Задачи за задължителна самоподготовка

по

Структури от данни и програмиране

Двоични дървета 2

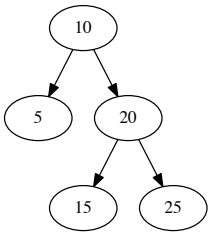
email: kalin@fmi.uni-sofia.bg

31 октомври 2017 г.

1. Да се реализира метод `vector<T> BTree<T>::listLeaves ()` намиращ списък със стойностите на листата на дървото.

2. Да се дефинира метод `string BTree<T>::findTrace (const T& x)`. Ако `x` е елемент на дървото, функцията да връща следата на `x` (според дефиницията на "следа", обсъдена на лекции). Ако `x` не е елемент на дървото, функцията да връща низа "_".

   Пример: За дървото от фигура 1, следата на елемента със стойност 25 е "RR".



Фигура 1. Двоично наредено дърво

3. Да се дефинира метод `void BTree<T>::prettyPrint ()`, отпечатващ дървото на конзолата по следния начин: (1) всеки наследник е

вдясно от родителя си, (2) елементите на еднакво ниво в дървото се отпечатват на еднаква колона от екрана, (3) десните наследници са на предишен ред от родителя си и (4) левите наследни са следващ ред спрямо родителя си.

Например, дървото от Фигура 1 би изглеждало по следния начин (включени са номерата на редовете на конзолата):

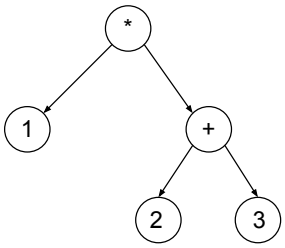
```
1:         25
2:    20
3:         15
4: 15
5:    5
```

4. Нека е даден израз, построен по правилата на следната грматика:

```
<expression> ::= <digit> | (<expression><operator><expresson>)
<digit> ::= 1..9
<operator> ::= + | - | * | /
```

Да се реалзира метод `void BTree<char>::parseExpression (string s)`, който по правилно построен израз, записан в низа `s`, създава двоично дърво от символи, представящо израза по следното правило:

- Ако изразът е от типа "`x`", където `x` е цифра, то съответното му дърво е листо със стойност символа `x`.
- Ако изразът е от типа "`(<израз 1><op><израз 2>)`", то съответното му дърво има като стойност на корена символа на съответния оператор, ляво поддърво, съответно на `<израз 1>` и дясно поддърво, съответно на `<израз 2>`.



Фигура 2. Дърво на израза `(1*(2+3))`.

Дървото на фигура 2 съответства на изарза `(1*(2+3))`.

5. Да се реализира метод `double BTree<char>::calculateExpresisonTree()`, който намира стойността на израз, построен от решението на предишната задача.

6. Да се дефинира оператор `T& BTree<T>::operator[](int i)`, който намира i-тият пореден елемент на дървото при обхождане коренляво-дясно.

   Пример: За дървото от фигура 1, елементът с пореден номер 0 е 15, с номер 1 е 5, с номер 2 е 20 и т.н.

7. Да се дефинира метод `vector<T> BTree<T>::level (int k)`, който намира и връща вектор, съдържащ стойностите на всички елементи на дървото, които са на ниво k (т.е. има път от корена до тях с дължина в брой върхове k).