运算符重载作业：

作业3:

源代码：

// main.cpp

// 作业3

// Created by mac on 16/4/20.

// Copyright © 2016年 mac. All rights reserved.

#include <iostream>

#include <cstring>

using namespace std;

class String{

private:

char \*m\_data;

int length;

public:

String(const char \*str=NULL);

String(const String &other);

~String(){

if(!m\_data)

delete []m\_data;

}

String &operator=(char \*str);

String &operator=(const String &other);

int operator==(String &other);

int operator==(char \*str);

};

String::String(const char \*str){

if(str){

m\_data=new char[strlen(str)+1];

strcpy(m\_data, str);

length=(int)strlen(str);

}else{

m\_data=NULL;

length=0;

}

}

String::String(const String &other){

m\_data=new char[strlen(other.m\_data)+1];

strcpy(m\_data, other.m\_data);

length=other.length;

}

String & String::operator=(char \*str){

if(!str){

if(!m\_data)delete []m\_data;

m\_data=new char[strlen(str)+1];

strcpy(m\_data, str);

length=(int)strlen(str);

}else{

if(!m\_data) delete []m\_data;

m\_data=NULL;

}

return \*this;

}

String & String::operator=(const String &other){

if(&other==this) return \*this;

else{

if(!m\_data)delete []m\_data;

m\_data=new char[strlen(other.m\_data)+1];

strcpy(m\_data, other.m\_data);

length=other.length;

}

return \*this;

}

int String::operator==(String &other){

if(&other==this) return 1;

if(m\_data==NULL&&other.m\_data==NULL) return 1;

else{

if (!m\_data) return 0;

else return strcmp(m\_data, other.m\_data)?0:1;

}

}

int String::operator==(char \*str){

if(m\_data==NULL&&str==NULL) return 1;

else{

if (!m\_data) return 0;

else return strcmp(m\_data, str)?0:1;

}

}

int main(int argc, const char \* argv[]) {

// insert code here...

//std::cout << "Hello, World!\n";

String obj1("Hello,I'm Steve!"),obj2;

obj2=obj1;

if(obj1==obj2){

printf("This two strings are equal!\n");

}else{

cout<<"Not equal"<<endl;

}

return 0;

}

实验运行结果：

**This two strings are equal!**

**Program ended with exit code: 0**

**作业4:**

源代码：

// main.cpp

// 作业4

// Created by mac on 16/4/20.

// Copyright © 2016年 mac. All rights reserved.

#include <iostream>

using namespace std;

class three\_d{

private:

int x,y,z;

public:

three\_d(int i=0,int j=0,int k=0);

~three\_d(){};

three\_d friend operator+(three\_d &,three\_d &);

friend three\_d operator-(three\_d &,three\_d &);

three\_d operator++(){

return three\_d(x+1,y+1,z+1);

}

three\_d operator--(){

return three\_d(x-1,y-1,z-1);

}

void show(){

cout<<"x="<<x<<" y="<<y<<" z="<<z<<endl;

}

};

three\_d::three\_d(int i,int j,int k):x(i),y(j),z(k){}

three\_d operator+(three\_d &a,three\_d &b){

return three\_d(a.x+b.x,a.y+b.y,a.z+b.z);

}

three\_d operator-(three\_d &a,three\_d &b){

return three\_d(a.x-b.x,a.y-b.y,a.z-b.z);

}

int main(int argc, const char \* argv[]) {

// insert code here...

//std::cout << "Hello, World!\n";

three\_d obj1(1,2,3),obj2(7,8,9);

(obj2-obj1).show();

return 0;

}

实验运行结果：

**x=6 y=6 z=6**

**Program ended with exit code: 0**

**运算符重载实验：**

实验（1）：

源代码：

// main.cpp

// 运算符重载实验（1）

// Created by mac on 16/5/13.

// Copyright © 2016年 mac. All rights reserved.

#include <iostream>

using namespace std;

class complex {

public: //外部接口

complex(double r=0.0,double i=0.0){real=r;imag=i;}

complex operator + (complex c2); //+重载为成员函数

complex operator - (complex c2); //-重载为成员函数

void display( );

private:

double real;

double imag;

};

complex complex::operator+(complex c2){

return complex(real+c2.real,imag+c2.imag);

}

complex complex::operator-(complex c2){

return complex(real-c2.real,imag-c2.imag);

}

void complex::display(){

if(imag==0)

cout<<"Current complex's value is:"<<real<<endl;

else if(imag>0)

cout<<"Current complex's value is:"<<real<<"+"<<imag<<"i"<<endl;

else

cout<<"Current complex's value is:"<<real<<imag<<"i"<<endl;

}

int main(void){

complex c1(3,5);

complex c2(5,-1);

complex c3=c1+c2;

c3.display();

(c1-c2).display();

return 0;

}

实验运行结果：

**Current complex's value is:8+4i**

**Current complex's value is:-2+6i**

**Program ended with exit code: 0**

**实验（2）**

**程序源代码：**

// main.cpp

// 运算符重载实验（2）

// Created by mac on 16/5/13.

// Copyright © 2016年 mac. All rights reserved.

#include <iostream>

using namespace std;

class complex

{

public:

complex(double r=0.0,double i=0.0){

real=r; imag=i;

}//构造函数

friend complex operator + (complex c1,complex c2); //重载运算符+为友元函数

friend complex operator - (complex c1,complex c2); //重载运算符-为友元函数

void display(); //显示复数的值

private:

double real;

double imag;

};

complex operator+(complex c1,complex c2){

return complex(c1.real+c2.real,c1.imag+c2.imag);

}

complex operator-(complex c1,complex c2){

return complex(c1.real-c2.real,c1.imag-c2.imag);

}

void complex::display(){

if(imag==0)

cout<<"Current complex's value is:"<<real<<endl;

else if(imag>0)

cout<<"Current complex's value is:"<<real<<"+"<<imag<<"i"<<endl;

else

cout<<"Current complex's value is:"<<real<<imag<<"i"<<endl;

}

int main(int argc, const char \* argv[]) {

// insert code here...

complex c1(5,3);

complex c2(1,7);

complex c3=c1+c2;

c3.display();

(c1-c2).display();

return 0;

}

**实验运行结果：**

**Current complex's value is:6+10i**

**Current complex's value is:4-4i**

**Program ended with exit code: 0**

**实验（3）与作业3内容相同在此不做赘述。**

**4.安全数组**

**源代码：**

// main.cpp

// 4安全数组

// Created by mac on 16/5/13.

// Copyright © 2016年 mac. All rights reserved.

#include <iostream>

#include <cstdlib>

using namespace std;

const int SIZE = 3;

class atype {

int a[SIZE];

public:

atype( ) {

int i;

for(i=0; i<SIZE; i++) a[i] = i;

}

int &operator[](int i);

};

int &atype::operator[](int i)

{

if(i<0 || i> SIZE-1) {

cout << "\nIndex value of ";

cout << i << " is out-of-bounds.\n";

exit(1);

}

return a[i];

}

int main( )

{

atype ob;

cout << ob[2]; // 输出 2

cout << " ";

ob[2] = 25; // 下标运算符[]出现在赋值运算符的左边

cout << ob[2]; // 输出 25

ob[3] = 44; // 产生运行时错误，下标3超出了数组边界

return 0;

}

**程序运行结果：**

**2 25**

**Index value of 3 is out-of-bounds.**

**Program ended with exit code: 1**