## 7.2

#include <iostream>

//using namespace std;

//

//const int glof=10;

//int ii=0;

//int \*grade=new int[glof];

//

//void enter();

//void display();

//void avr();

//

//int main()

//{

// enter();

// display();

// avr();

// return 0;

//}

//

//void enter()

//{

// double num;

// for(int i=0; i<glof ; i++)

// {

// if(cin) //之前并没有输入数据,cin将无法判断

// {

// cout << " Enter score # " << i << ": ";

// cin >> grade[i];

// ii++; //错误的ii++位置将多输出一个grade[i]

// }

// else

// {

// cout << "Have a Bad Input !!! ; "<<endl;break;

// }

// }

//}

// void enter() //正确的enter（）函数

//{

// int bb;

// for(int i=0; i<glof ; i++)

// {

// cout << " Enter score # " << i << ": ";

// cin >> bb;

// if (!cin)

// {

// cin.clear();

// while (cin.get() != '\n')

// continue;

// cout << "Bad input; input process terminated.\n";

// break;

// }

// grade[i]=bb;

// ii++;

// }

//}

//void display()

//{

// cout << "score :";

// for(int j=0; j<ii ;j++ )

// {

// cout << grade[j]<< " ";

// }

// cout <<endl;

//}

//

//void avr()

//{

// double aver=0 ;

// for(int m=0; m<ii ;m++ )

// {

// aver += grade[m];

// }

// cout << "The average of scores is:" << aver/ii <<endl;

//}

#include <iostream>

using namespace std;

double score[10];

int input(double [], int);

void average(double [], int);

void show(const double[], int);

int main()

{

int size = input(score, 10);

if (size > 0)

{

show(score, size);

average(score, size);

}

cout << "Done.\n";

return 0;

}

int input(double score[], int limit)

{

double a;

int i;

for (i = 0; i < limit; i++)

{

cout << "Your score: ";

cin >> a;

if (!cin)

{

cin.clear();

while (cin.get() != '\n')

continue;

cout << "Bad input; input process terminated.\n";

break;

}

else if (a<0)

break;

score[i] = a;

}

return i;

}

void show(const double ar[], int n)

{

double total = 0.0;

cout << "Score: ";

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

{

cout << ar[i] << " ";

}

cout << endl;

}

void average(double ar[], int n)

{

double av,total = 0.0;

int i;

for (i = 0; i < n; i++)

{

total += ar[i];

}

av =total /i;

cout << "The average score: " << av << endl;

}

## 7.3

#include <iostream>

struct box

{

char maker[40];

float height;

float width;

float length;

float volume;

};

void set\_box(box \*);

void show\_box(box);

int main()

{

using namespace std;

box carton = {"Bingo Boxer", 2, 3, 5};

set\_box(&carton);

show\_box(carton);

return 0;

}

void set\_box(box \*pb)

{

pb->volume = pb->height \* pb->length \* pb->width;

}

void show\_box(box b)

{

using namespace std;

cout << "Box maker: " << b.maker

<< "\nheight: " << b.height

<< "\nlwidth: " << b.width

<< "\nlength: " << b.length

<< "\nvolume: " << b.volume << endl;

}

## 7.4

#include <iostream>

long double probability(unsigned numbers, unsigned picks);

int main()

{

using namespace std;

double total, choices, mtotal; //第一次在total个球内选choices个号码

long double probability1, probability2;

cout << "Enter total number of game card choices and\n"<<"number of picks allowed for the field:\n";

while ((cin >> total >> choices) && choices < total)

{

cout << "Enter total number of game card choices and\n"<<"number of picks allowed for the mega:\n";

if (!(cin >> mtotal)) // 第二次在 mtotal个球内选1个号码

break;

probability1 = probability(total, choices);

probability2 = probability(mtotal, 1);

cout << "The chances of getting all " << choices << " picks is one in "<< probability1 << ".\n";

cout << "The chances of getting the megaspot is one in "<< probability2 << ".\n";

cout << "You have one chance in ";

cout << probability1 \* probability2;

cout << " of winning.\n";

cout << "Next set of numbers (q to quit): ";

}

cout << "bye\n";

return 0;

}

long double probability(unsigned numbers, unsigned picks)

{

long double result =1.0;

long double n;

unsigned p;

for (n = numbers, p = picks; p > 0; n--, p--)

result = result \* n / p;

return result;

