* 1. cout判断输入数据类型

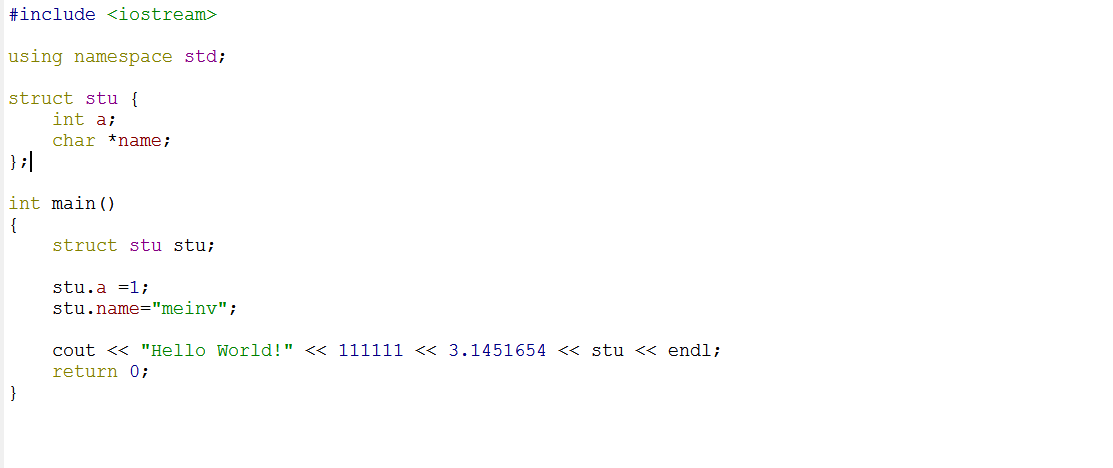
Q1:Cout如何判断输入数据的类型？

Cout的本质是对象，而操作符实际是Cout对象的一个成员函数，即Cout == Cout.operator << (类型)

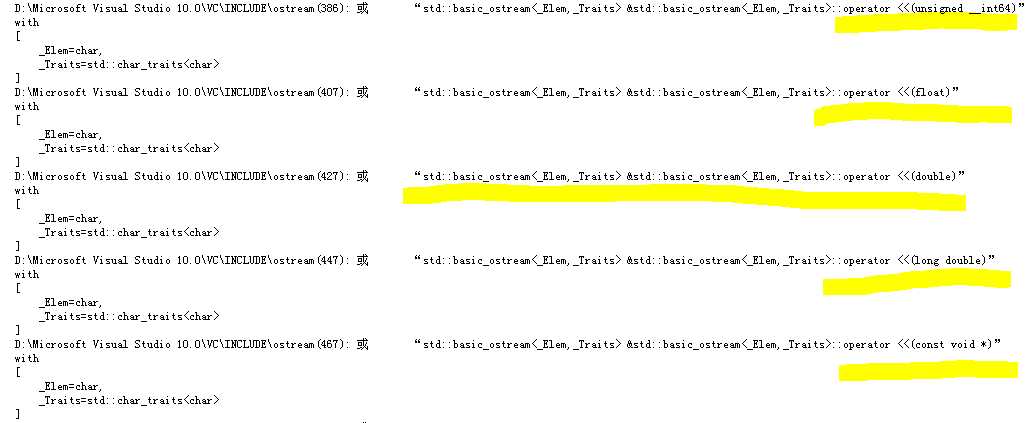
利用C++函数的可重载性，Cout可以即可实现各种输入数据的自动匹配。如：Cout.operator(int )，Cout.operator << (double)等等，实际也是如此。

如，下图是故意向Cout输入一个结构体变量时，编译器的报错信息，可以看到标准流库中，关于operator成员函数的重载。

构造的错误代码：



编译器（win7 64bit-Qtcreater2.5）报错：



* 1. 引用

引用本质是常量指针(对多数编译器来说)，创建引用的目的之一是规避指针可变性带来的风险。

所以，从C的角度理解引用，其本质是常量指针；从C++的角度理解引用

，它可以比喻成变量的别名，且编译器不会为它分配空间。

* 1. 类和对象
     1. 类和对象的区别

在C++里面，类和对象的区别在于编译器是否会为其分配空间，显然，类不会，而对象会。

推广到现实生活中，星球是类，地球是对象；大熊猫是类，成都动物园的大熊猫小美是对象，也就是说，类是存在意识中的概念或者定义，它不占据现实的物质空间，而对象必然是某种物质空间，或者说对象必然占据了一定的物质空间，比如地球，比如小美。

* + 1. 如何定义一个类

<C++里面，如何定义类（思想向）.xmind>

* 1. 类的成员函数

类的成员函数本质是个变量——指针变量，这个指针变量的初始化是在对象被创建时，就好比C结构体中定义的一个函数指针一样，只不过C中需要手动赋值，而C++由编译器自动完成。

C中函数指针的运用：

struct C\_STU {

int age;

string name;

void (\*c\_eat) (string );

};

void fun2(void)

{

C\_STU cstu1; //cpp支持不带struct关键词定义结构体（C中不允许）

C\_STU &cstu1\_1 = cstu1;

struct C\_STU cstu2;

cstu1.age = 20;

cstu1.c\_eat = c\_eat;

cstu1.c\_eat("cstu1 eat: 西红柿");

cstu2.age = 30;

cstu2.c\_eat = c\_eat;

cstu2.c\_eat("cstu2 eat: 红薯");

//对结构体进行引用

cstu1\_1.age = 20;

cstu1\_1.c\_eat = c\_eat;

cstu1\_1.c\_eat("cstu1\_1（引用） eat: 西红柿");

}