete(node i, BinaryTree* T)
```

U svakoj operaciji još treba paziti da li nekog čvora nema (to je ako u njemu piše '−' ili ako je njegov indeks veći od `MAXLENGTH-1`).

Napomena.

U poglavlju Općenite liste napravili smo funkciju `Merge` za spajanje dvije sortirane liste u jednu sortiranu listu. Spomenuli smo pritom problem određivanja redoslijeda spajanja više sortiranih lista u jednu (redoslijed spajanja u MERGE SORT algoritmu). Ovdje dajemo općeniti pristup tom problemu i rješenje.

Redoslijed spajanja opisujemo binarnim stablom kojem su listovi polazne liste (zapravo njihove veličine), unutrašnji čvorovi su rezultati pojedinih spajanja i korijen predstavlja konačnu listu. Uočimo da u ovom stablu svaki čvor ima 0 ili 2 djeteta, zbog toga kažemo da je ono 2-stablo. Cilj je pronaći raspored (tj. 2-stablo) za koji će ukupno vrijeme biti minimalno.

Primjer. $L_1 = 30$, $L_2 = 20$, $L_3 = 10$

Uočimo da elementi pojedine liste sudjeluju u onoliko operacija `Merge` koliko je dug put od pripadnog lista do korijena stabla (tj. nivo lista). Ukupno vrijeme je stoga (tzv. ponderirana duljina)

$$\sum_{\text{List } L_i} \omega_i \cdot p_i, \qquad \text{gdje je } \omega_i \text{ duljina liste } L_i \text{ i } p_i \text{ nivo lista } L_i.$$

Dakle, za zadane $\omega_1, \ldots, \omega_n$ trebamo naći stablo s n listova koji imaju oznake $\omega_1, \ldots, \omega_n$ i koje ima minimalnu ponderiranu duljinu. Ovaj problem rješava Huffmanov algoritam, koji pripada klasi pohlepnih algoritama.

HUFFMANOV ALGORITAM

Ulaz: niz (lista) 2-stabala koja se sastoje samo od korijena u kojima je informacija ω_i. Algoritam:

```
sve dok lista ima > 1 element {
    odaberi iz liste dva stabla ciji korijeni imaju najmanje oznake (neka su to T1 i T2);
    izbaci T1 i T2 iz liste;
    stvori novo 2-stablo T u cijem korijenu pise zbroj oznaka korijena T1 i T2 i kome je
        lijevo podstablo T1 i desno podstablo T2;
    ubaci T u listu;
}
```

Rješenje je jedino stablo ostalo u listi.

Zadatak 9.

Nađite optimalan plan spajanja šest sortiranih lista duljina $\omega_1 = 2$, $\omega_2 = 3$, $\omega_3 = 5$, $\omega_4 = 7$, $\omega_5 = 9$, $\omega_6 = 13$.

Rješenje.

Zadatak 10.

Dokažite da Huffmanov algoritam daje optimalan plan spajanja.

Rješenje. Matematičkom indukcijom po broju lista koje treba spojiti u jednu.

.

LaTeX-irala: Martina Stojić martina.stojic@gmail.com http://web.math.hr/~stojic 2014., listopad 2015.