1.2.

Code:

#include <iostream>

#include <iomanip>

#include <cstring>

using namespace std ;

class String

{

private:

char \*s ;

public:

String(const char \*a = "")

{

int i = strlen(a) ;

s = new char[i+1] ;

strcpy(s, a) ;

cout << "构造：" << s << endl ;

} ;

String(String &a)

{

int i = strlen(a.s) ;

s = new char[i+1] ;

strcpy(s, a.s) ;

cout << "复制构造：" << s << endl ;

}

~String()

{

cout << "析构:" << s << endl ;

delete []s ;

}

void Set(const char \*a)

{

int i = strlen(a) ;

s = new char[i+1] ;

strcpy(s, a) ;

}

void Print()

{

cout << s << endl ;

}

String operator + (String& s1) ;

String& operator = (String& s1) ;

String& operator += (String& s1) ;

friend ostream& operator << (ostream& output, String& s) ;

} ;

String& String:: operator = (String& s1)

{

char \*temp = new char[strlen(s1.s) + 1] ;

strcpy(temp, s1.s) ;

delete s ;

s = new char[strlen(temp) + 1] ;

strcpy(s, temp) ;

delete temp ;

return \*this ;

}

String String:: operator + (String& s1)

{

String sum("s") ;

delete sum.s ;

sum.s = new char[strlen(s) + strlen(s1.s) + 1] ;

strcpy(sum.s, s) ;

strcat(sum.s, s1.s) ;

return sum ;

}

String& String:: operator += (String& s1)

{

char \*temp1 = new char[strlen(s) + 1], \*temp2 = new char[strlen(s1.s) + 1] ;

strcpy(temp1, s) ;

strcpy(temp2, s1.s) ;

delete s ;

s = new char[strlen(temp1) + strlen(temp2) + 1] ;

strcpy(s, temp1) ;

delete temp1 ;

strcat(s, temp2) ;

return \*this;

}

ostream& operator << (ostream& output, String& s)

{

output << s.s ;

return output ;

}

void main()

{

String s1("S1"), s2("S2"), s3("S3"), s4("S4"), s5 ;

s5.Set("666") ;

s5.Print() ;

s4 = s2 ;

cout << "s4 = s2 = " << s4 << endl ;

cout << s2 ;

s4 = s1 + s2 + s4 ;

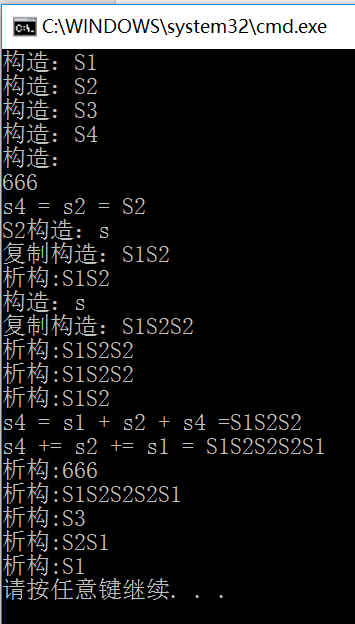
cout << "s4 = s1 + s2 + s4 =" << s4 << endl ;

s4 += s2 += s1 ;

cout << "s4 += s2 += s1 = " << s4 << endl ;

}

Result:



3.

Code:

#include <iostream>

#include <iomanip>

#include <cstring>

using namespace std ;

class String

{

private:

char \*s ;

public:

String(const char \*a)

{

int i = strlen(a) ;

s = new char[i+1] ;

strcpy(s, a) ;

cout << "构造：" << s << endl ;

} ;

String(String &a)

{

int i = strlen(a.s) ;

s = new char[i+1] ;

strcpy(s, a.s) ;

cout << "复制构造：" << s << endl ;

}

~String()

{

cout << "析构:" << s << endl ;

}

void Set(const char \*a)

{

int i = strlen(a) ;

s = new char[i+1] ;

strcpy(s, a) ;

}

void Print()

{

cout << s << endl ;

}

friend String operator + (String& s1, String& s2) ;

String& operator = (String& s1) ;

friend ostream& operator << (ostream& output, String& s) ;

} ;

String& String:: operator = (String& s1)

{

char \*temp = new char[strlen(s1.s) + 1] ;

strcpy(temp, s1.s) ;

delete s ;

s = new char[strlen(temp) + 1] ;

strcpy(s, temp) ;

delete temp ;

return \*this ;

}

String operator + (String& s1, String& s2)

{

String sum("s") ;

delete sum.s ;

sum.s = new char[strlen(s1.s) + strlen(s2.s) + 1] ;

strcpy(sum.s, s1.s) ;

strcat(sum.s, s2.s) ;

return sum ;

}

ostream& operator << (ostream& output, String& s)

{

output << s.s ;

return output ;

}

void main()

{

String s1("S1"), s2("S2"), s3("S3"), s4("S4") ;

s4 = s2 ;

cout << "s4 = s2 = " << s4 << endl ;

s4 = s1 + s2 + s4 ;

cout << "s4 = s1 + s2 + s4 =" << s4 << endl ;

}

Result:

