Společná část - otázka č. 7

Spojové seznamy, lineární seznamy, obecné spojové struktury, stromy, jejich vlastnosti, binární stromy, implementace (A0B36PR1)

June 2, 2012

1 Pojmy

- Lineární spojová struktura (spojový seznam) - každý prvek má nanejvýš jednoho následníka
- Nelineární spojová struktura (strom) - každý prvek může mít více následníků
- Binární strom - každý prvek (uzel) má nanejvýš dva následníky

1.1 (Abstraktní) datový typ

Počet složek:

- neměnný = statický datový typ, počet položek je konstantní, pole, řetězec, třída
- proměnný = dynamický datový typ, počet složek je proměnný, mezi operace patří vložení, odebrání určitého prvku

Typ položek, dat:

- homogenní = všechny položky stejného typu
- nehomogenní = různého typu

Existence bezprostředního následníka

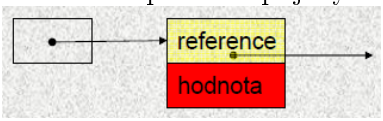
- lineární = existuje [např. pole, fronta, seznam,...]
- nelineární = neexistuje [strom, tabulka,...]

2 Spojové seznamy

Lineární seznam (také lineární spojový seznam) je dynamická datová struktura, vzdáleně podobná poli (umožňuje uchovat velké množství hodnot ale jiným způsobem), obsahující jednu a více datových položek (struktur) stejného typu, které jsou navzájem lineárně provázany vzájemnými odkazy pomocí ukazatelů nebo referencí. Aby byl seznam lineární, nesmí existovat cykly ve vzájemných odkazech.

Lineární seznamy mohou existovat jednosměrné a obousměrné. V jednosměrném seznamu odkazuje každá položka na položku následující a v obousměrném seznamu odkazuje položka na následující i předcházející položky. Zavádí se také ukazatel nebo reference na aktuální (vybraný) prvek seznamu. Na konci (a začátku) seznamu musí být definována zarážka označující konec seznamu. Pokud vytvoříme cyklus tak, že konec seznamu navážeme na jeho počátek, jedná se o kruhový seznam.

Základním prvkem spojových struktur je dvojice:



- reference - odkaz(y) na další prvek spojové struktury, definuje operátor new – znázorňujeme šipkou
- hodnota – informace libovolného typu

Nejjednodušší typ spojové struktury – jednosměrné spojové seznamy.

Spojové struktury jsou mocným implementačním prostředkem pro ADT, viz další přednáška o ADT a předmět Algoritmizace:

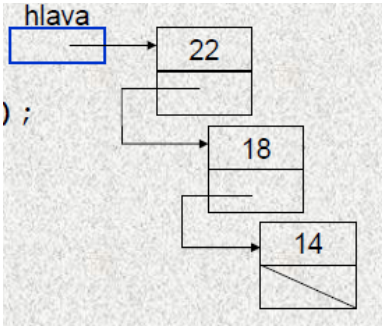
1. – Fronty
2. – Zásobníky
3. – Stromy
4. – Grafy

```
Prvek hlava = null;
int cislo= 14;
while (cislo<26) {

    hlava = new Prvek(cislo,hlava);
    cislo += 4;
```

```
}
```

Tento kód odpovídá struktuře (jednosměrný seznam s odkazem pouze na hlavu):



Jednosměrný seznam lze vylepšit, že se udržuje reference i na poslední, nebo volný prvek.

```
class Prvek {

    int hodnota;
    Prvek dalsi;
    // popr. dalsi konstruktory, treba prazdny, co nastavi vse na null
    public Prvek (int h, Prvek p) {
        hodnota = h;
        dalsi = p;
    }

}
public class Seznam {

    Prvek prvni;
    Prvek volny;
    public Seznam () {
        prvni = new Prvek(0, null);
        volny = prvni;
    }

    public void vlozNaKonec(int x) {
        volny.hodn = x;
        volny.dalsi = new Prvek();
        volny = volny.dalsi;
    }
    public void vypis() {
        Prvek pom = prvni;
        while (pom!=volny) {
            System.out.print(pom.hodn+" ");
            pom = pom.dalsi;
```

```
        }
        System.out.println();
    }
    public void vlozNaZacatek(int x) {
        prvni = new Prvek(x, prvni);
    }
    public boolean jePrvkem1(int x) {
        Prvek pom = prvni;
        while (pom.hodn!=x && pom!=volny)
            pom = pom.dalsi;
        return pom!=volny; }
} // konec tridy Seznam
// volani pr.
Seznam s = new Seznam();
s.vlozNaKonec(6);
s.vlozNaKonec(7);
s.vypis();
```

3 Stromy

V informatice je strom široce využívanou datovou strukturou, která představuje stromovou strukturu s propojenými uzly.

