目录

[练习1.3 练习1.4 练习1.5 3](#_Toc6664429)

[练习1.6 4](#_Toc6664430)

[1.4.1 节练习 4](#_Toc6664431)

[1.4.2 节练习 5](#_Toc6664432)

[1.4.4 节练习 7](#_Toc6664433)

[1.4.4 节练习 8](#_Toc6664434)

[1.5 节练习 9](#_Toc6664435)

[第一章小结 11](#_Toc6664436)

[第二章 变量和基本类型 11](#_Toc6664437)

[练习2.3 11](#_Toc6664438)

[练习2.40 练习2.41 练习2.42 11](#_Toc6664439)

[第二章小结 14](#_Toc6664440)

[第三章 字符串、向量和数组 14](#_Toc6664441)

[3.2 节练习 14](#_Toc6664442)

[3.2.3 节练习 15](#_Toc6664443)

[3.3.2 节练习 16](#_Toc6664444)

[3.3.3 节练习 17](#_Toc6664445)

[3.4.1 节练习 20](#_Toc6664446)

[3.4.2 节练习 22](#_Toc6664447)

[3.5.2 节练习 24](#_Toc6664448)

[3.5.3 节练习 25](#_Toc6664449)

[3.5.5 节练习 27](#_Toc6664450)

[第三章小结 29](#_Toc6664451)

[第四章 表达式 29](#_Toc6664452)

[4.2节练习 29](#_Toc6664453)

[4.3节练习 30](#_Toc6664454)

[4.7节练习 31](#_Toc6664455)

[4.9节练习 32](#_Toc6664456)

[第五章 语句 33](#_Toc6664457)

[5.3.2节练习 33](#_Toc6664458)

[5.5.1节练习 34](#_Toc6664459)

[第五章小结 35](#_Toc6664460)

[第六章 函数 35](#_Toc6664461)

[6.1节练习 35](#_Toc6664462)

[6.2.1节练习 37](#_Toc6664463)

[6.2.3节练习 38](#_Toc6664464)

[6.7节练习 40](#_Toc6664465)

[第八章 IO类 41](#_Toc6664466)

[8.1.2 节练习 42](#_Toc6664467)

[8.2.1 节练习 43](#_Toc6664468)

[8.3.1 节练习 44](#_Toc6664469)

[8.3.2 节练习 46](#_Toc6664470)

[第八章小结 47](#_Toc6664471)

[第九章 顺序容器 47](#_Toc6664472)

[9.2节练习 48](#_Toc6664473)

[9.2.5节练习 49](#_Toc6664474)

[9.2.7节练习 50](#_Toc6664475)

[9.3.2节练习 51](#_Toc6664476)

[第十章 53](#_Toc6664477)

[10.1节练习 53](#_Toc6664478)

[10.4.1节练习 54](#_Toc6664479)

[第十章小结 55](#_Toc6664480)

[11.2.1节练习 55](#_Toc6664481)

[11.2.3节练习 57](#_Toc6664482)

[11.3.6节练习 58](#_Toc6664483)

[第十一章小结 60](#_Toc6664484)

第一章 开始

/\*\*\*

## 练习1.3 练习1.4 练习1.5

2019年3月22日22:26:13

by PK

\*\*\*\*/

#include <iostream>

#include <stdio.h>

int main(void)

{

/\*练习1.3

输出hello world

\*/

printf("hello C hello world! \r\n"); //C

std::cout << "hello C++ hello singledog! \r\n"<<std::endl;

//以上为练习1.3

/\*\* 练习1.4

计算两个输入数的加和乘

\*\*/

int num1,num2,res;

std::cout<<"please input two number \r\n"<<std::endl;

std::cin>>num1>>num2;

res = num1+num2;

std::cout<<"num1+num2 = "<<res<<std::endl;

res = num1\*num2;

std::cout<<"num1\*num2 = "<<res<<std::endl;

printf("\r\n\r\n");

/\*\*练习1.5

将所有的输出操作放在一条语句中

\*\*/

std::cout<<"the sum of num1 + num2 is "<<num1+num2<<" and result of num1\*num2 is "<< num1\*num2<<"\r\n"<<std::endl;

getchar();

while(1);

return 0;

}

## 练习1.6

程序段是不合法的，因为<<运算符的用法错误，<<左侧为流对象，右侧为流输出内容，流输出可以有多个对象，但是不能用分号将其隔开，否则编译器会认为语句已经结束

修改：

代码修改为

Std::cout<<”The sum of ”<<v1<<”and”<<v2<<”is”<<v1+v2<<std::endl;

程序输出v1 和v2的和及对应字符串

/\*\*\*\*\*\*\*\*\*\*88

\*\*\* 2019年3月24日14:55:32

## 1.4.1 节练习

while语句实现循环控制

练习1.9

练习1.0

练习1.11

\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

int main(void)

{

//练习1.9 求50到100的循环加和

int num = 50,sum = 0;

while(num<101)

{

sum += num;

num++;

}

std::cout<<"sum = "<<sum<<"\r\n"<<std::endl;//结果为3825

//练习1.10 递减

num = 10;

while(num>=0)

{

std::cout<<" "<<num<<" "<<std::endl;

num--;

}

//练习1.11 输入2个整数，输出二者之间的所有整数

int innum1,innum2;

std::cout<<" please input two increasing integer"<<std::endl;

std::cin>>innum1>>innum2;

if(innum1 > innum2)

{std::cout<<" input error! \r\n please input two increasing integer!"<<std::endl;}

else

{

num = innum1;

while(num<innum2)

{

std::cout<<num<<std::endl;

num++;

}

}

while(1);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*88

\*\*\*

2019年3月24日15:23:18

## 1.4.2 节练习

for循环控制流练习

练习1.12

练习1.13

练习1.14

\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

int main(void)

{

/\*\*\*

练习1.12

描述for循环的功能，并求sum值

\*\*\*/

int sum = 0;

for(int i = -100;i<=100;++i)

sum+=i;

std::cout<<"sum = "<<sum<<"\r\n"<<std::endl;

/\*\*\*\*

\*\* 功能：循环求整数-100到100的和 sum输出值 0

\*\*\*\*/

//练习1.13

//练习1.13.1 求50到100的循环加和 for实现

sum = 0;

for(int i=50;i<101;i++)

{

sum += i;

}

std::cout<<"sum of 50 to 100 inclusive is "<<sum<<"\r\n"<<std::endl;//结果为3825

//练习1.13.2 递减

int num = 10;

for(int i =10;i>=0;i--)

{

std::cout<<i<<std::endl;

}

//练习1.13.2 输入2个整数，输出二者之间的所有整数

int innum1,innum2;

std::cout<<" input two increasing integer"<<std::endl;

std::cin>>innum1>>innum2;

if(innum1 > innum2)

{std::cout<<" input error! u idiot! \r\n please input two increasing integer!"<<std::endl;}

else

{

for(int i = innum1;i<=innum2;i++)

std::cout<<i<<std::endl;

}

//

/\*\*\*\*\*\*\*\*\*\*\*\*\*

//练习1.14

在for循环中，循环控制变量的初始化和修改都放在语句头部分，形式较简洁，且特别适用于循环次数已知的情况

在while循环中，循环控制变量一般在循环前初始化，形式没有for语句简洁，但是适用于循环次数未知的情况

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

while(1);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* 2019年3月24日16:03:34

## 1.4.4 节练习

\*\* 要点：if语句

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

//程序功能：记录连续输入整数中重复出现的数并输出

int main(void)

{

int currVal =0,val = 0;

if(std::cin>>currVal) //输入不合法就不执行

{

int cnt = 1; //计数

while(std::cin>>val)