}

## 7.5

#include <iostream>

using namespace std;

unsigned long sub(int);

int main()

{

cout << "Enter one integer: (q to quit)";

int num;

while(cin >> num) // num是int类型，当输入非int类型将停止循环

{

unsigned long result = sub(num);

cout << "The result of " << num << "! is: "<< result << endl<< "Next number: ";

}

return 0;

}

unsigned long sub(int n)

{

unsigned long result = n;

if (result > 0)

result = result \* sub(n - 1);

else

result = 1;

return result;

}

## 7.6

#include <iostream>

const int Size = 10;

int Fill\_array(double ar[], int n);

void Show\_array(const double ar[], int n);

void Reverse\_array(double ar[], int n);

int main()

{

using namespace std;

double values[Size];

int len = Fill\_array(values, Size);

cout << "Array values:\n";

Show\_array(values, len);

cout << "Array reversed:\n";

Reverse\_array(values, len);

Show\_array(values, len);

cout << "All but end values reversed:\n";

Reverse\_array(values+1, len-2);

Show\_array(values, len);

return 0;

}

int Fill\_array(double ar[], int n)

{

using namespace std;

double temp;

int i;

for (i=0; i<n; i++)

{

cout << "Enter value #" << i+1<< ": ";

cin >> temp;

if (!cin)

break;

ar[i] = temp;

}

cout << endl;

return i;

}

void Show\_array(const double ar[], int n)

{

using namespace std;

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

cout << "Property #" << i+1 << ": "<< ar[i] << endl;

cout << endl;

}

void Reverse\_array(double ar[], int n)

{

double temp;

for (int i=0,j=n-1; i<j; i++,j--)

{

temp =ar[i];

ar[i] = ar[j];

ar[j] = temp;

}

}

## 7.7

#include <iostream>

const int Max = 5;

double \* fill\_array(double \* begin, double \* end);

void show\_array(const double \* begin, const double \* end);

void revalue(double r, double \* begin, double \* end);

int main()

{

using namespace std;

double properties[Max];

double \* pbegin = properties;

double \* pend = fill\_array(pbegin, pbegin + Max); //可以写成下面的形式

//fill\_array(pbegin, pbegin + Max);

//double \* pend = pbegin + Max;

show\_array(pbegin, pend);

if (pend-pbegin > 0)

{

cout << "Enter revaluation factor: ";

double factor;

while (!(cin >> factor))

{

cin.clear();

while (cin.get() != '\n')

continue;

cout << "Bad input; Please enter a number: ";

}

revalue(factor, pbegin, pend);

show\_array(pbegin, pend);

}

cout << "Done.\n";

return 0;

}

double \* fill\_array(double \* begin, double \*end)

{

using namespace std;

double temp;

int i = 1;

while (begin < end)

{

cout << "Enter value #" << i<< ": ";

cin >> temp;

if (!cin)

{

cin.clear();

while (cin.get() != '\n')

continue;

cout << "Bad input; input process terminated.\n";

break;

}

else if (temp <0)

break;

\*begin = temp;

begin++;

i++;

}

return begin;

}

void show\_array(const double \* begin, const double \* end)

{

using namespace std;

int i = 1;

while (begin < end)

{

cout << "Property #" << i << ": $";

cout << \*begin << endl;

begin++;

i++;

}

}

void revalue(double r, double \* begin, double \* end)

{

while (begin < end)

{

\*begin \*= r;

begin++;

}

}

## 7.8

#include <iostream>

const int Seasons = 4;

const char \* Snames[] = {"Spring", "Summer", "Fall", "Winter"};

void fill(double ar[], int n);

void show(double ar[], int n);

int main()

{

using namespace std;

double expenses[Seasons];

fill(expenses, Seasons);

show(expenses, Seasons);

return 0;

}

void fill(double ar[], int n)

{

using namespace std;

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

{

cout << "Enter "<< Snames[i] << "expenses: ";

cin >> ar[i];

}

}

void show(double ar[], int n)

{

using namespace std;

cout << "\nEXPENSES\n";

double total = 0.0;

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

{

cout << Snames[i] << ": $" << ar[i] <<endl;

total += ar[i];

}

cout << "Total Expenses: $" << total << endl;

