

二叉排序树

1. 目的：节省排序时间，节省检索时间。
(不是排成有序表) (二分搜索思想)

2. 性质：若非空，则左子树所有结点关键码小于根结点关键码，右子树所有结点关键码大于根结点关键码。

3. 算法：

① 检索 $\text{int search } (\text{PBinSearchTree } ptree, \text{keyType key}, \text{PBinSearchNode } *position)$

{ $p = *ptree; q = p;$ (q记录 p 的父结点)

while ($p \neq \text{NULL}$) {

$q = p;$

if ($p \rightarrow \text{key} == \text{key}$) { $*position = p;$ return 1; }

else if ($p \rightarrow \text{key} > \text{key}$) { $p = p \rightarrow llink;$ }

else $p = p \rightarrow rlink;$

}

$*position = q;$ return 0;

}

(与二分法检索相似)。

② 插入结点

$\text{int insert } (\text{PBinSearchTree } ptree, \text{keyType key})$

{ ① if ($\text{Search}(ptree, key, \&position) == 1$) return 1; (已经有这个)

② $p = \text{malloc } (\text{size of } (\text{BinSearchNode}));$

$p \rightarrow \text{key} = \text{key}; p \rightarrow llink = p \rightarrow rlink = \text{NULL};$ (新结点)

③ if ($position == \text{NULL}$) $*ptree = p;$ (原树为空)

else if ($position \rightarrow \text{key} > \text{key}$) $position \rightarrow llink = p;$

else $position \rightarrow rlink = p;$

} return 1;

}

③ 构造二叉排序树

返值参数

```

int createSearchTree (PBinSearchTree *ptree, SeqDictionary *dic)
{
    *ptree = NULL;
    for (i=0; i<dic->element[i]; i++)
        if (!insert (ptree, dic->element[i])) return 0; // 没有足够的空间
    return 1;
}

```

建树

④ 删除元素

描述：~~略~~ (1) 找到结点 P，待删除。（同时找到其父结点）

(2) 若 P 无左子树，则用 P 的右子女代替 P。

(3) 否则，在 P 的左子树中找到左子键码最大的 r（一直向右走），
用 r 结点代替 P，并用 r 原来的左子女代替 r。

~~代码： int delete (PBinSearchTree ptree, keyType key).~~

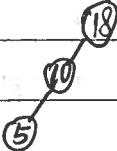
4. 图示 $k = \{18, 10, 5, 99, 41, 32\}$

①

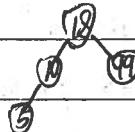
②



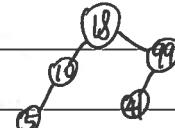
③



④



⑤



⑥