{

if(val == currVal)++cnt; //记录重复的值

else

{

std::cout<<currVal<<" occurs "<< cnt << " times " <<std::endl;

currVal = val;

cnt = 1;

}

}

std::cout<<currVal<<" occurs "<< cnt << " times " <<std::endl;

}

while(1);

return 0;

}

1.17：输入值都相等，程序会只输出一次 xx occurs x times 就结束；若值全不相等，程序输出xx occurs 1 times ，次数与输入的数的个数相同

/\*\*\*\*\*\*\*\*\*\*88

\*\*\*

2019年3月24日16:18:50

## 1.4.4 节练习

练习1.19

\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

int main(void)

{

//练习1.19 输入2个整数，输出二者之间的所有整数

//修正逻辑以使输入不必升序

int innum1,innum2;

std::cout<<" input two integer"<<std::endl;

std::cin>>innum1>>innum2;

if(innum1 > innum2)

{

int temp;

temp = innum2;

innum2 = innum1;

innum1 = temp;

}

for(int i = innum1;i<=innum2;i++)

std::cout<<i<<std::endl;

while(1);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*

2019年3月24日16:33:47

## 1.5 节练习

书店程序

练习1.21 1.22 1.23 1.24

关键：类

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include "Sales\_item.h"

#define work1\_23 //编译开关

int main(void)

{

//课本例程 & 练习1.21

//功能：计算两个类对应成员的和

#ifdef work1\_21

Sales\_item item1,item2;

std::cin>> item1>>item2;

if(item1.isbn() == item2.isbn())

{

std::cout<<item1+item2<<std::endl;

//return 0;

}

else

{

std::cerr << "Data must refer to same ISBN"<<std::endl;

//return -1;

}

//以上代码对应练习1.21 实际为课本例程

#endif

//练习1.22 读取多个具有相同isbn的销售记录，输出所有记录和

#ifdef work1\_22

Sales\_item item1,item2;

if (std::cin>> item1)

{

int cnt = 1; //计数

while(std::cin>> item2)

{

if(item1.isbn() == item2.isbn())

{

cnt ++;

item1 += item2;

}

else

std::cerr << "Data must refer to same ISBN"<<std::endl;

}

std::cout<<"total record is "<<cnt<<"\r\n"<<item1<<std::endl;

}

#endif

#ifdef work1\_23

//练习2.23与2.24

//此程序其实有bug，必须连续输入相同的条目才可以统计到相同的书本信息，而且输入和输出混在一起非常难看

Sales\_item item1,item2;

if (std::cin>> item1)

{

int cnt = 1; //计数

while(std::cin>> item2)

{

if(item1.isbn() == item2.isbn())

{

cnt ++;

item1 += item2;

}

else

{

std::cout<<"total record of "<< item1.isbn()<<" is "<<cnt<<"\r\n"<<item1<<std::endl;

item1 = item2;

cnt = 1;

}

}

std::cout<<"total record of "<< item1.isbn()<<" is "<<cnt<<"\r\n"<<item1<<std::endl;

}

#endif

while(1);

}

## 第一章小结

第一章其实没什么特别的内容，主要收获是认识了输入输出两个“流”，掌握了cin和cout的用法

# 第二章 变量和基本类型

/\*\*\*

\*\* 2019年3月26日19:32:03

## 练习2.3

\*

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

int main(void)

{

unsigned u = 10,u2 = 42; //输出

std::cout << u2 - u << std::endl; //30

std::cout << u-u2 << std::endl; //很大的一个数

int i = 10,i2 = 42;

std::cout<<i2-i<<std::endl; //30

std::cout<<i-i2<<std::endl; //-30

std::cout<<i-u<<std::endl; //0

std::cout<<u-i<<std::endl; //0

while(1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* 2019年3月26日20:02:38

## 练习2.40 练习2.41 练习2.42

\*\* 自行编写一个头文件，定义类，重写输出书本销售记录的程序

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include "Sales\_data.h"

using namespace std;

#define work1\_23

int main(void)

{

#ifdef work1\_21

Sales\_data item1,item2;

double avg;

cin >> item1.bookNo >> item1.units\_sold >>item1.price ;

item1.revenue = item1.units\_sold\*item1.price ;

cin >> item2.bookNo >> item2.units\_sold >> item2.price;

item2.revenue = item2.units\_sold\*item2.price ;

if(item1.bookNo == item2.bookNo)

{

item2.units\_sold += item1.units\_sold;

item2.revenue += item1.revenue;

avg = item2.revenue / item2.units\_sold;

std::cout<<item2.bookNo<< item2.units\_sold <<item2.revenue<<avg<<std::endl;

//return 0;

}

else

{

std::cerr << "Data must refer to same ISBN"<<std::endl;

//return -1;

}

//以上代码对应练习1.21 实际为课本例程

#endif

//练习1.22 读取多个具有相同isbn的销售记录，输出所有记录和

#ifdef work1\_22

Sales\_data item1,item2;

double avg;

if (cin >> item1.bookNo >> item1.units\_sold >>item1.price)

{

int cnt = 1; //计数

item1.revenue = item1.units\_sold\*item1.price ;

while(cin >> item2.bookNo >> item2.units\_sold >>item2.price)

{

if(item1.bookNo == item2.bookNo)

{

cnt ++;

item2.revenue = item2.units\_sold\*item2.price ;

item1.units\_sold += item2.units\_sold;

item1.revenue += item2.revenue;

}

else

std::cerr << "Data must refer to same ISBN"<<std::endl;

}

avg = item1.revenue /item1.units\_sold;

std::cout<<"total record is "<<cnt<<"\r\n"<<item1.bookNo<<item1.units\_sold<<item1.revenue<<avg<<std::endl;

}

#endif

#ifdef work1\_23

//练习2.23与2.24

//此程序其实有bug，必须连续输入相同的条目才可以统计到相同的书本信息,而且输入和输出混在一起非常难看

Sales\_data item1,item2;

if (cin >> item1.bookNo >> item1.units\_sold >>item1.price)

{

int cnt = 1; //计数

item1.revenue = item1.units\_sold\*item1.price ;

while(cin >> item2.bookNo >> item2.units\_sold >>item2.price)

{

if(item1.bookNo == item2.bookNo)

{

cnt ++;

item2.revenue = item2.units\_sold\*item2.price ;

item1.units\_sold += item2.units\_sold;

item1.revenue += item2.revenue;

}

else

{

std::cout<<"total record is "<<cnt<<"\r\n"<<item1.bookNo<<item1.units\_sold<<item1.revenue<<std::endl;

item1 = item2;

cnt = 1;

}

}

std::cout<<"total record of "<< item1.bookNo<<" is "<<cnt<<"\r\n"<<item1.bookNo<<item1.units\_sold<<item1.revenue<<std::endl;

}

#endif

while(1);

}

程序段使用了条件编译，还是比较习惯叫结构体而不是类；

## 第二章小结

C++的基本变量类型和C差不多，图书馆程序调试起来有点难度，比较花时间。学习了类这种数据类型，但是对类的操作还非常不熟练，感觉类赋值非常繁琐。

# 第三章 字符串、向量和数组

/\* 3.2.2节练习

\*\*\*

## 3.2 节练习

\*\*

\*/

#include <iostream>

#include <string>

using namespace std;

#define work3\_5

int main()

{

#ifdef work3\_2

//练习3.2

string line;

string word;

getline(cin,line);

cout << line << endl;

while(cin>> word)

cout << word << endl;

#endif

#ifdef work3\_4

//练习3.4

string line[2];

getline(cin,line[0]);

getline(cin,line[1]);

