03-D2

列表

有序列表：查找

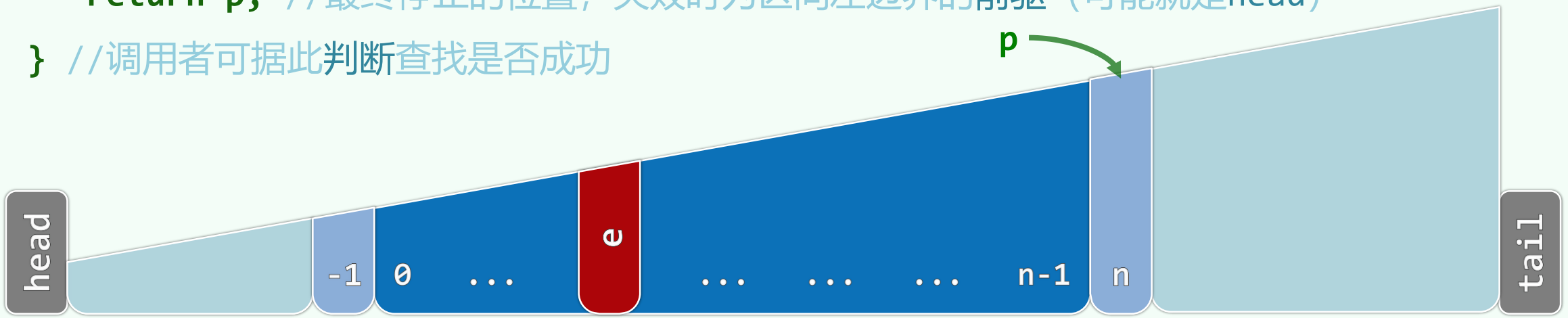
于是，他们急忙把自己的布袋卸在地上，各人打开自己的布袋。管家就搜查，
从最大的开始，查到最小的。那杯竟在便雅悯的布袋里搜出来了

种种念起，无不出于找。离了现前，向外去找，根本让我们直觉麻木的是这个找

邓俊辉
deng@tsinghua.edu.cn

search()

```
template <typename T> //在有序列表内节点p的n个真前驱中，找到不大于e的最靠后者
ListNodePosi<T> List<T>::search( T const & e, Rank n, ListNodePosi<T> p ) const {
   do { //初始有：0 <= n <= rank(p) < _size；此后，n总是等于p在查找区间内的秩
      p = p->pred; n--; //从右向左
   } while ( ( -1 != n ) && ( e < p->data ) ); //逐个比较，直至越界或命中
   return p; //最终停止的位置：失败时为区间左边界的前驱（可能就是head）
} //调用者可据此判断查找是否成功
```



性能 + 拓展

- 最好 $\mathcal{O}(1)$，最坏 $\mathcal{O}(n)$；等概率时平均 $\mathcal{O}(n)$，正比于区间宽度
- 语义与向量相似，便于插入排序等后续操作：`insert( 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