2. 折半查找

有序表：如果顺序表中的记录按关键字值有序，

即：R[i].key≤R[i+1].key（或R[i].key≥R[i+1].key），

i=1,2,...,n-1，则称顺序表为有序表 。

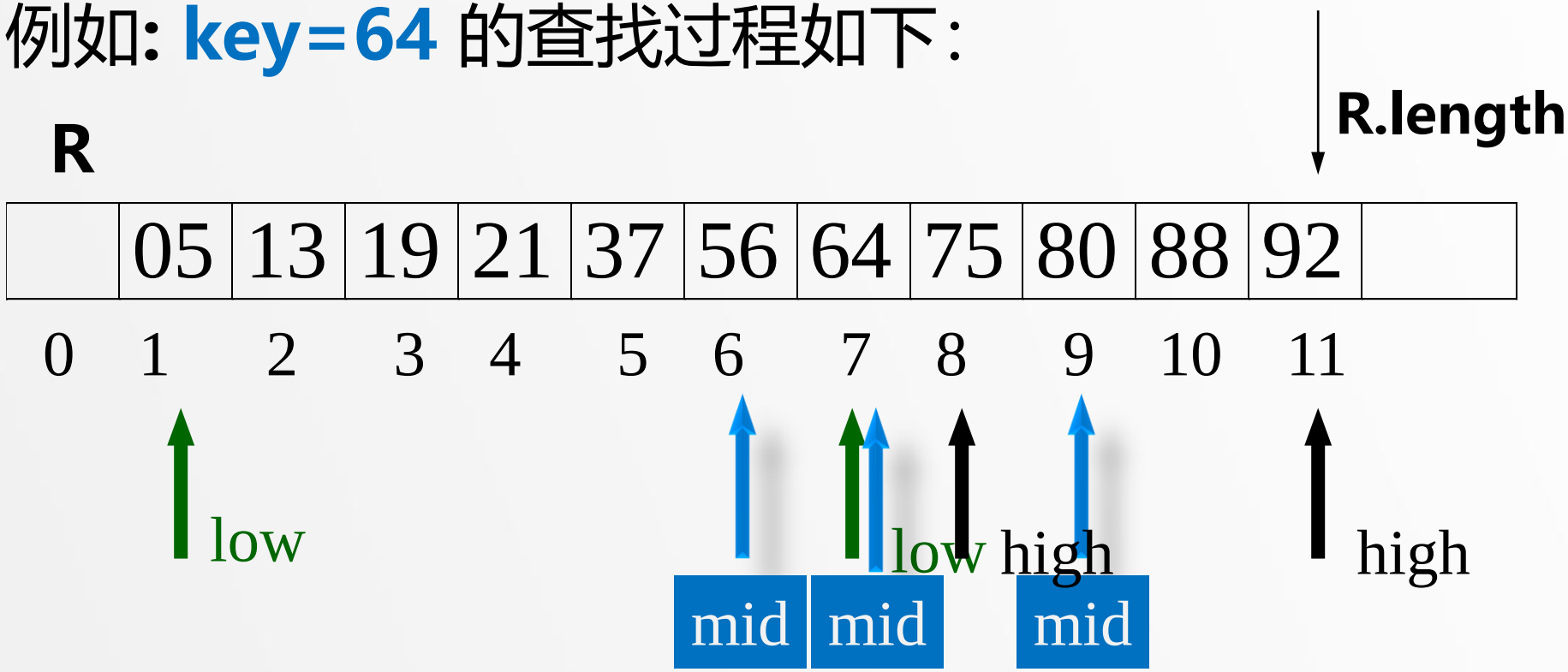
1 3 7 10 12 124　　有序表

1 3 7 13 12 124　　无序表

2. 折半查找 | 查找过程：

将待查关键字与有序表中间位置的记录进行比较，若相等，查找成功，若小于，则只可能在有序表的前半部分，若大于则只可能在有序表的后半部分，因此，经过一次比较，就将查找范围缩小一半，这样一直进行下去直到找到所需记录或记录不在查找表中。