//cout<<line[0]<<line[1];

if(line[0] == line[1])

cout <<"ture";

else

cout<< ((line[0]>line[1])?line[0]:line[1]); //比较是否想等并输出大的字符串

cout << endl;

if(line[0].size() == line[1].size())

cout <<"ture";

else cout<<((line[0].size() > line[1].size())?line[0]:line[1]); //比较长度是否想等，若不等输出较长字符串

cout << endl;

#endif

#ifdef work3\_5

//练习3.5 输入多个字符串，将其连起来输出总和

string sum;

string line;

while(cin>>line) //将输入的所有字符串连接起来

sum += line;

cout << sum<<endl; //使用结束符会导致后面的输入也一同结束

sum = "";

while(cin>>line) //将输入的所有字符串用空格隔开连起来

sum = sum + line + " ";

cout << sum<<endl;

#endif

while(1);

return 0;

}

/\*

\*\* 2019年4月3日20:19:57

## 3.2.3 节练习

知识点：范围for语句

\*\*

\*\*

\*\*\*/

#include <iostream>

#include <string>

using namespace std;

int main(void)

{

//练习3.6

string somestr("whosyourdaddy");

string otherstr;

cout << somestr<<endl;

for(auto &c : somestr)c = 'x';

cout << somestr;

cout << somestr[0] << endl;

//

//练习3.10

//输入一串包含标点符号的字符串，将标点去除后输出

somestr = "how are you ? how old are you ? how old are you two ? go ,leave me alone.";

cout << somestr << endl;

for (auto &c : somestr)

{

if (ispunct(c))continue;

else otherstr += c;

}

cout << otherstr << endl;

while(1);

}

/\*\*\*\*\*\*\*

2019年4月10日21:49:56

## 3.3.2 节练习

P91

练习3.14 ：编写一段程序，用cin读入一组整数并将其存入一个vector对象；

练习3.15 ： 改写上题程序，使之读入的对象为字符串；

练习3.16 ：将输入vector的元素输出

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*8\*/

#include <vector>

#include <iostream>

#include <string>

using namespace std;

int main(void)

{

//练习3.14

vector<int> num;

int temp;

for (int i = 0; i < 10; i++)

{

cin >> temp;

num.push\_back(temp);

}

for (auto &i : num) //必须要用for语句逐一访问

cout << i << ' ';

//练习3.14

string word; //我记得在C语言中变量只能在执行语句前定义声明

vector<string> someStr;

while (cin >> word)

{

someStr.push\_back(word);

}

for (auto &i : someStr) //必须要用for语句逐一访问

cout << i << ' ';

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月10日22:33:40

P94

## 3.3.3 节练习

练习3.16 将练习3.13中vector对象的内容输出出来

练习3.17 从cin读入一组词并把他们存入一个vector对象，然后将所有词改写为大写格式输出

练习3.20 读入一组整数并把他们存入一个vector对象，将每对相邻整数的和输出出来。改写程序，先输出第一个和最后一个元素的和，接着输出第二个和倒数第二个元素和，以此类推

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

using namespace std;

#define work\_3\_16

int main()

{

#ifdef work\_3\_16

vector<int> v1;

vector<int> v2(10);

vector<int> v3(10,42);

vector<int> v4{};

vector<int> v5{ 10, 42 };

vector<int> v6{ 10 };

vector<string> v7{10,"hi"};

cout << "v1: ";

for (auto c : v1)cout << c << " ";

cout << endl;

cout << "v2: ";

for (auto c : v2)cout << c << " ";

cout << endl;

cout << "v3: ";

for (auto c : v3)cout << c << " ";

cout << endl;

cout << "v4: ";

for (auto c : v4)cout << c << " ";

cout << endl;

cout << "v5: ";

for (auto c : v5)cout << c << " ";

cout << endl;

cout << "v6: ";

for (auto c : v6)cout << c << " ";

cout << endl;

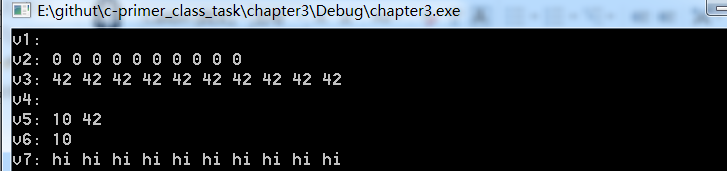
cout << "v7: ";

for (auto c : v7)cout << c << " ";

cout << endl;

#endif

输出



#ifdef word\_3\_17

//练习3.17

vector<string> inputWord;

string temp;

while (cin >> temp)

{

inputWord.push\_back(temp);

}

for (auto i : inputWord)

cout << i ;

cout << endl;

for (int i = 0; i < inputWord.size(); i++)

{

for (int j = 0; j < inputWord[i].length();j++)

{

inputWord[i][j] = toupper(inputWord[i][j]);

}

}

for (auto i : inputWord)

cout << i;

cout << endl;

#endif

#ifdef work\_3\_19

vector<int> vv1 {42,42,42,42,42,42,42,42,42,42};

vector<int> vv2(10, 42);

vector<int> vv3 = { 42, 42, 42, 42, 42, 42, 42, 42, 42, 42 };

//在定义重复的元素时，方法2是最便捷的，在需要定义列表元素时，显然方法1更直观更合适；

#endif

#ifdef work\_3\_20

vector<int> uu;

int temp;

while (cin >> temp)

{

uu.push\_back(temp);

}

for (int i = 0; i < uu.size(); i++)

{

if (i + 1 ==uu.size())

{

cout << uu[i] << " "; //引入防止下标溢出的边界判断

break;

}

else cout << uu[i] + uu[i + 1] << " ";

i++;

}

cout << "\n"<<"输出第i个和最后n-i个元素和" << endl;

for (int i = 0; i < uu.size()/2 ; i++)

{

cout << uu[i] + uu[uu.size() - i - 1] << " ";

}

if (uu.size() % 2 == 1) //对于奇数个数，中间的数输出它本身的两倍

cout << uu[uu.size() / 2]\*2 << " ";

#endif

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月16日20:43:16

P99

## 3.4.1 节练习

练习3.21 使用迭代器重做3.16

练习3.22 修改之前输出text的第一段程序，首先把text的第一段全改成大写，再输出它

练习3.23 编写一段程序，创建一个含有10个整数的vector对象，然后使用迭代器将所有元素的值变成原来的两倍。

输出vector对象的内容看是否正确

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

using namespace std;

#define work\_3\_23

int main()

{

#ifdef work\_3\_21

vector<int> v1;

vector<int> v2(10);

vector<int> v3(10, 42);

vector<int> v4{};

vector<int> v5{ 10, 42 };

vector<int> v6{ 10 };

vector<string> v7{ 10, "hi" };

cout << "v1: ";

for (auto it = v1.begin(); it != v1.end();it ++)cout << \*it << " ";

cout << endl;

cout << "v2: ";

for (auto it = v2.begin(); it != v2.end(); it++)cout << \*it << " ";

cout << endl;

cout << "v3: ";

for (auto it = v3.begin(); it != v3.end(); it++)cout << \*it << " ";

cout << endl;

cout << "v4: ";

for (auto it = v4.begin(); it != v4.end(); it++)cout << \*it << " ";

cout << endl;

cout << "v5: ";

for (auto it = v5.begin(); it != v5.end(); it++)cout << \*it << " ";

cout << endl;

cout << "v6: ";

for (auto it = v6.begin(); it != v6.end(); it++)cout << \*it << " ";

cout << endl;

cout << "v7: ";

for (auto it = v7.begin(); it != v7.end(); it++)cout << \*it << " ";

cout << endl;

