02-C4

向量

无序向量：遍历

让他们每个人轮流到你的宝座下，同样诚恳地坦白他们的内心，
然后再看有没有一个人敢向你说：“我比这个人好。”

邓俊辉
deng@tsinghua.edu.cn

遍历

❖ **对向量中的每一元素，统一实施visit()操作** //如何指定visit()？如何将其传递到向量内部？

❖ template <typename T> //函数指针，只读或局部性修改

```
void Vector<T>::traverse( void ( * visit )( T & ) )
   { for ( Rank i = 0; i < _size; i++ ) visit( _elem[i] ); }
```

❖ template <typename T> template <typename VST> //函数对象，全局性修改更便捷

```
void Vector<T>::traverse( VST & visit )
   { for ( Rank i = 0; i < _size; i++ ) visit( _elem[i] ); }
```

实例：统一地将向量中的所有元素各自加一

❖ 先实现一个可使单个T类型元素加一的类（结构）

```
template <typename T> //假设T可直接递增或已重载操作符“++”
struct Increase //函数对象：通过重载操作符“()”实现
   { virtual void operator()( T & e ) { e++; } }; //加一
```

❖ 再将其作为参数传递给遍历算法

```
template <typename T> void increase( Vector<T> & V )
   { V.traverse( Increase<T>() ); } //即可以之作为基本操作，遍历向量
```

❖ 练习：模仿此例，实现统一的减一、加倍、求和等更多的遍历功能