函数原型:

set_intersection(iterator beg1, iterator end1, iterator beg2, iterator end2, iterator dest);

// 求两个集合的交集

// 注意:两个集合必须是有序序列

// dest 目标容器开始迭代器 //beg1容器1开始迭代器 // end2 容器2结束迭代器 // beg2 容器2开始迭代器 // end1 容器1结束迭代器

> vTarget.resize(min(v1.size(), v2.size())); //取两个里面较小的值给目标容器开辟空间

vector<int> vTarget;

/返回目标容器的最后一个元素的迭代器地址

vector<int>::iterator itEnd =

set_intersection(v1.begin(), v1.end(), v2.begin(), v2.end(), vTarget.begin());

vTarget.resize(min(v1.size(), //目标容器需要提前开辟空间 //最特殊情况 大容器包含小容器 v2. size())); 开辟空间 取小容器的size即可

vector<int>vTarget;

main 算法 取较小值

// 获取交集

Yector<int>::iterator itEnd= set_intersection(v1.begin(), v1.end(), v2.end(), vTarget.begin());

总结:

- 求交集的两个集合必须的有序序列
- 目标容器开辟空间需要从两个容器中取小值
- set_intersection返回值既是交集中最后一个元素的位置