#endif

#ifdef work\_3\_22

string s("hello world");

if (s.begin() != s.end())

{

auto it = s.begin();

\*it = toupper(\*it);

}

cout <<"改写首字母为大写： " <<s << endl;

for (auto it = s.begin(); it != s.end()&& !isspace(\*it);it++)

{

\*it = toupper(\*it);

}

cout <<"改写第一个单词为大写： "<< s << endl;

#endif

#ifdef work\_3\_23

vector<int> someNum(10,25);

for (auto c : someNum)

cout << c << " ";

cout << endl;

for (auto it = someNum.begin(); it != someNum.end(); it++)

{

\*it \*= 2;

}

for (auto c : someNum) //访问时必须用这种形式

cout << c << " ";

cout << endl;

for (auto it = someNum.begin(); it != someNum.end(); it++)

{

(\*it)++; //测试发现\*it++语句无效,必须加上括号

}

for (auto c : someNum) //访问时必须用这种形式

cout << c << " ";

cout << endl;

#endif

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月16日19:38:14

P101

## 3.4.2 节练习

练习3.24 使用迭代器重做3.20

练习3.25 使用迭代器实现3.3.3节的划分分数段的程序

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

using namespace std;

#define work\_3\_25

int main()

{

#ifdef work\_3\_24

vector<int> uu;

int temp;

while (cin >> temp)

{

uu.push\_back(temp);

}

cout << "输出相邻两个元素和，若一共奇数个，单独输出最后一个数" << endl;

for (auto it = uu.begin(); it != uu.end(); it++) //C++的迭代器其实跟C的指针很像

{

if (it + 1 == uu.end())

{

cout << \*it<< " "; //引入防止下标溢出的边界判断

break;

}

else cout << \*it + \*(it+1) << " ";

it++;

}

cout << "\n"<<"输出第i个和最后n-i个元素和" << endl;

auto itf = uu.begin();

auto ite = uu.end();

for (itf = uu.begin(); itf != uu.begin() + uu.size() / 2; itf++)

{

cout << \*itf + \*(ite-1) << " "; //uu.end()是一个不可访问的地址，访问它会导致溢出

ite--;

}

itf = uu.begin();

if (uu.size() % 2 == 1) //对于奇数个数，中间的数输出它本身的两倍

cout << \*(itf +(uu.size() / 2)) \* 2 << " ";

#endif

#ifdef work\_3\_25

vector<string> score\_part = { "0~9", "10~19", "20~29", "30~39", "40~49", "50~59", "60~69", "70~79", "80~89", "90~99","100" };

vector<unsigned> scores(11, 0);

unsigned grade;// , temp1;

// auto it = scores.begin();

//auto tempIt = scores.begin();

while (cin >> grade)

{

if (grade < 101)

{

auto it = scores.begin() + (grade / 10);

// tempIt = it + temp1; //

(\*it)++;

}

}

for (auto c : score\_part)

cout << c << " ";

cout << endl;

for (auto c : scores)

cout << c << " ";

#endif

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月16日22:10:57

P104

## 3.5.2 节练习

练习3.31 编写一段程序定义一个含有10个int的数组，令每个元素的值为其下标值

练习3.32 将上一题创建的数组拷贝给另外一个数组，利用vector重写程序实现类似的功能；

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

#include <stdio.h>

using namespace std;

int main()

{

//练习3.31

int a[10];

for (int i = 0; i < 10; i++)

a[i] = i;

cout << "a :";

for (int i = 0; i < 10; i++)

cout << a[i] << " ";

cout << endl;

//练习3.32

int b[10];

//strcpy(b,a);z只能拷贝字符串 char 类型

for (int i = 0; i < 10; i++)

b[i] = a[i];

cout << "b : ";

for (int i = 0; i < 10; i++)

cout << b[i] << " ";

cout << endl;

//使用vector重写

cout <<"vector reprogram "<< endl;

vector<int> v1 = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

vector<int> v2(10, 0);

cout << "v1: ";

for (auto it = v1.begin(); it != v1.end(); it++)

cout << \*it << " ";

auto it = v1.begin();

for (auto &c : v2)

{

c = \*it;

if (it != v1.end())

it++;

}

cout <<endl << "v2: ";

for (auto c : v2)

cout << c << " ";

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月16日22:33:56

P108

## 3.5.3 节练习

练习3.35 编写一段程序利用指针将数组中的元素置为0

练习3.36 编写一段程序比较两数组是否想等，再写一段程序比较连个vector对象是否想等

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

#include <stdio.h>

using namespace std;

int main()

{

//练习3.31

int a[10];

int \*p;

p = a;

for (int i = 0; i < 10; i++)

{

\*p = 0;

p++;

}

cout << "a :";

for (int i = 0; i < 10; i++)

cout << a[i] << " ";

cout << endl;

//练习3.36

//未指明比较数组比较的是什么，如果是比较数组名，是比较数组首地址；如果比较数组所有元素，则需要逐一遍历数组元素才能确认是否想等，会写，时间关系不写了

//

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月16日22:10:57

P104

练习3.5.5

练习3.41 编写一段程序，用整形数组初始化一个vector对象

练习3.42 将含有整数元素的vector对象拷贝给整形数组

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

#include <stdio.h>

using namespace std;

int main()

{

//练习3.41

int a[10];

for (int i = 0; i < 10; i++)

a[i] = i;

cout << "a :";

for (int i = 0; i < 10; i++)

cout << a[i] << " ";

cout << endl;

vector<int> v1(begin(a),end(a));

cout << "传过来的v1 ：";

for (auto c : v1)

cout << c << " ";

cout << endl;

//练习3.42

int b[20];

cout << "array b :";

for (int i = 0; i < v1.size(); i++)

{

b[i] = v1[i];

}

for (int i = 0; i < v1.size(); i++)

cout << b[i] << " ";

cout << endl;

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月16日22:10:57

P104

## 3.5.5 节练习

练习3.43 编写3个版本程序，均输出ia的元素，版本1使用范围for语句管理迭代过程，版本2、3使用普通for语句，版本2使用下标运算符，版本3使用指针，

不允许使用类型别名，auto关键字或者decltype关键字

练习3.44 将3.43的循环变量用类型别名

练习3.45 使用auto关键字实现

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

#include <stdio.h>

using namespace std;

int main(void)

{

int ia[3][4] = { 1, 2, 3, 4, 5 ,6,7,8,9,10,11,12};

//练习3.43

//版本1

cout << "vertion 1 :" << endl;

for (int(&row)[4] : ia)

{

for (int col : row)

cout << col << " ";

cout << endl;

}

cout << endl;

////版本2

cout << "vertion 2 :" << endl;

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 4; j++)

cout << ia[i][j] << " ";

cout << endl;

}

cout << endl;

////版本3

int (\*p)[4] = ia; //指向含有4个整形的数组的指针

int \*q;

cout << "vertion 3 :" << endl;

for (; p < ia + 3; p++)

{

for (q = \*p; q < 4 + \*p; q++)

cout << \*q << " ";

cout << endl;

}

cout << endl;

//练习3.44

cout << "接下来使用类型声明控制循环变量" << endl;

//版本1

typedef int int\_array[4];

//这个实在不知道怎么写

for (int\_array \*p1 = ia; p1 != ia + 3;++p1)

{

for (int \*q1 = \*p1; q1 < 4 + \*p1; q1++)

cout << \*q1 << " ";

cout << endl;

}

cout << endl;

//练习3.45 使用范围for语句

cout << "使用范围for语句控制循环变量" << endl;

for (auto &row : ia)

{

for (auto col : row)

cout << col << " ";

cout << endl;