Pojmy:

1. „Cesta“ k nějakému uzlu je definována jako posloupnost všech uzlů od kořene k uzlu.
2. „Délka cesty“ je rovna počtu hran, které cesta obsahuje, tedy počtu uzlů posloupnosti – 1 (mínus jedna).
3. „Hloubka uzlu“ je definována jako délka cesty od kořene k uzlu. Prvky se stejnou hloubkou jsou na „téže úrovni“.
4. „Výška stromu“ je rovna hodnotě maximální hloubky uzlu, se označuje též za „hloubku stromu“.
5. „Šířkou stromu“ je počet uzlů na stejné úrovni.
6. Strom má „nejmenší výšku“ právě tehdy, když na všech úrovních (s možnou výjimkou té poslední) má tato struktura plný počet uzlů. Úroveň všech listů je stejná nebo se liší maximálně o 1.

3.1 Uspořádanost

„Uspořádaný“ nebo také „seřazený strom“ je takový strom, ve kterém jsou všichni přímí potomci každého uzlu seřazeni. Tudíž, pokud uzel má n dětí, lze určit prvního přímého potomka, druhého přímého potomka, až n-tého přímého potomka. U „neuspořádaného stromu“ se jedná o strom v čistě strukturálním smyslu. To znamená, že pro daný uzel nejsou uspořádáni potomci.

3.2 Vyvážený strom

„Vyvážený strom“ je takový strom, který má uzly rovnoměrně rozložené, tedy má nejmenší výšku. Ideální situace je taková, kdy má strom v každé hladině, kromě poslední, maximální počet uzlů, a v poslední hladině má uzly co nejvíce vlevo.

3.3 Procházení stromu (podrobněji pravděpodobně v ALG)

3.3.1 Do šířky

Procházením „stromu do šířky“ (anglicky „level-order“) se rozumí procházení stromem po vrstvách úrovní (tzn. po hladinách).

3.3.2 Do hloubky

Procházení začíná v kořeni stromu a postupuje se vždy na potomky daného vrcholu. Procházení končí, když v žádné větvi (tj. v žádném podstromu) již není následník. Podle pořadí, ve kterém se prochází uzly uspořádaného stromu, se rozlišují tři základní metody:

3.3.2.1 PREORDER

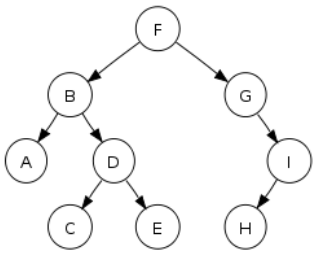
1. proveď akci
2. projdi levý podstrom
3. projdi pravý podstrom

3.3.2.2 INORDER

1. projdi levý podstrom
2. proveď akci
3. projdi pravý podstrom

3.3.2.3 POSTORDER

1. projdi levý podstrom
2. projdi pravý podstrom
3. proveď akci



Výsledky způsobů procházení v binárním vyhledávacím stromu,

N = navštívený uzel, **L** = levý, **R** = pravý

- ***Preorder*** (NLR): F, B, A, D, C, E, G, I, H
- ***Inorder*** (LNR): A, B, C, D, E, F, G, H, I
- ***Postorder*** (LRN): A, C, E, D, B, H, I, G, F
- Procházení do šířky (*po vrstvách*) ***Level-order***: F, B, G, A, D, I, C, E, H

3.4 Binární stromy

Každý prvek (uzel) má nanejvýš dva následníky (obecně lze prvkům stromu říkat také node, z ang., čteno [nouda]). Binární strom obsahuje uzly, které mají nejvíce dva syny.

Důležité pojmy:

1. kořen stromu - nejvyšší prvek, nemá rodiče
2. levý podstrom - strom levých potomků
3. pravý podstrom - pravých
4. vnitřní uzel - prvek, který má potomky (často se sem nepočítá kořen, někdy ano)
5. list - prvek, který již nemá potomky

U binarního stromu rozlišujeme další pojmy:

- Plný binární strom - všechny jeho listy jsou ve stejné hloubce.
- Úplný binární strom - každý vnitřní uzel má dva syny.
- Vyvážený binární strom - hloubka listů se od sebe liší maximálně o jedna.

3.4.1 Trochu matiky

- h – hloubka stromu,
- n – počet uzlů
- n0 – počet listů
- n2 – počet vnitřních uzlů

Úplný binární strom:
minimální počet uzlů: $n = 2^h + 1$
maximální počet uzlů: $n{=}2^{h+1} - 1$
počet listů: n/2 (zaokrouhleno nahoru)

4 Abstraktní datové typy

Tato část předmětu PRG2 není napsána v zadání u této otázky a je součástí ALG (je pouze obsahem slajdů k PRG2). Proto zde není popsána.