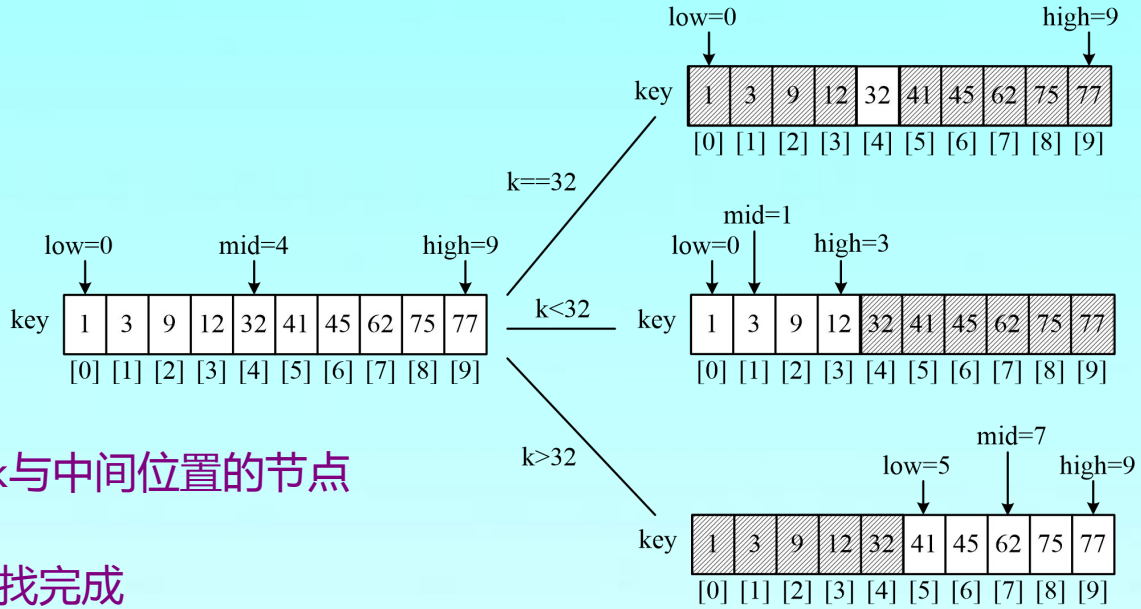
本节主题：

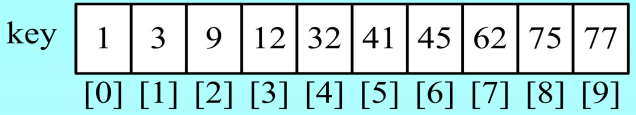
线性表上的折半查找

折半查找的思想

- 折半查找也称为二分查找
- 要求
  - 线性表中的节点，按关键字值的递增或递减顺序排列。
- 思路
  - 首先用要查找的关键字k与中间位置的节点的关键字比较
    - 若比较结果相等则查找完成
    - 若不相等，再根据k与该中间节点关键字的比较大小确定下一步查找哪个子表
  - 递归进行下去，直到找到满足条件的节点或者该线性表中没有这样的节点。



算法实现




```
int BinSearch(SeqList R,int n,KeyType k)
{
    int low=0,high=n-1,mid;
    while (low<=high)
    {
        mid=(low+high)/2;
        if (R[mid].key==k)
            return mid+1;
        if (R[mid].key>k)
            high=mid-1;
        else
            low=mid+1;
    }
    return 0;
}
```

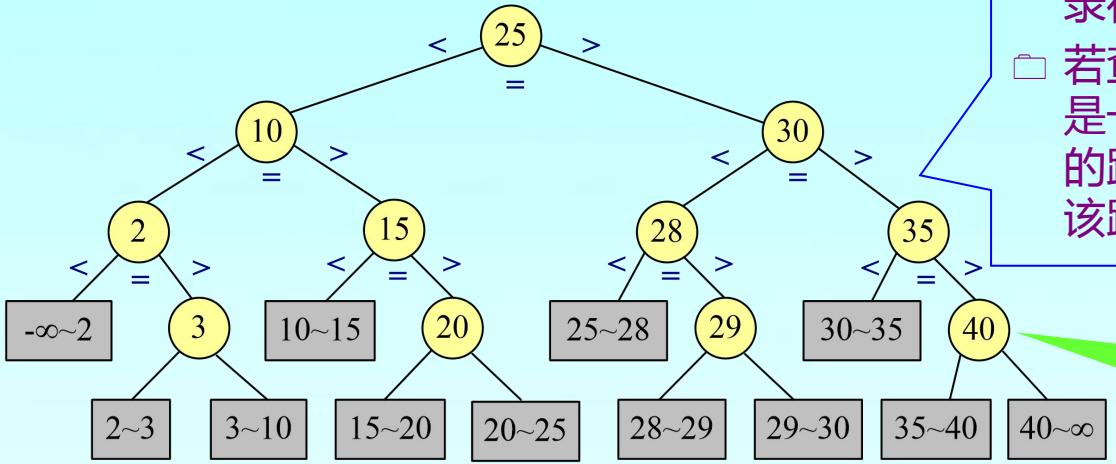
```
int BinSearch1(SeqList R,int low,int high,KeyType k)
{                                                  //递归算法
    int mid;
    if (low<=high)
    {
        mid=(low+high)/2;
        if (R[mid].key==k)
            return mid+1;
        if (R[mid].key>k)
            BinSearch1(R,low,mid-1,k)
        else
            BinSearch1(R,mid+1,high,k);
    }
    else
        return 0;
}
```

```
result = BinSearch1(R,0,n-1,x);
```