}

while (1);

}

## 第三章小结

学习了string类型，vector类型，迭代器的用法，数组我在C语言中已经比较熟练掌握了。C++11引入了范围for语句和auto变量赋值，开始不太熟悉，用起来之后感觉真的好用。

# 第四章 表达式

/\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月17日21:02:12

## 4.2节练习

练习4.5 写出表达式求值结果

练习4.6 写一个确定一个整数是奇数还是偶数的表达式

\*\*\*\*/

#include<iostream>

using namespace std;

int main()

{

//练习4.5

int a, b, c, d;

a = -30 \* 3 + 21 / 5;

b = -30 + 3 \* 21 / 5;

c = 30 / 3 \* 21 % 5;

d = -30 / 3 \* 21 % 4;

cout << "a = " << a << endl << "b = " << b << endl << "c = " << c << endl << "d = " << d << endl;

//练习4.6

int number;

cin >> number;

if (number%2 == 0)

{

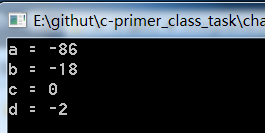
cout << number << " 是偶数" << endl;

}else if(number%2 == 1)cout << number << " 是奇数" << endl;

while (1);

}

表达式的值：



/\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月17日21:02:12

## 4.3节练习

练习4.10 为while循环写一个条件，从标准输入中读入整数，遇到42的时候停止

练习4.11 写一个表达式判断a/b/c/d值的大小关系，确保a大于b，b大于c，c大于d

\*\*\*\*/

#include<iostream>

using namespace std;

int main()

{

//练习4.10

int temp = 0;

while (temp != 42) //遇到42就停止

{

cin >> temp;

cout << "input is " << temp << endl;

}

cout << "break " << endl;

//练习4.11

int a = 50, b = 4, c = 3, d = 5;

if (a > b && b > c && c > d)cout << "ture" << endl;

else cout << "false" << endl;

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月17日21:33:10

## 4.7节练习

条件运算符

练习4.21 使用条件运算符从vector<int>中找到奇数值元素，将其翻倍

练习4.22.1 使用条件运算符将示例程序划分成high pass 、pass 、fail 三种，扩展程序进一步将60分到75分成绩设定为low pass

练习4.22.2 使用if语句完成4.22的程序

\*\*\*\*/

#include<iostream>

#include<vector>

#include<string>

using namespace std;

int main()

{

//练习4.21

vector<int> vv = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

for (auto c : vv)

cout << c << " ";

cout << endl << "将奇数翻倍后输出: " << endl;

for (auto &c : vv)

(c % 2 == 1) ? (cout << 2\*c << " ") : (cout << c << " ");

//练习4.22.1

#if 0

int grade;

string finalgrade = "what ??";

while (cin >> grade)

{

finalgrade = (grade > 100) ? "error " : (grade > 90) ? "high pass " : (grade >= 75) ? "pass" : (grade >= 60) ? "low pass" : "fail";

cout << finalgrade << endl;

}

#endif

//练习4.22.2

int grade;

while (cin >> grade)

{

if (grade > 100)

cout << "error";

else if (grade > 90)

cout << "high pass";

else if (grade >= 75)

cout << "pass";

else if (grade >= 60)

cout << "low pass";

else

cout << "fail";

cout << endl;

}

//分析：从可读性来说，使用if语句更容易理解，从程序简洁性来说，条件表达式更简洁，但是条件表达式因为结合顺序问题在处理过长的判断逻辑的时候极易写错程序

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月17日21:33:10

## 4.9节练习

P140

sizeof

练习4.28 输出每一种内置类型所占的空间大小

\*\*\*\*/

#include<iostream>

#include<vector>

#include<string>

using namespace std;

int main()

{

//练习4.28

cout << "类型名称\t" << "所占空间/byte" << endl;

cout << "bool\t\t" << sizeof(bool) << endl;

cout << "char\t\t" << sizeof(char) << endl;

cout << "wchar\_t\t\t" << sizeof(wchar\_t) << endl;

cout << "char16\_t\t" << sizeof(char16\_t) << endl;

cout << "char32\_t\t" << sizeof(char32\_t) << endl;

cout << "short\t\t" << sizeof(short) << endl;

cout << "int\t\t" << sizeof(int) << endl;

cout << "long\t\t" << sizeof(long) << endl;

cout << "long long\t" << sizeof(long long) << endl;

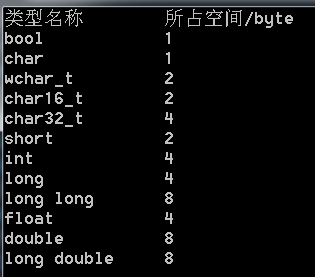
cout << "float\t\t" << sizeof(float) << endl;

cout << "double\t\t" << sizeof(double) << endl;

cout << "long double\t" << sizeof(long double) << endl;

while (1);

}



这个结果很有意义，很多地方会涉及到sizeof获取长度，需要熟记；

第四章小结

C++的基本运算符跟C是一致的，这些我都已经比较熟悉了。值得说的是sizeof语句，面试的时候经常会考这个，对于不同的编译系统，sizeof的结果会有点区别，同时sizeof作用于指针时结果也会不一样。另外运算符中的条件运算符我一直记不熟，写了两段程序，感觉印象比较深刻了。

# 第五章 语句

/\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月17日22:19:11

## 5.3.2节练习

练习5.9 使用一系列语句统计从cin读入的文本中有多少元音字母

练习5.10 遇到大写字母时也计数

练习5.12 统计加上ff fl fi

\*\*\*\*/

#include<iostream>

#include<vector>

#include<string>

using namespace std;

int main()

{

//练习5.9 & 练习5.10

#if 0

char ch;

int cnto = 0;

while (cin >> ch)

{

if (ch == 'a'|| ch == 'A')cnto++;

else if (ch == 'e' || ch == 'E')cnto++;

else if (ch == 'i' || ch == 'I')cnto++;

else if (ch == 'o' || ch == 'O')cnto++;

else if (ch == 'u' || ch == 'U')cnto++;

}

cout << "元音字母数量: " << cnto<< endl;

#endif

string word;

int cntfi = 0, cntfl = 0, cntff = 0;

while (cin >> word)

{

//switch (word) string不能用于switch

if (word == "ff")cntff++;

else if (word == "fl")cntfl++;

else if (word == "fi")cntfi++;

}

cout << "ff :" << cntff << endl << "fl :" << cntfl << endl << "fi :" << cntfi << endl;

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月17日22:19:11

## 5.5.1节练习

练习5.20 从标准输入读取对象序列直到出现两个重复单词或者读完为止，若出现连个重复单词，中断循环，输出重复单词或者无重复单词

\*\*\*\*/

#include<iostream>

#include<vector>

#include<string>

using namespace std;

int main()

{

//练习5.20

string word, wordpre;

char flag = 0;

while (cin >> word)

{

if(word == wordpre)

{

flag = 1;

break;

}

wordpre = word;

}

if (flag)cout << word << endl;

else cout << "没有重复单词" << endl;

while (1);

}

## 第五章小结

基本语句if else while dowhile swtich break continue goto for都已经很熟悉，所以这章的作业写的很少。异常处理的try和throw感觉使用的机会不大，暂时不管吧，用到了再翻回来看看。

# 第六章 函数

/\*\*\*\*\*\*\*\*\*

2019年4月17日23:01:01

## 6.1节练习

练习6.3 编写fact函数

练习6.4 编写一个与用户交互的函数，用户输入一个数字，生成该数字的阶乘；

练习6.5 编写一个函数输出其实参的绝对值

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

using namespace std;

