

6.12 《STL 标准模板库》 课堂笔记

本周作业

1. 1918 (用 map 方法)
2. wood.cpp

作业2 wood.cpp (set写)

博艾市有一个木材仓库，里面可以存储各种长度的木材，但是保证没有两个木材的长度是相同的。作为仓库负责人，你有时候会进货，有时候会出货，因此需要维护这个库存。有不超过 100000 条的操作：

进货，格式1：在仓库中放入一根长度为 Length(不超过 10^9) 的木材。如果已经有相同长度的木材那么输出Already Exist。

出货，格式2：从仓库中取出长度为 Length 的木材。如果没有刚好长度的木材，取出仓库中存在的和要求长度最接近的木材。如果有多根木材符合要求，取出比较短的一根。输出取出的木材长度。如果仓库是空的，输出Empty。

输入输出样例

输入 #1

7
1 1
1 5
1 3
2 3
2 3
2 3
2 3
输出 #1
3
1
5
Empty

上周作业答案

1. P1443

```
1. #include <iostream>
2. #include <iomanip>
3. #include <cstdio>
4. using namespace std;
5. int _changeX[8] = {-1, -2, -2, -1, 1, 2, 2, 1};
6. int _changeY[8] = {-2, -1, 1, 2, 2, 1, -1, -2};
7. struct foot {
8.     int x;
9.     int y ;
10.    int step;
11. } _fst[100010];
```

```

12. int _map[410][410];
13. int _n, _m, _hx, _hy;
14.
15. void ParseIn() {
16.     cin >> _n >> _m >> _hx >> _hy;
17. }
18. void Bfs() {
19.     int head = 1;
20.     int tail = 1;
21.     int curX = 0;
22.     int curY = 0;
23.     while(head <= tail) {
24.         for(int i = 0; i < 8; i++) {
25.             curX = _changeX[i] + _fst[head].x;
26.             curY = _changeY[i] + _fst[head].y;
27.             if(curX >= 1 && curX <= _n && curY >= 1 && curY <= _m
&& _map[curX][curY] == -1) {
28.                 tail++;
29.                 _fst[tail].x = curX;
30.                 _fst[tail].y = curY;
31.                 _fst[tail].step = _fst[head].step + 1;
32.                 _map[curX][curY] = _fst[tail].step;
33.             }
34.         }
35.         head++;
36.     }
37. }
38.
39. void Core () {
40.     for(int i = 1; i <= _n; i++) {
41.         for(int j = 1; j <= _m; j++) {
42.             _map[i][j] = -1;
43.         }
44.     }
45.     _map[_hx][_hy] = 0;
46.     _fst[1].x = _hx;
47.     _fst[1].y = _hy;
48.     _fst[1].step = 0;
49.     Bfs();
50. }
51.
52. void WriteOut() {
53.     for(int i = 1; i <= _n; i++) {
54.         for(int j = 1; j <= _m; j++) {

```

```

55.         printf("%-5d", _map[i][j]);
56.     }
57.     cout << endl;
58. }
59. }
60.
61. int main () {
62.     ParseIn();
63.     Core();
64.     WriteOut();
65.     return 0;
66. }
67.

```

2. P2858

```

1. #include <bits/stdc++.h>
2. using namespace std;
3. int n;
4. int a[2110],sum[2110],dp[2110][2110];
5. int main()
6. {
7.     cin >> n;
8.     for(int i=1;i<=n;i++)
9.     {
10.         cin >> a[i];
11.         sum[i]=sum[i-1]+a[i];
12.     }
13.     for(int i=n;i>=1;i--)
14.     {
15.         for(int j=i;j<=n;j++)
16.         {
17.             dp[i][j]=max(dp[i+1][j],dp[i][j-1]);
18.             dp[i][j]+=(sum[j]-sum[i-1]);
19.         }
20.     }
21.     cout << dp[1][n] << endl;
22.     return 0;
23. }
24.