}

//ex7.8b（传递结构值）

#include <iostream>

const int Seasons = 4;

struct data

{

double arr[Seasons];

};

const char \* Snames[] = {"Spring", "Summer", "Fall", "Winter"};

data fill();

void show(data);

int main()

{

using namespace std;

data expenses = fill();

show(expenses);

return 0;

}

data fill()

{

using namespace std;

data expenses;

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

{

cout << "Enter " << Snames[i] << "expenses: ";

cin >> expenses.arr[i];

}

return expenses;

}

void show(data expenses)

{

using namespace std;

cout << "\nEXPENSES\n";

double total = 0.0;

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

{

cout << Snames[i] << ": $" << expenses.arr[i] <<endl;

total += expenses.arr[i];

}

cout << "Total Expenses: $" << total << endl;

}

## 7.9

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

using namespace std;

const int SLEN = 30;

struct student {

char fullname[SLEN];

char hobby[SLEN];

int ooplevel;

};

int getinfo(student pa[], int n);

void display1(student st);

void display2(const student \* ps);

void display3(const student pa[], int n);

int main()

{

cout << "Enter class size: ";

int class\_size;

cin >> class\_size;

while (cin.get() != '\n')

continue;

student \* ptr\_stu = new student[class\_size];

int entered = getinfo(ptr\_stu, class\_size);

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

{

display1(ptr\_stu[i]);

display2(&ptr\_stu[i]);

}

display3(ptr\_stu, entered);

delete [] ptr\_stu;

cout << "Done\n";

return 0;

}

// getinfo() has two arguments: a pointer to the first element of

// an array of student structures and an int representing the

// number of elements of the array. The function solicits and

// stores data about students. It terminates input upon filling

// the array or upon encountering a blank line for the student

// name. The function returns the actual number of array elements

// filled.

int getinfo(student pa[], int n)

{

int num\_array\_elem =n;

char tmp[SLEN];

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

{

cout << "Enter name: ";

cin.getline(tmp, SLEN);

bool blank\_line = true;

for (unsigned j = 0; j < strlen(tmp); ++j)

{

if (!isspace(tmp[j]))

{

blank\_line =false;

break;

}

}

if (blank\_line)

{

num\_array\_elem =i;

break;

}

strcpy(pa[i].fullname, tmp);

cout << "Enter hobby: ";

cin.getline(pa[i].hobby, SLEN);

cout << "Enter ooplevel: ";

cin >> pa[i].ooplevel;

cin.get();

}

cout << endl;

return num\_array\_elem;

}

// display1() takes astudent structure as an argument

// and displays its contents

void display1(student st)

{

cout << st.fullname << '\t' << st.hobby << '\t' << st.ooplevel <<endl;

}

// display2() takes the address of student structure as an

// argument and displays the structure’¡¥s contents

void display2(const student \* ps)

{

cout << ps->fullname << '\t' << ps->hobby << '\t' << ps->ooplevel<< endl;

}

// display3() takes the address of the first element of an array

// of student structures and the number of array elements as

// arguments and displays the contents of the structures

void display3(const student pa[], int n)

{

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

cout << pa[i].fullname << '\t' << pa[i].hobby << '\t' <<pa[i].ooplevel << endl;

}

## 7.10使用指针数组调用函数（将函数作为指针数组的成员）

#include <iostream>

double calculate(double x, double y, double (\*pf)(double, double));

double add(double x, double y);

double sub(double x, double y);

double mean(double x, double y);

int main()

{

using namespace std;

double a, b;

double (\*pf[3])(double, double) = {add, sub, mean};

char \* op[3] = {"add", "sub", "mean"};

cout << "Enter pairs of numbers (q to quit): ";

while (cin >> a >> b)

{

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

{

cout << op[i] << ": "<< a<< "and "<< b<< "="<< calculate(a, b, pf[i]) << endl; //名称:pf 值:0x0017fae8 类型:double [3](double, double)\*

//名称:pf[0] 值:0x010e10be 类型:add(double,double) \*

}

}

}

}

double calculate(double x, double y, double (\*pf)(double, double))

{

return (\*pf)(x, y);

}

double add(double x, double y)

{

return x+y;

}

double sub(double x, double y)

{

return x-y;

}

double mean(double x, double y)

{

return (x+y) /2.0;