例如: **key=64** 的查找过程如下:



low 指示查找区间的下界

high 指示查找区间的上界

mid = (low+high)/2

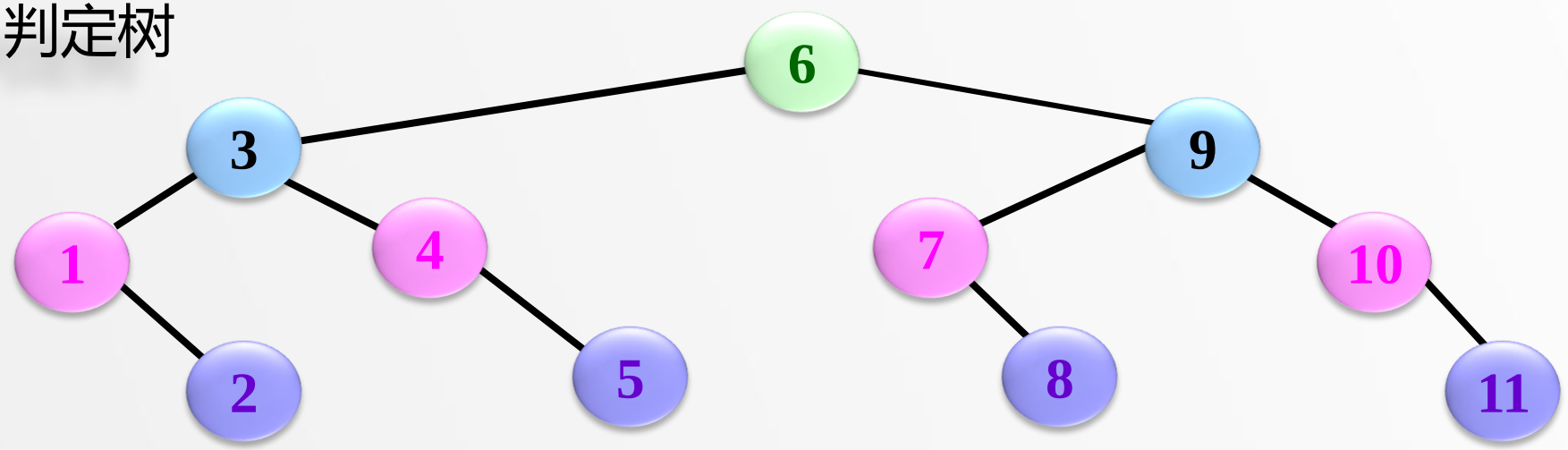
2. 折半查找 | 查找算法：

```
int BinarySearch(DataType SL[], KeyType key, int n){
/*在长度为n的有序表SL中折半查找其关键字等于key的记录*/
/*查找成功返回其在有序表中的位置，查找失败否返回0*/
  int low=1;
   int high=n;
   while(low<=high){
            mid=(low+high)/2;
            if(key = = SL[mid].key) {return mid;}
            else if( key>SL[mid].key) low=mid+1;
                        else high=mid-1;
            }
            return 0;
            }
```

折半查找的性能分析

i	1	2	3	4	5	6	7	8	9	10	11
Ci	3	4	2	3	4	1	3	4	2	3	4

判定树



3.折半查找性能分析

以深度为h的满二叉树为例，即： $n=2^h-1$ 并且查找概率相等，则

$$ASL=\frac{1}{n}\sum_{i=1}^{n}C_i=\frac{1}{n}\left[\sum_{j=1}^{h}j\times 2^{j-1}\right]=\frac{n+1}{n}\log_2(n+1)-1$$

当n>50时，可得近似结果

$$ASL\approx\log_2(n+1)-1$$

4.折半查找特点

折半查找的查找效率高；

平均查找性能和最坏性能相当接近；

折半查找要求查找表为**有序表**；

并且，折半查找只适用于**顺序存储结构。**