```

课堂内容

1. STL (Standard Template Library) 标准模板库

是一些“容器”的集合，这些“容器”有 list,vector,set,map 等，也是算法和其他一些组件的集合，是 C++ 的一部分，因此不用安装额外的库文件。

组件	描述
容器(containers)	容纳各种数据类型的数据结构 c++提供了各种不同类型的容器
迭代器(iterators)	可依次存取容器中元素
算法(algorithm)	算法作用于容器

在 C++ 标准中, STL 被组织为下面的 13 个头文件: <algorithm>、<deque>、<functional>、<iterator>、<vector>、<list>、<map>、<memory.h>、<numeric>、<queue>、<set>、<stack>和<utility>。

2. 迭代器

- 1) 定义：迭代器是一种检查容器内元素并遍历元素的数据类型。C++ 更趋向于使用迭代器而不是下标操作，因为标准库为每一种标准容器（如 vector）定义了一种迭代器类型，而只有少数容器（如 vector）支持下标操作访问容器元素。
- 2) 作用：用来指向、遍历、修改容器中元素的变量，类似指针，但不是指针。也可以理解为是具有部分指针特性的一种 STL 接口。
- 3) 迭代器常用操作：
 - 返回当前位置的元素值
 - ++ 迭代器移动到下一元素
 - == 和 != 判断两迭代器位置是否重合
 - = 为迭代器赋值

此外 begin(), 返回指向第一个元素的迭代器；end(), 返回一个指向最后一个元素之后的迭代器：

3. 关联容器

关联式容器也是用来存储数据的，与**序列式容器**不同的是，其里面存储的是<key, value>结构的键值对，在数据检索时比序列式容器效率更高。

4. set 集合

- 1) set 本质就是一个有序的排列,且不能有重复的元素，支持双向迭代器，不能直接修改 set 中的值。正确做法是：先删除，再加入
- 2) 常用函数：
 - insert(value)
 - erase(值)
 - erase(起始位置，结束位置)
 - erase(位置)
 - clear()

- **find(value)** 查找匹配的元素迭代器，若不存在，返回 **set.end()**
 - **count(value)** 如果返回 1 说明查找的数据存在，如果查找的数据不存在则返回零
5. **lower_bound()与 upper_bound()**
- 1) **lower_bound(begin,end,num)**: 从 begin 位置到 end-1 位置二分查找第一个大于或等于 num 的数字，找到返回该数字的地址，不存在则返回 end，作用数组时得到对应的下标。
 - 2) **upper_bound(begin,end,num)**: 从 begin 位置到 end-1 位置二分查找第一个大于 num 的数字，找到返回该数字的地址，不存在则返回 end，作用数组时得到对应的下标。
6. **map**
1. map 是一种关联容器，它提供一对一的数据处理能力。第一个称为关键字(key)，每个关键字只能在 map 中出现一次；第二个称为该关键字的值(value)；
 2. 特征：键(key)和值(value)可以是任意类型，对 key 的类型要求是必须支持<操作符。map 内部所有的数据都是有序的。默认是按 key 值递增存储查找和删除、插入的效率都是 logN；
 3. pair 作为键值对插入 map；
 4. 常见操作
 - 插入：数组方法和 insert 方法；
 - 删除：erase(iterator it) , erase(iterator first, iterator last), erase(key)
 - 查找：find(key) //如果找到了，返回当前元素位置，否则返回.end()
count(key) //存在返回 1，不存在返回 0
 5. map<typename1, typename2, greater<typename1>>表示按 key 类型降序排序，less 表示升序；
 6. 二分查找三件套：binary_search, lower_bound()与 upper_bound()
 7. 其他查找算法：
 - 查找容器元素 find(): 它用于查找等于某值的元素。它在迭代器区间[first,last)(闭开区间)上查找等于 value 值的元素,返回位置信息
 - 子序列搜索 search(): search(v1.begin(), v1.end(), v2.begin(), v2.end());返回第一个子序列的起始位置
 - 最后一个子序列搜索 find_end(): find_end(v1.begin(),v1.end(),v2.begin(),v2.end());在 V1 中要求的位置查找 V2 中要求的序列。
 - 统计等于某值的容器元素个数 count: count (v.begin(), v.end(), value)
 8. 字符串处理
- 1) 查找和截取
 - .find(子串 substr):查找字符串第一次出现的起始下标，否则返回-1；
 - .rfind(substr): 从后往前查找子串或字符出现的位置。
 - .find(子串 substr, i):从下标 i 开始，查找字符串第一次出现的下标，否则返回-1；
 - .find(子串 substr, i):从下标 i 开始，查找字符串第一次出现的下标，否则返回-1；
 - .substr(开始下标 i):从 i 下标开始，截取到最后的子字符串；
 - 2) 插入和删除
 - 插入：.insert(插入下标 i, 插入字符串 s):在字符串下标为 i 的位置插入一个字符串 s
 - 删除：string s1("Real Steel");
s1.erase(1, 3); //删除子串(起始位置 i, len)，此后 s1 = "R Steel"
s1.erase(5); //删除下标 5 及其后面的所有字符，此后 s1 = "R Ste"