}

## 8.1

#include<iostream>

void silly(const char \* , int n = 0);

int main(void)

{

using namespace std;

char str[10] = "调用";

// call 1

cout << "无参数调试" << endl;

silly(str);

// call 2

cout << "验证无参数调试" << endl;

silly(str);

// call 3

cout << "有参数调试调用" << endl;

silly(str, 1);

return 0;

}

void silly(const char \* str, int n)

{

using namespace std;

static int times = 1;

if(n)

for(int i = 1; i <= times; ++i)

cout << str << endl;

else

cout << str << endl;

times++;

}

## 8.2

#include<iostream>

//#include<string>

struct CandyBar

{

char\* name ;

double weight;

int heat;

};

void set(CandyBar & ,char\* , double , int); //声明可以不写形参

void show(const CandyBar );

int main(void)

{

using namespace std;

CandyBar car;

char \*p=" Millennium Munch";

double x=2.85;

int y=350;

set(car ,p ,x , y);

show(car);

return 0;

}

void set(CandyBar & car , char\* p, double x , int y ) //调用结构内成员，成员必须是参数条件

{

car.name = p;

car.weight = x;

car.heat = y;

}

void show(const CandyBar car) //定义需写形参

{

std::cout<< car.name<<" "<< car.weight << " "<< car.heat<< std::endl;

}

## 8.3

#include <iostream>

#include <string>

#include <cctype>

using namespace std;

void str\_to\_upper(string & str);

int main()

{

string str1;

cout << "Enter a string (q to quit): ";

while (getline(cin, str1) && str1!="q" && str1!="Q")

{

str\_to\_upper(str1);

cout << str1 << endl;

cout << "Next string (q to quit): ";

}

cout << "Bye.";

return 0;

}

void str\_to\_upper(string &str)

{

int limit =str.size();

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

{

if (isalpha(str[i]))

str[i] = toupper(str[i]);

}

}

## 8.4

#include <iostream>

using namespace std;

#include <cstring> // for strlen(),strcpy()

struct stringy{

char \* str;

int ct;

};

void set(stringy &bny,char\* str);

void show(const char\* str, int n=1); //必须记得给n赋初值；

void show(stringy &bny,int n=1);

int main()

{

stringy beany;

char testing[] = "Reality isn't what it used to be.";

set(beany,testing);

show(beany);

show(beany,2);

testing[0] = 'D';

testing[1] = 'u';

show(testing);

show(testing, 3);

show("Done!");

return 0;

}

void set(stringy &bny,char\* str)

{

bny.str= str;

}

void show(stringy &bny,int n) //可以共用一个原型,n值可以没有参数

{

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

cout << bny.str << endl;

}

void show(const char\* str, int n)

{

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

cout << str << endl;

}

#include <iostream>

#include <cstring> // for strlen(),strcpy()

using namespace std;

struct stringy

{

char \* str;

int ct;

};

void set(stringy & bny, const char \* str);

void show(const char \*str, int cnt = 1);

void show(const stringy & bny, int cnt = 1);

int main(void)

{

stringy beany;

char testing[] = "Reality isn't what it used to be.";

set(beany,testing);

show(beany);

show(beany,2);

testing[0] = 'D';

testing[1] = 'u';

show(testing);

show(testing, 3);

show("Done!");

return 0;

}

void set(stringy &bny, const char \*str)

{

bny.ct =strlen(str); //计算需要的内存空间

bny.str = new char[bny.ct+1]; //需要添加结束标志符

//bny.str=str; //指针浅拷贝，const修饰限制

strcpy(bny.str, str);

}

void show(const char \*str, int cnt)

{

while(cnt-- >0)

{

cout << str << endl;

}

}

void show(const stringy &bny, int cnt)

{

while(cnt-- >0)

{

cout << bny.str << endl;

}

}

## 9.5

//#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <cstring> // for strlen(),strcpy()

using namespace std;

const int Limit = 5;

template <class T>

T max5(T ar[]);

int main(void)

{

using namespace std;

int ari[Limit] = {1, 2, 3, 5, 4};

double ard[Limit] = {1.1, 2.2, 3.3, 5.5, 4.4};

int maxi = max5(ari);

double maxd = max5(ard);

cout << "maxi = " << maxi << endl;

cout << "maxd = " << maxd << endl;

return 0;

