3.列表

有序列表：查找

邓俊辉

deng@tsinghua.edu.cn

❖ 在有序列表内节点p的n个（真）前驱中，找到不大于e的最后者

❖
```
template <typename T>
Posi(T) List<T>::search(T const & e, int n, Posi(T) p) const {
    while ( 0 <= n-- ) //对于p的最近的n个前驱，从右向左
        if ( ( ( p = p->pred ) -> data ) <= e ) break; //逐个比较
    return p; //直至命中、数值越界或范围越界后，返回查找终止的位置
} //最好O(1)，最坏O(n)；等概率时平均O(n)，正比于区间宽度
```

- `template <typename T>`

  `Posi(T) List<T>::search(T const & e, int n, Posi(T) p) const ;`

- 语义与向量相似，便于插入排序等后续操作：`insertA( search(e, r, p) , e )`

- 为何未能借助有序性提高查找效率？实现不当，还是根本不可能？

- 按照循位置访问的方式，物理存储地址与其逻辑次序无关

  依据秩的随机访问无法高效实现，而只能依据元素间的引用顺序访问