课堂练习

1. pingpang.cpp

练习2:

pingpang.cpp

有一群人，打乒乓球比赛，两两捉对厮杀，每两个人之间最多打一场比赛。球赛的规则如下：

如果A打败了B，B又打败了C，而A与C之间没有进行过比赛，那么就认定，A一定能打败C。

如果A打败了B，B又打败了C，而且，C又打败了A，那么A、B、C三者都不可能成为冠军。

根据这个规则，无需循环较量，或许就能确定冠军。你的任务就是面对一群比赛选手，在经过了若干场厮杀之后，确定是否已经实际上产生了冠军。

Input

输入含有一些选手群，每群选手都以一个整数 n ($1 < n < 1000$)开头，后跟 n 对选手的比赛结果，比赛结果以一对选手名字（中间隔一空格）表示，前者战胜后者。

Output

对于每个选手群，若你判断出产生了冠军，则在一行中输出“Yes”，否则在一行中输出“No”。

样例输入1:

```
3
Alice Bob
Smith John
Alice Smith
样例输出1:
Yes
```

样例输入2:

```
5
a c
c d
d e
b e
a d
样例输出2:
No
```

代码:

```
1 #include <iostream>
2 #include <set>
3 using namespace std;
4 set<string> seLos;
5 set<string> seTol;
6 int n;
7 int main(){
8     cin >> n;
9     string Los, Win;
10    for ( int i = 0; i < n; i++){
11        cin >> Win >> Los;
12        seTol.insert(Win);
13        seTol.insert(Los);
14        seLos.insert(Los);
15    }
16    if(seTol.size() - seLos.size() == 1)
17        cout << "Yes";
18    else cout << "No";
19    return 0;
20 }
```

2. 计算数对个数

练习一（必须要用 set 做）

样例输入复制

题目描述

9

考虑一组 n 个不同的正整数 a_1, a_2, \dots, a_m ，它们的值在 1 到 1000000 之间。

5 12 7 10 9 1 2 3 11

给定一个整数 x 。写一个程序 sumx 计算这样的数对个数 (a_i, a_j) ， $1 \leq i < j \leq n$ 并且 $a_i + a_j = x$ 。

13

样例输出复制

输入

3

标准输入的第一行是一个整数 n ($1 \leq n \leq 1000000$)。第二行有 n 个整数表示元素。第三行是一个整数 x ($1 \leq x \leq 2000000$)。

提示

<set> 不同的和为 13 的数对是 (12, 1), (10, 3) 和 (2, 11)。

输出

输出一行包含一个整数表示这样的数对个数。

代码:

```
1 #include <iostream>
2 #include <set>
3 using namespace std;
4 set<int> se;
5 int n, x, count = 0;
6 int main(){
7     int curInt;
8     cin >> n;
9     for(int i = 1; i <= n; i++){
10         cin >> curInt;
11         se.insert(curInt);
12     }
13     cin >> x;
14     set<int>::iterator temp;
15     //题目中需要两个大于0的数相加为x,并且set为升序排列,因此循环截止条件到x的位置即可停止循环
16     for(set<int>::iterator it = se.begin(); it != se.lower_bound(x); it++){
17         //find()的时间复杂度是logn,此时遍历的时间复杂度为nlogn,而两层for循环遍历是n^2
18         temp = se.find( x - (*it) );
19         if( temp != se.end() )
20             count++;
21     }
22     cout << count / 2;
23     return 0;
24 }
```

3. shopping.cpp

练习3 shopping.cpp

T宝网进行优惠活动,每个商品当天第一个订单可以打75折。这天网站先后依次收到了N (N < 100000) 个订单,订单的数据形式是:商品名 价格(单价*数据),商品名是长度不超过20的字符串。请按商品名字典序输出当天的每种商品名和其售出的总价。

样例输入:

```
3
cup 100
paper 200
cup 200
```

样例输出:

```
cup 275
paper 150
```

降序怎么办?