}

template <typename T>

T max5(T ar[])

{

T max = ar[0];

for (int i=1; i<Limit; i++)

{

if (max <ar[i])

max = ar[i];

}

return max;

}

## 8.6

//#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <cstring> // for strlen(),strcpy()

using namespace std;

template <class T>

T max(T ar[],int n);

//template <> const char\* max<const char \*>(const char \* [], int); //显示具体化

const char\* max(const char \* [], int);

int main(void)

{

using namespace std;

int a[6] = {1, 2, 3, 4, 5, 6};

double b[4] = {1.1, 2.2, 3.3, 4.4};

int maxi = max(a, 6);

double maxd = max(b, 4);

const char \* c[5] = {

"a",

"bb",

"ccc",

"ddddd",

"eeee"

};

cout << "maxi: " << maxi << endl

<< "maxd: " << maxd << endl

<< "The max string of array is: " << max(c, 5) << endl;

return 0;

}

template <typename T>

T max(T ar[],int n)

{

T max = ar[0];

for (int i=1; i<n; i++)

{

if (max <ar[i])

max = ar[i];

}

return max;

}

//template <> const char\* max<const char \*>(const char \*ar[],int n)

const char\* max(const char \* ar[], int n)

{

const char \* maxs =ar[0];

for (int i=1; i<n; i++)

{

if (strlen(maxs) <strlen(ar[i]))

maxs = ar[i];

}

return maxs;

}

## 8.7

#include <iostream>

template <typename T>

T SumArrray(T arr[], int n);

template <typename T>

T SumArrray(T \* arr[], int n);

struct debts

{

char name[50];

double amount;

};

int main()

{

using namespace std;

int things[6] = {13, 31, 103, 301, 310, 130};

struct debts mr\_E[3] = {

{"Ima Wolfe", 2400.0},

{"Ura Foxe", 1300.0},

{"Iby Stout", 1800.0}

};

double \* pd[3];

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

pd[i] = &mr\_E[i].amount;

cout << "Sum: Mr.E's counts of things: "<< SumArrray(things, 6) << endl;

cout << "Sum: Mr.E's debts: "<< SumArrray(pd, 3) << endl;

//cout作为输出流，有一个缓冲区，先从右往左将输出读入缓冲区，在从缓冲区中读出（类似堆栈）

return 0;

}

template <typename T>

T SumArrray(T arr[], int n)

{

using namespace std;

T sum = 0;

cout << "template A\n";

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

sum += arr[i];

return sum;

}

template <typename T>

T SumArrray(T \* arr[], int n)

{

using namespace std;

T sum = 0;

cout << "template B\n";

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

sum += \*arr[i];

return sum;

}

## 14.1

//Pair.h

#pragma once

#ifndef WINEC\_H\_

#define WINEC\_H\_

#include <iostream>

#include <string>

#include <valarray>

using namespace std;

template<class T1, class T2>

class Pair

{

private:

T1 year;

T2 bottles;

public:

Pair(const T1 &yr, const T2 &bt):year(yr), bottles(bt){}

Pair(){}

void Set(const T1 &yr, const T2 &bt);

int Sum()const;

void Show(int y)const;

};

template<class T1, class T2>

void Pair<T1,T2>::Set(const T1 &yr, const T2 &bt)

{

year = yr;

bottles = bt;

}

template<class T1,class T2>

int Pair<T1, T2>::Sum()const

{

return bottles.sum();

}

template<class T1, class T2>

void Pair<T1,T2>::Show(int y)const

{

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

cout<< "\t" << year[i] << "\t" << bottles[i] <<endl;

}

#endif

//winec.h

#include "Pair.h"

typedef std::valarray<int> ArrayInt;

typedef Pair<ArrayInt,ArrayInt> PairArray;

class Wine

{

private:

PairArray yb;

string fullname;

int yrs;

public:

Wine(){}

Wine(const char \*l, int y, const int yr[], const int bot[]);

Wine(const char \*l, int y);

void GetBottles();

string& Label();

void Show()const;

int sum()const;

};

//winec.cpp

#include "winec.h"

Wine::Wine(const char \*l, int y, const int yr[], const int bot[])

{

fullname = l;

yrs = y;

yb.Set(ArrayInt(yr, yrs), ArrayInt(bot, yrs));

