



北京大学
PEKING UNIVERSITY

信息科学技术学院

程序设计实习

郭炜 微博 <http://weibo.com/guoweiofpku>

<http://blog.sina.com.cn/u/3266490431>

刘家瑛 微博 <http://weibo.com/pkuliujiaying>



北京大学
PEKING UNIVERSITY

信息科学技术学院《程序设计实习》 郭炜 刘家瑛

流插入运算符和流提取运算符的重载

(教材P218)

问题

- `cout << 5 << “this” ;`

为什么能够成立？

- `cout` 是什么？

“<<” 为什么能用在 `cout` 上？

流插入运算符的重载

- `cout` 是在 `iostream` 中定义的, `ostream` 类的对象。
- “<<” 能用在 `cout` 上是因为, 在 `iostream` 里对 “<<” 进行了重载。
- 考虑, 怎么重载才能使得 `cout << 5;` 和 `cout << “this”` 都能成立?

流插入运算符的重载

- 有可能按以下方式重载成 ostream 类的成员函数：

```
void ostream::operator<<(int n)
{
    ..... //输出n的代码
    return;
}
```

流插入运算符的重载

`cout << 5 ;` 即 `cout.operator<<(5);`

`cout << "this";` 即 `cout.operator<<("this");`

○ 怎么重载才能使得

`cout << 5 << "this" ;`

成立？

流插入运算符的重载

```
ostream & ostream::operator<<(int n)
{
    ..... //输出n的代码
    return * this;
}
```

```
ostream & ostream::operator<<( const char * s )
{
    ..... //输出s的代码
    return * this;
}
```

流插入运算符的重载

cout << 5 << “this”;

本质上的函数调用的形式是什么？

cout.operator<<(5).operator<<(“this”);

流插入运算符的重载

- 假定下面程序输出为 5hello, 该补写些什么

```
class CStudent{  
    public: int nAge;  
};  
  
int main(){  
    CStudent s ;  
    s.nAge = 5;  
    cout << s << "hello";  
    return 0;  
}
```

流插入运算符的重载

```
ostream & operator<<( ostream & o, const CStudent & s){  
    o << s.nAge ;  
    return o;  
}
```

例题(教材P218)

假定c是Complex复数类的对象，现在希望写“`cout << c;`”，就能以“`a+bi`”的形式输出c的值，写“`cin>>c;`”，就能从键盘接受“`a+bi`”形式的输入，并且使得`c.real = a, c.imag = b`。

例题

```
int main() {  
    Complex c;  
    int n;  
    cin >> c >> n;  
    cout << c << ", " << n;  
    return 0;  
}
```

程序运行结果可以如下：

13.2+133i 87 ✓
13.2+133i, 87

```
#include <iostream>
#include <string>
#include <cstdlib>

using namespace std;

class Complex {
    double real,imag;
public:
    Complex( double r=0, double i=0):real(r),imag(i){ };
    friend ostream & operator<<( ostream & os,
                                const Complex & c);
    friend istream & operator>>( istream & is,Complex & c);
};

ostream & operator<<( ostream & os,const Complex & c)
{
    os << c.real << "+" << c.imag << "i"; //以"a+bi"的形式输出
    return os;
}
```

```
istream & operator>>( istream & is,Complex & c)
{
    string s;
    is >> s; //将"a+bi"作为字符串读入,“a+bi”中间不能有空格
    int pos = s.find("+",0);
    string sTmp = s.substr(0,pos); //分离出代表实部的字符串
    c.real = atof(sTmp.c_str()); //atof库函数能将const char*指针指向的内容转换成
    float
    sTmp = s.substr(pos+1, s.length()-pos-2); //分离出代表虚部的字符串
    c.imag = atof(sTmp.c_str());
    return is;
}
```

```
int main()
{
    Complex c;
    int n;
    cin >> c >> n;
    cout << c << ", " << n;
    return 0;
}
```

程序运行结果可以如下：

$$\frac{13.2+133i \quad 87 \checkmark}{13.2+133i, \quad 87}$$