代码:

```
1 #include <iostream>
2 #include <map>
3 using namespace std;
4 map<string, double> shop;
5 int n;
6 int main(){
7     string name;
8     double price;
9     cin >> n;
10    for(int i = 1; i <= n; i++){
11        cin >> name >> price;
12        //若找到count()的返回值为1,反之返回值为0,若第一次出现其返回值应该是0
13        if(!shop.count(name)){
14            shop[name] = price * 0.75;
15        }
16        else shop[name] += price;
17    }
18    for(map<string, double>::iterator it = shop.begin(); it != shop.end(); it++){
19        cout << (*it).first << " " << (*it).second << endl; //map输出key和value
20    }
21    return 0;
22 }
```



```

1 #include <iostream>
2 #include <map>
3 using namespace std;
4 map<string, string> stu;
5 int main(){
6     stu.insert(pair<string, string>("stu1", "zhuang1")); //map的插入方法
7     stu.insert(pair<string, string>("stu2", "zhuang2")); //map的插入方法
8     stu.insert(pair<string, string>("stu3", "zhuang3")); //map的插入方法
9
10    for(map<string, string>::iterator it = stu.end(); it != stu.begin(); it--){
11        if( it == stu.end() ){//如果要是到end(), 则需要跳过
12            continue;
13        }
14        cout << (*it).first << " " << (*it).second << endl;
15    }
16
17    map<string, string>::iterator it = stu.begin(); //begin()作为终止条件, 需要单独输出
18    cout << (*it).first << " " << (*it).second << endl;
19    return 0;
20 }

```

```

5 map<string, double, greater<string> > _dingDan;
6 int _n = 0;
7 int main () {
8     string name;
9     double value = 0;
10    cin >> _n;
11    for(int i = 1; i <= _n; i++) {
12        cin >> name >> value;
13        if(_dingDan.count(name) == 0) {
14            // _dingDan.insert(make_pair(name, value * 0.75));
15            _dingDan[name] = value * 0.75;
16        }
17        else {
18            _dingDan[name] += value;
19        }
20    }
21
22    for(map<string, double, greater<string> >::iterator it = _dingDan.begin(); it != _dingDan.end(); it++) {
23        cout << (*it).first << " " << it->second << endl;
24    }
25
26    return 0;
27 }

```

4. 最多书籍数量

综合练习1 exam.cpp (map写)

期末考试即将来临, 小T由于同时肩负了学习、竞赛、班团活动等多方面的任务, 一直没有时间好好整理他的课桌抽屉, 为了更好地复习, 小T首先要把课桌抽屉里的书分类整理好。小T的抽屉里堆着N本书, 每本书的封面上都印有学科名称, 学科名称用一个字符串表示, 如语文学科的书封面上都印有“chinese”。现在, 你的任务是帮助小T找出哪个学科的书最多

输入

第一行包含一个自然数N (0 < N ≤ 10000) 表示抽屉中书的总数。接下来N行每行包含一本书的学科名称, 学科名称是一个长度不超过15的由小写英文字母组成的字符串。

输出

仅有一行包含一个字符串, 表示最多的那种书的学科名称。数据保证答案一定是唯一的。

样例输入

```

5
english
chinese
physics
chinese
chinese

```

样例输出

```

chinese

```

代码


```

1 #include <bits/stdc++.h>
2 using namespace std;
3 map<string, int> BookList;
4 int n;
5 int main(){
6     string curString, maxClass;
7     int maxNum = 0;
8     cin >> n;
9     for(int i = 1; i <= n; i++){
10         cin >> curString;
11         if(BookList.count(curString) == 0){
12             BookList[curString] = 1;
13         }
14         else BookList[curString] += 1;
15     }
16     for( map<string, int>::iterator it = BookList.begin(); it != BookList.end(); it++){
17         if( (*it).second > maxNum){
18             maxClass = (*it).first;
19             maxNum = (*it).second;
20         }
21     }
22     cout << maxClass;
23     return 0;
24 }

```

5. 得分排名

综合练习2 score.cpp

小x很关心自己在学校的表现。
班主任手上有一本“个人得分记录本”，如果一位同学表现好就会加分，表现差则会扣分。
学期结束，每位同学都得知道了自己的个人得分。小x想知道其他同学情况如何，但由于排名不公布，他只好一个个去问班里的其他同学。
现在，小x手上有班里共N位同学的个人得分，他想知道每位同学的排名（得分相同则排名相同，见样例），可并不知道该如何计算，希望你帮帮他。