}

Wine::Wine(const char \*l, int y)

{

fullname = l;

yrs = y;

}

void Wine::GetBottles()

{

ArrayInt yr(yrs), bt(yrs);

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

{

cout<< "Enter the year: ";

cin>>yr[i];

cout<< "Enter the bottles: ";

cin>>bt[i];

}

while (cin.get() != '\n')

continue;

yb.Set(yr, bt);

}

string&Wine::Label()

{

return fullname;

}

void Wine::Show()const

{

cout<< "Wine: " <<fullname<<endl;

cout<< "\tYear\tBottles\n";

yb.Show(yrs);

}

int Wine::sum()const

{

return yb.Sum();

}

//main.c

#include "winec.h"

int main(void)

{

cout<< "Enter name of wine: ";

char lab[50];

cin.getline(lab, 50);

cout<< "Enter number of years: ";

int yrs;

cin >> yrs;

Wine holding(lab, yrs);

holding.GetBottles();

holding.Show();

const int YRS = 3;

int y[YRS] = { 1993, 1995, 1998 };

int b[YRS] = { 48, 60, 72 };

Wine more("Gushing Grape Red", YRS,y,b);

more.Show();

cout<< "Total bottles for " <<more.Label()

<< ": " <<more.sum() <<endl;

cout<< "Bye\n";

system("pause");

return 0;

}

## 14.2

//winec.h

#pragma once

#ifndef WINEC\_H\_

#define WINEC\_H\_

#include <iostream>

#include <string>

#include <valarray>

using namespace std;

template<class T1, class T2>

class Pair

{

private:

T1 year;

T2 bottles;

public:

Pair(const T1 &yr, const T2 &bt) :year(yr), bottles(bt){}

Pair(){}

void Set(const T1 &yr, const T2 &bt);

int Sum()const;

void Show(int y)const;

};

template<class T1, class T2>

void Pair<T1,T2>::Set(const T1 &yr, const T2 &bt)

{

year = yr;

bottles = bt;

}

template<class T1,class T2>

int Pair<T1, T2>::Sum()const

{

return bottles.sum();

}

template<class T1, class T2>

void Pair<T1,T2>::Show(int y)const

{

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

cout<< "\t" << year[i] << "\t" << bottles[i] <<endl;

}

typedef valarray<int> ArrayInt;

typedef Pair<ArrayInt, ArrayInt> PairArray;

class Wine :private PairArray, private string

{

private:

int yrs;

public:

Wine(){}

Wine(const char \*l, int y, const int yr[], const int bot[]);

Wine(const char \*l, int y);

void GetBottles();

string& Label();

void Show()const;

int sum()const;

};

#endif

//winec.cpp

#include "winec.h"

Wine::Wine(const char \*l, int y, const int yr[], const int bot[]):string(l), yrs(y), PairArray(ArrayInt(yr,y), ArrayInt(bot, y))

{

}

Wine::Wine(const char \*l, int y) : string(l), yrs(y)

{

}

void Wine::GetBottles()

{

ArrayInt yr(yrs),bt(yrs);

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

{

cout<< "Enter the year: ";

cin>>yr[i];

cout<< "Enter the bottles: ";

cin>>bt[i];

}

while (cin.get() != '\n')

continue;

PairArray::Set(yr, bt);

}

string&Wine::Label()

{

return (string &)(\*this);

}

void Wine::Show()const

{

cout<< "Wine: " << (string &)(\*this) <<endl;

cout<< "\tYear\tBottles\n";

PairArray::Show(yrs);

}

int Wine::sum()const

{

return PairArray::Sum();

}

//main.cpp

#include "winec.h"

int main(void)

{

cout<< "Enter name of wine: ";

char lab[50];

cin.getline(lab, 50);

cout<< "Enter number of years: ";

int yrs;

cin>>yrs;

Wine holding(lab, yrs);

holding.GetBottles();

holding.Show();

//const int YRS = 3;

//int y[YRS] = { 1993, 1995, 1998 };

//int b[YRS] = { 48, 60, 72 };

//Wine more("Gushing Grape Red", YRS, y, b);

//more.Show();

//cout<< "Total bottles for " <<more.Label()

//<< ": " <<more.sum() <<endl;

//cout<< "Bye\n";

//system("pause");

return 0;

}