1. STL概论
   1. STL六大组件
      1. 容器、算法、迭代器、仿函数、适配器、空间配置器
   2. STL优点
      1. 内建在C++编译器中，不需要安装额外内容
      2. 不需要了解具体实现内容，只要熟练运用即可
      3. 高复用，可移植，高性能
   3. 容器划分
      1. 序列式容器
      2. 关联式容器 有个key起到索引作用
   4. 算法
      1. 质变算法
      2. 非质变算法
2. STL初识
   1. vector容器 vector<int>v
   2. vector<int>::iterator itBegin = v.begin(); //v.begin() 起始迭代器，指向容器中第一个数据
   3. vector<int>::iterator itEnd = v.end(); //v.end()；结束迭代器，指向的是容器中最后一个元素的下一个位置
   4. 利用算法遍历容器
      1. for\_each(v.begin(),v.end() , myPrint ) 头文件 algorithm
   5. 容器中存放自定义数据类型
   6. 容器中存放自定义数据类型指针
   7. 容器嵌套容器
3. string
   1. 构造、赋值
   2. []、at 字符存取
      1. [] 和 at区别？ []访问越界 直接挂掉，at抛出out\_of\_range异常
   3. 字符串拼接 += append
   4. 查找 find rfind(从右往左查找)
   5. 替换 replace
   6. 比较 compare
   7. 子串 substr
   8. 案例
      1. 求email中的用户名
      2. string str = "www.itcast.com.cn";
      3. 将 www itcast com cn 单词截取到 vector容器中
   9. 插入 insert
   10. 删除 erase
   11. string和const char\* 转换
       1. string 转 const char \* .c\_str();
       2. const char \* 转string 利用string 有参构造
       3. const char \*可以隐式转换为 string 反之不可以
   12. 小写转大写 toupper
   13. 大写转小写 tolower
4. vector容器
   1. 单端数组
   2. 动态数组，自动扩展内存，所谓动态扩展内存，并不是在原有空间后续进行扩展，而是找一个更大的内存空间，将原有数据拷贝到新空间下，并且释放原有空间
   3. 接口
      1. 构造、赋值
      2. 交换 swap
      3. 大小 size
      4. 是否为空 empty
      5. 重置大小 resize
         1. 如果重置的比原来大，有默认值填充新位置
         2. 如果重置的比原来小，超出的部分删除掉
      6. front 返回容器中第一个元素
      7. back 返回容器中最后一个元素
   4. 插入 insert （迭代器）
   5. 删除 erase （迭代器）
   6. 尾插 push\_back
   7. 尾删 pop\_back
   8. 清空 clear
   9. 案例1 ：巧用swap收缩内存
   10. 案例2: ： 巧用reserve 预留内存
   11. 逆序遍历 reverse\_iterator 非质变
   12. 判断容器的迭代器是否支持随机访问
5. deque容器
   1. 双端数组
   2. 可以对头部进行插入和删除操作，内部有中控器控制数据
   3. 接口
      1. 构造、赋值
      2. 交换 swap
      3. 大小 size
      4. 是否为空 empty
      5. 重置大小 resize
      6. front 返回容器中第一个元素
      7. back 返回容器中最后一个元素
      8. 插入 insert （迭代器）
      9. 删除 erase （迭代器）
      10. 头部插入 push\_front
      11. 头部删除 pop\_front
      12. 尾插 push\_back
      13. 尾删 pop\_back
      14. 清空 clear
   4. sort排序 sort(v.begin(),v.end(), 回调函数)