//练习6.3

int fact(int val)

{

int ret = 1;

while (val > 1)

ret \*= val--;

return ret;

}

//练习6.4

int inputNum(void)

{

int val;

cin >> val;

int ret = 1;

while (val > 1)

ret \*= val--;

return ret;

}

//练习6.5

int outputABS(int val)

{

if (val >= 0)

return val;

else

return (- 1 \* val);

}

int main()

{

int num1;

num1 = fact(4);

cout << num1 << endl;

num1 = inputNum();

cout << num1<<endl;

cout << outputABS(50) << " " << outputABS(-20) << " " << endl;

while (1);

}

/\*\*\*\*\*\*\*\*\*

2019年4月17日23:12:54

6.1.2节练习

编写一个头文件，包含6.1节练习的函数声明

\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef \_\_CHAPTER6\_H\_\_

#define \_\_CHAPTER6\_H\_\_

int fact(int val);

int inputNum(void);

int outputABS(int val);

#endif

/\*\*\*\*\*\*\*\*\*

2019年4月17日23:01:01

## 6.2.1节练习

6.2.2节练习

练习6.10 写一个使用指针交换两个整数只的函数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

using namespace std;

//指针形式交换数据

int swapBP(int \*p,int \*q)

{

int temp;

temp = \*p;

\*p = \*q;

\*q = temp;

return 0;

}

//引用形式交换数据

int swapRe(int &a, int &b)

{

int temp;

temp = a;

a = b;

b = temp;

return 0;

}

int main()

{

//练习6.10

int num1=55,num2=22;

int \*p =&num1, \*s=&num2;

cout << "交换前: " << "num1 = " << num1 << endl << "num2 = " << num2 << endl;

swapBP(p,s);

cout << "交换后: " << "num1 = " << num1 << endl << "num2 = " << num2 << endl;

//练习6.12

swapRe(num1, num2);

cout << "使用引用形式再次交换: " << "num1 = " << num1 << endl << "num2 = " << num2 << endl;

//引用形式不需要额外定义指针进行操作，更简便，缺点是C语言不支持引用形式

while (1);

}

/\*\*\*\*\*\*\*\*\*

2019年4月17日23:01:01

## 6.2.3节练习

练习6.17 写一个函数判断string中是否有大写字母，另一个函数将所有对象都改写成小写形式

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <cctype>

using namespace std;

//是否含有大写字母

int hadUpper0(string ss)

{

for (auto c : ss)

{

if (isupper(c))

{

cout << "有大写字母 " << endl;

return 1;

}

}

cout << "没有大写字母 " << endl;

return 0;

}

//将所有大写字母转换成小写字母

int allToLower(string &ss)

{

for (auto &c : ss)

{

if (isupper(c))

{

c = tolower(c);

}

}

return 0;

}

int main()

{

//练习6.17

string ss1 = "hello world";

string ss2 = "SURPRISE mother fucker";

cout << "ss1 :";

hadUpper0(ss1);

cout << "ss2 :";

hadUpper0(ss2);

allToLower(ss2);

cout << "转换后的ss2 ：" << ss2 << endl;

//一个是引用型，一个是非引用型参数

while (1);

}

/\*\*\*\*\*\*\*\*\*

2019年4月18日19:35:49

6.5.2节练习

练习6.44 将6.2.2节的isShorter函数改写成内联函数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

using namespace std;

inline const string isShorter(const string &s1, const string &s2) //内联函数可以消除函数调用的开销

{

return s1.size() <= s2.size() ? s1:s2;

}

int main()

{

//练习6.44

string s1 = "hello", s2 = "world!!";

cout << isShorter(s1, s2)<< endl;

//一个是引用型，一个是非引用型参数

while (1);

}

/\*\*\*\*\*\*\*\*\*

2019年4月18日19:35:49

## 6.7节练习

练习6.54 编写函数声明，令其接受两个int形参并且返回值也是int；然后声明一个vector对象，令其元素指向该函数的指针；

练习6.55 编写4个函数，分别对两个int值执行加减乘除运算，在上一题创建的vector对象中保存指向这些函数的指针；

练习6.56 调用上述vector对象中的每个元素并输出其结果；

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <string>

#include <vector>

using namespace std;

//练习6.54

int fuction\_add(int a, int b)

{

return a + b;

}

int fuction\_sub(int a, int b)

{

return a - b;

}

int fuction\_mul(int a, int b)

{

return a\*b;

}

int fuction\_div(int a, int b)

{

if (b != 0)

return a / b;

else return 0;

}

int main()

{

//练习6.54 6.55 6.56

vector<decltype(fuction\_add)\*>ff; ///声明写的有点难理解，还需要更多的练习熟悉

int(\*pf)(int a, int b) = fuction\_add;

pf = fuction\_add;

ff.push\_back(pf);

pf = fuction\_sub;

ff.push\_back(pf);

pf = fuction\_mul;

ff.push\_back(pf);

pf = fuction\_div;

ff.push\_back(pf);

for (auto c : ff)

{

cout << c(50, 10) << endl;

}

while (1);

}

第六章小结

这章主要都在讲函数的事情，函数定义、声明、调用，参数设置等等这些已经比较熟悉了，但是函数指针、数组作为函数参数等这些内容因为用的其实不多所以不太熟悉，另外函数指针这块可以说非常生疏，光写一个联系恐怕还不能完全掌握。

# 第八章 IO类

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## 8.1.2 节练习

练习8.1

练习8.2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

using namespace std;

istream& f\_io(istream& input)

{

int val;

while (input >> val, !input.eof())

{

if (input.bad())

throw runtime\_error ("IO流出错了！");

if (input.fail())

{

cerr << "输入格式错误，请重试 " << endl; //cerr专门用于输出错误信息

input.clear();

input.ignore(5, '\n'); //如果不写这一句，那么函数会一直跑个不停

continue;

}

cout << val << endl;

}

input.clear();

return input;

}

int main(void)

{

f\_io(cin);

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## 8.2.1 节练习

练习8.4 编写函数，以读模式打开一个文件，将其内容输入到一个string的vector中，将每一行作为一个独立的元素进行存储

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

using namespace std;

int main(void)

{

//练习8.4

vector<string> vec1;

string line;

ifstream input("E:/text.txt"); //文件路径需要用反斜杠

//练习8.4

if (!input)

{

cout << "can't not open file,please retry!" << endl;

}

else

{

while (getline(input, line))

{

vec1.push\_back(line);

}

cout << "以一行为元素" << endl;

for (auto c : vec1)

{

cout << c << endl;

}

}

input.close();

//练习8.5

input.open("E:/text.txt");

vector<string> vec2;

if (!input)

{

cout << "can't not open file,please retry!" << endl;

}

else

{

while (input >> line)

{

vec2.push\_back(line);

}

cout << "以单词为元素" << endl;

for (auto c : vec2)

{

cout << c << endl;

}

}

input.close();

while (1);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月18日21:46:08

## 8.3.1 节练习

练习8.9 改写f\_io函数，使其能输出一个istringstream对象的内容

练习8.10 写程序读取一个文件的行保存在vector对象中，然后使用一个istringsteam从vector读取数据元素，每次读取一个单词

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

using namespace std;

istringstream& f\_io(istringstream& input)

{

string val;

while (input >> val, !input.eof())

{

if (input.bad())

throw runtime\_error("IO流出错了！");

if (input.fail())

{

cerr << "输入格式错误，请重试 " << endl; //cerr专门用于输出错误信息

input.clear();

input.ignore(5, '\n'); //如果不写这一句，那么函数会一直跑个不停

continue;

}

cout << val << endl;

