Week3总结： STL标准容器的特点和典型的使用场景。

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 特点 | 使用场景 |
| 序  列  容  器  类  型 | vector | 最简单的序列类型，是数组的一种类表示。使用数组表示法来访问各个元素，查找较慢，但能很快索引。在尾部增减元素是固定时间，在头部增减元素是线性时间。强调通过随机访问进行快速访问。可反转。 | 除非其他类型容器的特殊优点能更好地满足程序要求，否则应默认使用vector。 |
| deque | 表示双端队列。类似vector容器，但从开始位置插入和删除元素的时间固定。对元素的随机访问和在序列中部执行插入、删除操作的速度比vector慢。 | 如果多数操作发生在序列的起始和结尾处，则考虑使用deque数据结构。 |
| list | 表示双向链表。在链表中任一位置进行插入或删除操作的时间都是固定的。强调元素的快速插入和删除。不支持数组表示法和随机访问。可反转。 | 适用于元素快速插入和删除的场景。 |
| forward\_list | 每个节点都只链接到下一个节点。比list更简单、紧凑，但功能更少。 | 单向链接的场景可用。 |
| queue | 适配器类。不允许随机访问或遍历队列，只能进行定义队列的基本操作。 | 不需要随机访问时可用。 |
| priority\_queue | 适配器类。支持的操作于queue相同，但最大的元素被移到队首，默认的底层类是vector。 | 需要最大元素在队首时可用。 |
| stack | 适配器类。不允许随机访问或遍历队列，只能进行定义栈的基本操作。 | 不需要随机访问时可用。 |
| array | 模板类。长度固定。 | （不是STL容器） |
| 关联容器 | set | 值与键的类型相同。可反转，可排序，键是唯一的，不能存储多个相同的值。 | 需要排序时可用。需要快速查找可用。不能制定元素插入位置。 |
| multiset | 与set类似，但同一个键可能和多个值相关联。 |
| map | 值与键类型不同，元素不可重复。 |
| multimap | 与map类似，但一个键可与多个值相关联。 |