折半查找的性能分析基础

key	2	3	10	15	20	25	28	29	30	35	40
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]

- 折半查找的**判定树**或**比较树**
  - 将折半查找过程用二叉树来描述
  - 把当前查找区间的中间位置上的记录作为根；
  - 左子表和右子表中的记录分别作为根的左、右子树。



- 成功的折半查找过程，恰好是走了一条从根到被查记录的路径，比较次数恰为该记录在树中的层数。
- 若查找失败，则其比较过程是一条从根到某个外部节点的路径，所需的比较次数是该路径上内部节点的总数。

判定树实例

- 问题：对于给定n=11个数据元素的有序表{2,3,10,15,20,25,28,29,30,35,40}，采用折半查找

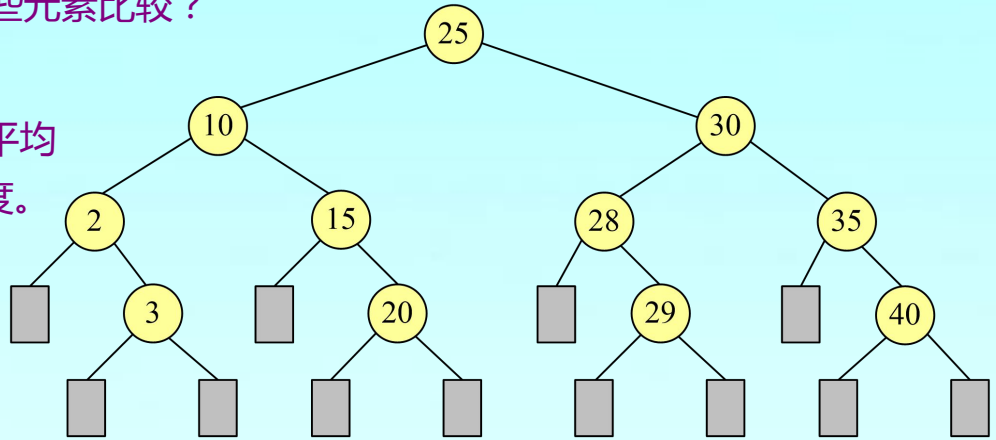
（1）若查找给定值为20的元素，将依次与表中哪些元素比较？

依次与表中25,10,15,20元素比较，共比较4次

（2）若查找给定值为26的元素，将依次与哪些元素比较？

依次与25,30,28元素比较，共比较3次。

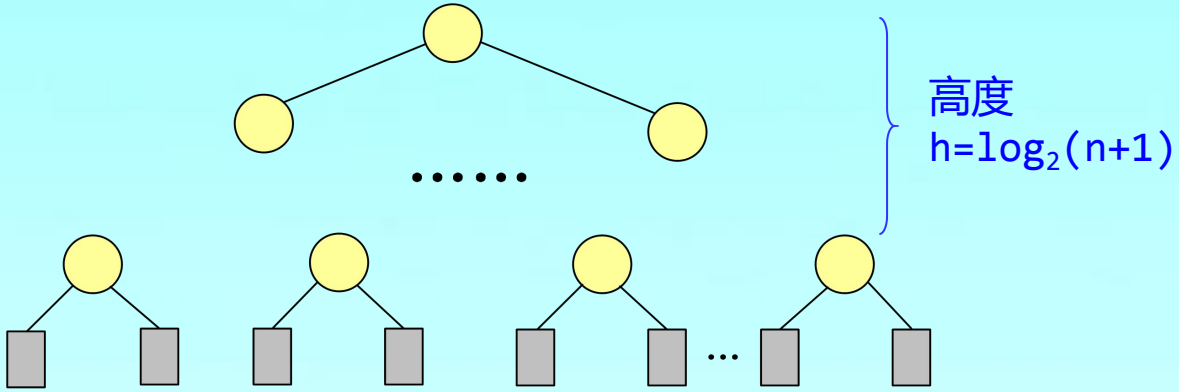
（3）假设元素等概率出现，求查找成功时的平均查找长度和查找不成功时的平均查找长度。



$$ASL_{succ}=\frac{1}{11}\times 1+\frac{2}{11}\times 2+\frac{4}{11}\times 3+\frac{4}{11}\times 4=\frac{1\times 1+2\times 2+4\times 3+4\times 4}{11}=3$$

$$ASL_{unsucc}=\frac{4\times 3+8\times 4}{12}=3.67$$

折半查找的性能分析(基于等概率假设)



对于n个元素的折半查找

- 成功时最多比较次数为：⌈log₂(n+1)⌉
- 不成功时关键字比较次数为：⌈log₂(n+1)⌉

——最坏性能和平均相当，均为O(log₂n)

- 借助二叉判定树，求得二分查找的平均查找长度
  - 设内部节点的总数为n=2^h-1，则判定树是高度为h=$\log_2(n+1)$的满二叉树（h不计外部节点）
  - 树中第i层上的记录个数为2^{i-1}，查找该层上的每个记录需要进行i次比较。

$$ASL_{bn}=\sum_{i=1}^{n}p_i c_i=\frac{1}{n}\sum_{i=1}^{h}2^{i-1}\times i$$
$$=\frac{n+1}{n}\times\log_2(n+1)-1$$
$$\approx\log_2(n+1)-1$$