}

input.clear();

return input;

}

int main(void)

{

#if 0

//练习8.9

ostringstream some\_str;

some\_str << "whosyourdaddy" << endl;

istringstream input(some\_str.str());

//cout << some\_str;

f\_io(input);

#endif

//练习8.10

ifstream input("E:/text.txt");

string line;

vector<string> str\_group;

while (getline(input, line))

{

str\_group.push\_back(line);

}

for (auto c : str\_group)

{

cout << c << endl;

}

input.close();

cout << endl << "使用 istringstream 一个一个单词的读取数据 " << endl ;

auto it = str\_group.cbegin();

while (it != str\_group.cend())

{

istringstream str\_line(\*it); //绑到迭代器上面

string word;

while (str\_line >> word)

cout << word << endl;

it++;

}

while (1);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月18日22:23:55

## 8.3.2 节练习

练习8.13 重写电话本程序，从一个命名文件而非cin读取数据

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

using namespace std;

struct PersionInfo{

string name;

vector<string> phones;

};

int main(void)

{

//练习8.13

string line, word;

vector<PersionInfo> people;

ifstream input("E:/phone.txt");

if (!input)cout << "打开文件失败" << endl;

else

{

while (getline(input, line))

{

PersionInfo info;

istringstream record(line);

record >> info.name;

while (record >> word)

{

info.phones.push\_back(word);

}

people.push\_back(info);

}

}

for (auto c : people)

{

cout << c.name << " ";

for (auto out : c.phones)

cout << out << " ";

cout << endl;

}

while (1);

return 0;

}

## 第八章小结

这章的内容感觉非常生疏，第一次见到，写程序都是参考着标准作业写的，不过很有收获的是学会了读取txt文件的操作，之前没有涉及过这样的操作。

# 第九章 顺序容器

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月19日19:13:20

## 9.2节练习

练习9.2 定义一个list对象，元素是int的deque

练习9.4 编写函数，接收一对指向vector<int>的迭代器和int值，在两个迭代器指定的范围内查找给定的值，返回一个布尔值来指出是否找到

练习9.5 重写函数，返回一个迭代器指向找到的元素

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<vector>

#include<list>

#include<deque>

using namespace std;

//练习9.4

bool find\_val(int val, vector<int>::iterator beg, vector<int>::iterator end)

{

for (vector<int>::iterator it = beg; it < end; it++)

{

if (\*it == val)

return 1;

}

return 0;

}

//练习9.5

vector<int>::iterator find\_val\_it(int val, vector<int>::iterator beg, vector<int>::iterator end)

{

for (vector<int>::iterator it = beg; it < end; it++)

{

if (\*it == val)

return it;

}

return end;

}

int main(void)

{

list<int> list1;

list<deque<int>> object\_list; //必须声明命名空间namespace，否则不合法

//9.4

vector<int> vect = {11,22,33,44,55,66};

int value = 66;

if (find\_val(value, vect.begin(), vect.end()))

cout << "查找到" << value << "值" << endl;

else

cout << "查找失败，未找到该值 : " << value << endl;

//9.5

if (vect.end() != find\_val\_it(value, vect.begin(), vect.end()))

{

cout << "查找到" << \*(find\_val\_it(value, vect.begin(), vect.end())) << "值" << endl;

}

else cout << "查找失败，未找到该值 : " << value << endl;

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月19日19:13:20

## 9.2.5节练习

练习9.14 编写程序，将一个list中的char\*指针（指向C风格字符串）元素赋值给一个vector中的string

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<vector>

#include<list>

#include<deque>

#include<string>

using namespace std;

int main(void)

{

//char str[] = { "surprise mother fxxker!" };

list<char\*> list1 = { "surprise mother fxxker!" ,"haha","who say that?","whosyourdaddy"};

vector<string>vect = {"go go go","speak"};

//vector<int>vect(10,9);

vect.assign(list1.begin(), list1.end()); //少了string的头文件，然后到处找原因耽误了半个小时

cout << vect.capacity() << " " << vect.size() << endl;

for (auto it = vect.begin(); it!=vect.end();it++)

{

cout << \*it << endl;

}

while (1);

}

写这个程序的时候因为没有把string头文件包进来，编译一直通不过懵逼了半小时，以后应该少犯这种浪费时间的低级错误。

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月19日19:13:20

## 9.2.7节练习

练习9.15 写程序判断两个vector<int>是否想等

练习9.16 重写程序，比较一个list<int>中的元素和一个vector<int>中的元素

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<vector>

#include<list>

#include<deque>

#include<string>

using namespace std;

int main(void)

{

vector<int> ivec1(10, 8);

vector<int> ivec2(10, 8);

list<int> ilist1(10, 2);

if (ivec1 == ivec2) cout << "相等的哦，惊不惊喜意不意外？" << endl;

//for (int i = 0; i < ivec1.size() && i<ilist1.size(); i++)

//{

// (ivec1[i] == ilist1[i]) ? (cout << "ture") : (cout << "false"); 不行，无法用下标访问list

//}

//if (ilist1 == ivec1) 容器类型不相同，无法直接比较

auto l\_it = ilist1.begin();

//auto l\_end = ilist1.end();

auto i\_it = ivec1.begin();

for (; l\_it != ilist1.end() && i\_it != ivec1.end(); l\_it++, i\_it++)

{

(\*i\_it == \*l\_it) ? (cout << "ture" <<endl): (cout << "false" << endl);

}

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月19日21:10:08

## 9.3.2节练习

练习9.24 分别使用at 下标运算符 front和begin提前vector中的第一个元素

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<vector>

#include<list>

#include<deque>

#include<string>

using namespace std;

int main(void)

{

vector<string> empty;

string str;

cout << empty.at(0);

cout << endl;

cout << empty[0];

cout << endl;

cout << empty.front();

cout << endl;

cout << \*(empty.begin());

cout << endl;

//因为vector是空的，引起了花式报错

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月19日21:10:08

9.3.3节练习

练习9.26 使用下面代码定义ia，将ia拷贝到一个vector和一个list中，使用单迭代版本的erase从list中删除奇数元素，从vector中删除偶数元素

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<vector>

#include<list>

#include<deque>

#include<string>

using namespace std;

int main(void)

{

int ia[] = {0,1,1,2,3,5,8,13,21,55,89};

vector<int> ivec1;

list<int> ilist;

cout << "ia [] :";

for (int i = 0; i < sizeof(ia)/4; i++)

{

ivec1.push\_back(ia[i]);

ilist.insert(ilist.begin(), ia[i]);

cout << ia[i] << " ";

}

cout << endl;

for (auto it = ivec1.begin(); it < ivec1.end(); it++)

{

if (\*it % 2 == 0)

{

it = ivec1.erase(it);

}

}

cout << "ivec1 :";

for (auto c : ivec1)cout << c << " ";

cout << endl;

auto iterList = ilist.begin();

while (iterList != ilist.end())

{

if (\*iterList % 2)

{

iterList = ilist.erase(iterList);

}

else

{

iterList++;

}

}

//for (auto it = ilist.begin(); it != ilist.end(); it++) //会因为迭代器++指令导致出错

//{

// if (\*it % 2)

// {

// it = ilist.erase(it);

// }

//}

cout << "ilist :";

for (auto c : ilist)cout << c << " ";

cout << endl;

while (1);

}

9.4节是vector数据对内存处理的理解，练习不写了

9.5节涉及string常用的操作处理，更多的是熟悉操作，暂时跳过；

第九章小结

本章对容器做了更深入的探究，vector，list，deque是几种常用的容器，其中list不支持随机访问，vector和deque用法上其实差不多。还是需要具体到项目的程序中理解其用法才能有更深的体会。

