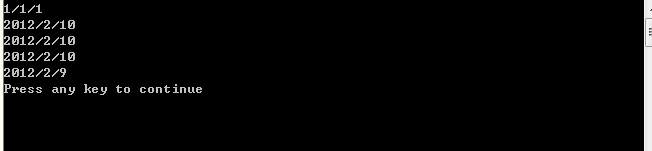
一

1.

1. Date(int y,int m,int d)
2. Date(const Date& dt)
3. Date& operator=(const Date& dt)
4. return temp;
5. 

2

① A operator + 改为friend A operator +

② friend void operator << 改为friend ostream& operator <<

③ void operator 改为ostream & operator

④ return 改为 return out

⑤



二

1

(1)b++;

(2)return \*this;

(3)Complex operator++(int)

(4)Complex tmp = \*this;

(5) 运行截图：（略）

2

①strlen(c) + s.size;

②new char[len +1];

③strcpy(w.str, c);

④strcat(w.str, s.str);

⑤return w;

三

1.

#include <iostream>

#include <string>

using namespace std;

template<typename T>void Sort(T\* a,int n){

int i,j;

T t;

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

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

if (a[j]>a[j+1])

{ t=a[j];

a[j+1]=t;

a[j]=a[j+1];

}

}

template<typename T>void Print(T\* a,int n){

int i;

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

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

cout<<endl;

}

int main(){

int Int[8]={25,12,6,-5,9,-22,6,13};

double Dou[7]={4.08,6.12,-0.35,6,12,-9,8.23};

Sort<int>(Int,8);

Sort<double>(Dou,7);

Print(Int,8);

Print(Dou,7);

return 0;

}

2.

#include <iostream>

using namespace std;

class student {

public:

virtual void printScore() = 0;

};

class senior : public student {

public:

senior(double p, double c) : physics(p), chemistry(c){};

void printScore() {

cout << "senior student score" << endl;

cout << "physics:" << physics << endl;

cout << "chemistry:" << chemistry << endl;

cout << endl;

}

protected:

double physics = 85;

double chemistry = 90;

};

class college : public senior {

public:

college(double p, double c, double oop) : senior(p, c), OOP(oop){};

void printScore() {

cout << "college student score" << endl;

cout << "physics:" << physics << endl;

cout << "chemistry:" << chemistry << endl;

cout << "OOP:" << OOP << endl;

cout << endl;

}

protected:

double OOP = 95;

};

int main() {

student \*pstu;

senior sen(85, 90);

college col(85, 90, 95);

pstu = &sen;

pstu->printScore();

pstu = &col;

pstu->printScore();

return 0;

}

运行结果



3.

（1）

template<class T>

typename Vector<T>::iterator Vector<T>::Insert(iterator itr, const T& item)

{

if (size == max)

{

int n = itr - Begin();

Reserve(2 \* max + 1);

itr = Begin() + n;

}

for (iterator p = data + size, q = data + size - 1; p != itr; --p, --q)

\*p = \*q;

\*itr = item;

size++;

return itr;

}

（2）

template<class T>

typename Vector<T>::iterator Vector<T>::Erase(iterator first, iterator last)

{

while(first != last)

{

Erase(first);

last--;

}

return last;

}

（3）

template<class T>

void Vector<T>::Insert(iterator pos, iterator first, iterator last)

{

for (iterator itr = first; itr != last; itr++)

{

pos = Insert(pos, \*itr);

pos++;

}

}

(4)

template<class T>

void Vector<T>::Swap(Vector<T>& v)

{

Vector<T> temp = v;

v = \*this;

\*this = temp;

}

一个测试参考程序如下：

#include<iostream>

#include"Vector.h"

using namespace std;

template<class Iterator> //在指定范围内输出元素

void display\_vector(Iterator first, Iterator last)

{

for (; first != last; ++first)

cout << \*first << '\t';

cout << endl;

}

int main()

{

Vector<int> V1;

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

{

V1.Push\_back(i);

}

cout << "original V1:\t";

display\_vector(V1.Begin(), V1.End());

Vector<int> V2;

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

{

V2.Push\_back(i \* 2 + 1);

}

cout << " original V2:\t ";

display\_vector(V2.Begin(), V2.End());

V1.Swap(V2);

cout << "After swap, V1:";

display\_vector(V1.Begin(), V1.End());

cout << "V2: ";

display\_vector(V2.Begin(), V2.End());

Vector<int>::iterator itr = V1.Begin()+ 3;

V1.Insert(itr, V2.Begin(), V2.End());

cout << "After insert v2 into v1:\n";

display\_vector(V1.Begin(), V1.End());

V1.Erase(V1.Begin(), V1.Begin() + 5);

cout << "After erase 5 elements, v1:\n";

display\_vector(V1.Begin(), V1.End());

return 0;

}