ЛЕКЦИЯ 23
Двоичное дерево поиска. Продолжение.

1. Класс Дерево. Продолжение.

Программа №1.1. Класс Дерево

```
1   class Tree:
2       class Node: # Класс в классе можно делать, т.к. в классе описано свое
        пространство имен
3           def __init__(self, data):
4               self.parent = None
5               self.left = None
6               self.right = None
7               self.key = data
8       def __init__(self):
9           self.root = None
10      def find(self, data):
11          p = self.root
12          while p is not None and p.key != data: # Важна последовательность
        написаний условий, т.к. может быть ошибка при проверке ключа
13              if data > p.key:
14                  p = p.right
15              else:
16                  p = p.left
17          return p
18      def insert(self, data):
19          p = self.find(data) # Одно и то же число не может храниться дважды,
        как в множестве, но значения могут повторяться
20          if p is not None:
21              return
22          node = Tree.Node(data)
23          if self.root is None:
24              self.root = node
25              return
26          p = self.root
27          while True:
28              if data < p.key:
29                  if p.left is None:
30                      p.left = node
31                      node.parent = p
32                      break
33                  else:
34                      p = p.left
35              else:
36                  if p.right in None:
37                      p.right = node
```

```
38                      node.parent = p
39                      break
40                  else:
41                      p = p.right
```

2. Балансировка дерева

Двоичное дерево поиска сбалансированно, если для каждой его вершины высота левого и правого поддерева отличаются не более чем на единицу.

Инвариант: та вершина, которая левее других вершин, должна остаться левее всех вершин, т.е двигать вверх–вниз вершины можно, но влево–вправо двигать нельзя, иначе нарушится последовательность чисел.

Алгоритм балансировки подробно описан в википедии.

АВЛ–дерево — сбалансированное по высоте двоичное дерево поиска: для каждой его вершины высота её двух поддеревьев различается не более чем на 1.

Красно–чёрное дерево — это одно из самобалансирующихся двоичных деревьев поиска, гарантирующих логарифмический рост высоты дерева от числа узлов и быстро выполняющее основные операции дерева поиска: добавление, удаление и поиск узла. Сбалансированность достигается за счёт введения дополнительного атрибута узла дерева — «цвета». Этот атрибут может принимать одно из двух возможных значений — «чёрный» или «красный».

Свойства красно–черного дерева:

1. Узел либо красный, либо чёрный.
2. Корень — чёрный. (В других определениях это правило иногда опускается. Это правило слабо влияет на анализ, так как корень всегда может быть изменен с красного на чёрный, но не обязательно наоборот).
3. Все листья — чёрные.
4. Оба потомка каждого красного узла — чёрные.
5. Всякий простой путь от данного узла до любого листового узла, являющегося его потомком, содержит одинаковое число чёрных узлов.

Г. С. Демьянов, VK
С. С. Клявинек, VK