输入

第一行包含一个整数N。
接下来N行，第i行包含一个整数Ai，表示第i位同学的得分。

输出

N行，第i行包含一个整数，表示第i位同学的排名。

提示

数据范围

- 对于30%的数据， $N \leq 10$ 。
- 对于60%的数据， $N \leq 1000$ 。
- 对于100%的数据， $1 \leq N \leq 100000$ ， $0 \leq A_i \leq 100000$ 。

样例输入	5
	95
	100
	99
	99
	96
样例输出	5
	1
	2
	2
	4

代码：

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 map<int, int> score;
4 int scoreList1[100001], scoreList2[100001];
5 int n;
6 bool cmp(int a, int b){
7     return a > b;
8 }
9 int main(){
10     int curInt;
11     cin >> n;
12     for(int i = 1; i <= n; i++){
13         cin >> curInt;
14         scoreList1[i] = curInt;
15         scoreList2[i] = curInt;
16     }
17     sort(scoreList1 + 1, scoreList1 + n + 1, cmp);
18     for(int i = 1; i <= n; i++){
19         score.insert(pair<int, int>(scoreList1[i], i));
20     }
21     for(int i = 1; i <= n; i++){
22         cout << (*score.find(scoreList2[i])).second << endl;
23     }
24     return 0;
25 }

```

6.

例1、
给出一串数以及一个数字 C，要求计算出所有 $A - B = C$ 的数对的个数（不同位置的数字一样的数对算不同的数对）。

输入格式

输入共两行。

第一行，两个整数 N,C。

第二行，N 个整数，作为要求处理的那串数。

输出格式

一行，表示该串数中包含的满足 $A - B = C$ 的数对的个数。

输入 #1

4 1

1 1 2 3

输出 #1

3

说明/提示

对于 75% 的数据， $1 \leq N \leq 2000$ 。

对于 100% 的数据， $1 \leq N \leq 2 \times 10^5$

保证所有输入数据都在 32 位带符号整数范围内。

代码：

```
1 #include <iostream>
2 #include <algorithm>
3 using namespace std;
4
5 int _n = 0;
6 int _c = 0;
7 int _list[200010];
8 long long _cnt = 0;
9 int main() {
10     cin >> _n >> _c;
11
12     for(int i = 1; i <= _n; i++) {
13         cin >> _list[i];
14     }
15
16     sort(_list + 1, _list + _n + 1);
17
18     for(int i = 1; i <= _n; i++) {
19         if(lower_bound(_list + 1, _list + _n + 1, _list[i] - _c) - _list == _n) { //没有查到
20             continue;
21         }
22         else {
23             _cnt += upper_bound(_list + 1, _list + _n + 1, _list[i] - _c) - lower_bound(_list + 1, _list + _n + 1, _list[i] - _c);
24         }
25     }
26
27     cout << _cnt;
28     return 0;
29 }
```

7.

例2maximum.cpp

设有n个整数($3 \leq n \leq 100$)，将这些整数拼接起来，可以形成一个最大的整数。

例如：n=3，三个整数分别为21 7 34，拼接后最大的整数为：73421

再比如：n=3，三个整数分别是1 10 110，拼接后最大的整数是：111010

输入

第一行一个整数n，表示有n个整数。

第二行n个整数，数与数之间用一个空格分隔。

输出

输出到屏幕。一个拼接后的最大的整数。

样例输入

3

21 7 34

样例输出

73421

代码:

```
1  #include <iostream>
2  #include <algorithm>
3  using namespace std;
4
5  int _n = 0;
6  //样例输入
7  //3
8  //21 7 34
9  //样例输出
10 //73421
11 string _strList[110];
12
13 bool Cmp(string one, string two) {
14     return one + two > two + one;
15 }
16 int main () {
17     cin >> _n;
18     for(int i = 1; i <= _n; i++) {
19         cin >> _strList[i];
20     }
21
22     sort(_strList + 1, _strList + _n + 1, Cmp);
23
24     for(int i = 1; i <= _n; i++) {
25         cout << _strList[i];
26     }
27     return 0;
28 }
```