3.列表

无序列表：查找

邓俊辉

deng@tsinghua.edu.cn

❖ 在节点p（可能是trailer）的n个（真）前驱中，找到等于e的最后者

```
❖ template <typename T> //从外部调用时，0 <= n <= rank(p) < _size
  Posi(T) List<T>::find( T const & e, int n, Posi(T) p ) const { //O(n)
     while ( 0 < n-- ) //顺序查找：从右向左，逐个将p的前驱与e比对
        if ( e == ( p = p->pred )->data ) //这里假定类型T已重载操作符“==”
           return p; //直至命中或范围越界
     return NULL; //若越出左边界，意味着查找失败
  } //header的存在使得处理更为简洁
```

❖ 典型的调用模式：通过返回值判定

```
    x = L.find( e, n, p ) ?
          cout << x->data :
          cout << "not found";
```

❖ Posi(T) find(T const & e) const { //重载全局查找接口

```
    return find( e, _size, trailer ); //从内部调用时，rank(trailer) == _size
  }
```