# 第十章

/\*\*\*\*\*\*\*

2019年4月19日22:19:34

## 10.1节练习

练习10.1 头文件algorithm中有一个cout函数，使用它输出有多少个值等于给定值

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <algorithm>

#include <vector>

#include <iostream>

using namespace std;

int main(void)

{

vector<int> vec = { 0, 11, 22, 55, 11, 22, 44, 55, 33 };

int res,val;

val = 55;

res = count(vec.begin(),vec.end(),val);

cout << val << "的个数是：" << res << endl;

while (1);

}

/\*\*\*\*\*\*\*

2019年4月19日22:19:34

## 10.4.1节练习

练习10.27 使用unique将一个vector中不重复的元素拷贝到一个初始为空的list中

练习10.28 一个vector中保存1到9，使用三种容器拷贝元素并输出

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <algorithm>

#include <vector>

#include <iostream>

#include <sstream>

#include <numeric>

#include <iterator>

#include <list>

using namespace std;

int main(void)

{

//10.27

vector<int> number = { 1, 1, 2, 1, 2, 3, 3, 4, 5, 6, 6, 7, 7, 8, 8, 8, 9, 9, 9, 4, 3, 5 };

vector<int> vec1, vec2, vec3, transit\_vec;

cout << "the initial bumber list is :";

for (auto c : number) cout << c << " "; cout << endl;

//先把所有重复的元素排序一下，聚到一块

stable\_sort(number.begin(), number.end());

unique\_copy(number.begin(), number.end(), inserter(transit\_vec, transit\_vec.begin()));

for (auto c : transit\_vec) cout << c << " "; cout << endl;

for (auto c : transit\_vec)

{

\*inserter(vec1, vec1.begin()) = c;

}

cout << "inserter copy :";

for (auto c : vec1) cout << c << " "; cout << endl;

for (auto c : transit\_vec)

{

\*back\_inserter(vec2) = c;

}

cout << "back\_inserter copy :";

for (auto c : vec2) cout << c << " "; cout << endl;

//for (auto c : transit\_vec) 成员列表里没有push\_front ，无法使用这个迭代器

//{

// \*front\_inserter(vec3) = c;

//}

//cout << "front\_inserter copy :";

//for (auto c : vec3) cout << c << " "; cout << endl;

while (1);

}

# 第十章小结

这一章有很多不错的算法函数，像无重复拷贝，排序这些基本算法，处理数据的时间直接调用，简直好用哭了，不用再自己造轮子超开心好吗。不得不说，C++博大精深，啥玩意都有。时间关系，作业也没有一个个细细的写完，先这样了。

第十一章 关联容器

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月20日11:36:37

## 11.2.1节练习

练习11.7 定义一个map，关键字是家庭的姓，值是一个vector，保存家中的孩子的名，编写代码，时间添加新的家庭以及向已有家庭中添加新的孩子

11.2.2. 节练习

练习11.9 定义一个map，将单词与一个行号的list关联，list中保存的是单词所出现的行号

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

#include <string>

#include <vector>

#include <map>

#include <list>

using namespace std;

void add\_family(map<string, vector<string>> &ff,const string first\_name)

{

ff[first\_name] = vector<string> (); //添加一个没有娃的家庭

}

void add\_child(map<string, vector<string>> &ff, const string first\_name,const string name)

{

ff[first\_name].push\_back(name);

}

int main()

{

map<string, vector<string>> families;

add\_family(families, "王");

add\_family(families, "波");

add\_family(families, "雷");

add\_child(families, "王", "思聪");

add\_child(families, "王", "大头");

add\_child(families, "波", "儿霸");

add\_child(families, "波", "多野结衣");

add\_child(families, "雷", "不思议");

for (auto fff : families)

{

for (auto child : fff.second)

{

cout << fff.first << child << " ";

}

cout << endl;

}

//练习11.9

ifstream input("E:/text.txt"); //文件路径需要用反斜杠

if (!input)

{

cout << "can't not open file,please retry!" << endl;

}

map<string, list<int>> word\_line\_no;

string line, word;

int line\_no = 0;

while (getline(input, line))

{

line\_no++;

istringstream in\_line(line);

while (in\_line >> word)

word\_line\_no[word].push\_back(line\_no);

}

for (auto ww : word\_line\_no)

{

for (auto line\_number : ww.second)

cout << ww.first << "所在的行是：" << line\_number << endl;

}

while (1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月20日11:36:37

## 11.2.3节练习

练习11.12 读取string 和int 序列，将每个string和int存入一个pair中，pair保存在一个vector中

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

#include <string>

#include <vector>

#include <map>

#include <list>

#include <utility>

using namespace std;

int main()

{

vector<pair<string, int>> ppp;

string word;

int num;

for (int i = 0; i < 4; i++)

{

cin >> word >> num;

//练习11.3 三种方法创建

//ppp.push\_back({ word, num });

//ppp.push\_back(pair<string, int>(word, num));

ppp.push\_back(make\_pair(word, num));

}

for (auto out : ppp)cout << out.first << ":" << out.second << endl;

while (1);

}

11.3.1节，不会做

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2019年4月20日13:59:52

## 11.3.6节练习

练习11.33 单词转换程序

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <fstream>

#include <sstream>

#include <string>

#include <vector>

#include <map>

#include <list>

#include <utility>

using namespace std;

//打开转换规则文件，产生映射map

map<string, string> buildMap(ifstream &in) {

map<string, string> trans\_map;

string key;

string value;

while (in >> key && getline(in, value)){

if (value.size() > 1) {

trans\_map[key] = value.substr(1);//除去空格号

}

else {

throw runtime\_error("No rule for " + key);

}

}

return trans\_map;

}

//单词转换函数

const string& transform(const string &s, const map<string, string> &mp) {

auto it = mp.find(s);

if (it != mp.cend()) {

return it->second;//返回转换后的字符串

}

else {

return s;

}

}

//将转换后的数据输出到标准输出流

void word\_transform\_to\_cout(ifstream &map\_file, ifstream &input)

{

auto trans\_map = buildMap(map\_file);

string text;

while (getline(input, text))

{

istringstream stream(text);

string word;

bool firstword = 1;

while (stream >> word)

{

if (firstword)

firstword = 0;

else cout << " ";

cout << transform(word, trans\_map);

}

cout << endl;

}

}

//将转换后的数据输出到txt文件中

void word\_transform\_to\_file(ifstream &map\_file, ifstream &input,ostream &out)

{

auto trans\_map = buildMap(map\_file);

string text;

while (getline(input, text))

{

istringstream stream(text);

string word;

bool firstword = 1;

while (stream >> word)

{

if (firstword)

firstword = 0;

else out << " ";

out << transform(word, trans\_map);

}

out << endl;

}

}

int main()

{

ifstream infileMap("E:/wordmap.txt");

if (!infileMap){ cout << "open Map fail !" << endl; }

ifstream infileText("E:/inputext.txt");

if (!infileText)cout << "open input file fail!" << endl;

ofstream outfileText("E:/outputext.txt");

if (!outfileText)cout << "open output txt fail!" << endl;

word\_transform\_to\_cout(infileMap, infileText);

word\_transform\_to\_file(infileMap, infileText, outfileText); //不知道写入文件为什么失败了，text没有内容输出

while (1);

}

## 第十一章小结

11章学了两个新容器map和pair，通过“抄写”课本例程，编译查看输出，对它有了一个初步的认识，时间关系作业只是选做了一些，正式上课了再细细的把能做的作业都过一遍。另外最后一个练习，不知道为什么ostream流输出到文件，txt文件里始终没有任何内容