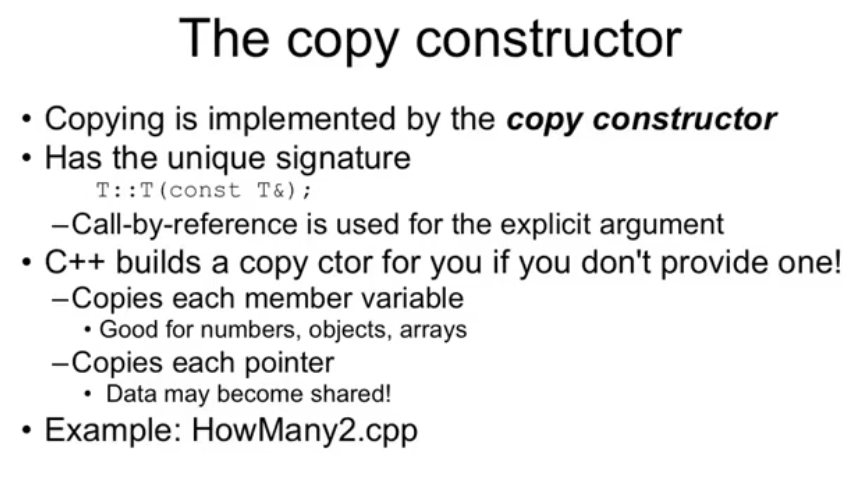


初始化 vs 赋值？



#include <iostream>

using namespace std;

static int objectCount = 0;

class HowMany{

public:

HowMany(){

objectCount++; print("HowMany()");

}

void print(const string& msg=""){

if(msg.size() != 0) cout << msg << ": ";

cout << "objectCount = "

<< objectCount << endl;

}

~HowMany() {

objectCount --;

print("~HowMany()");

}

};

//# pass and return by value

HowMany f(HowMany x){

cout << "begin of f" << endl;

x.print("x argument inside f()");

cout << "end of f" << endl;

return x;

}

int main(){

HowMany h;

h.print("after cosntruction of h");

HowMany h2 = f(h);

h.print("after call to f()");

return 0;

}

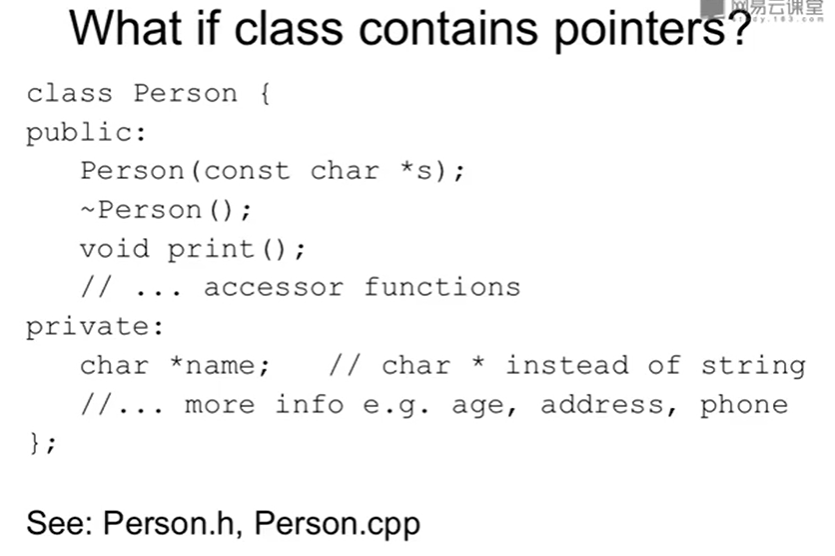
构造函数

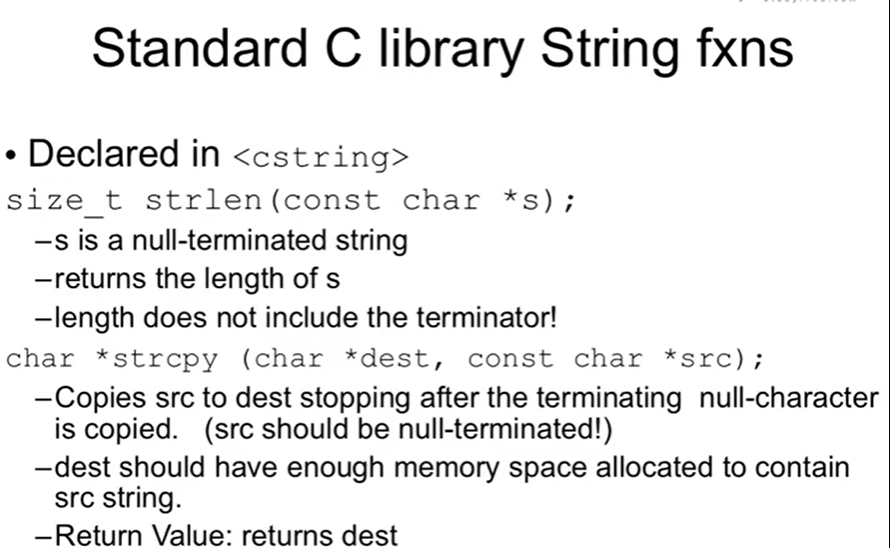
拷贝构造函数（签名固定***T::T(const T&)***，如果没有指定blabla，则）， 如果没有const，则构造会进入**递归**。

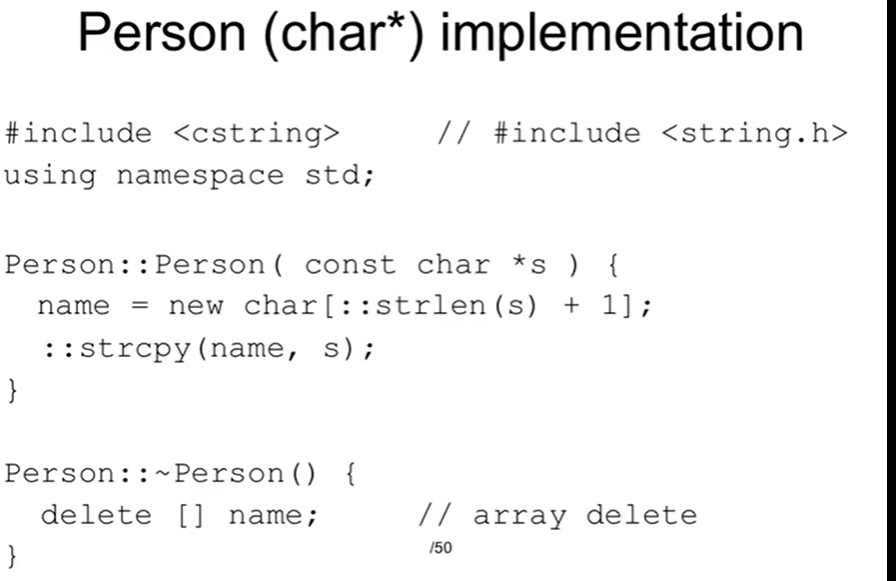
没有const，则其传递方式为传值，会构造一个与传入参数一样内容的一个对象【调用拷贝构造】；如果有const，则

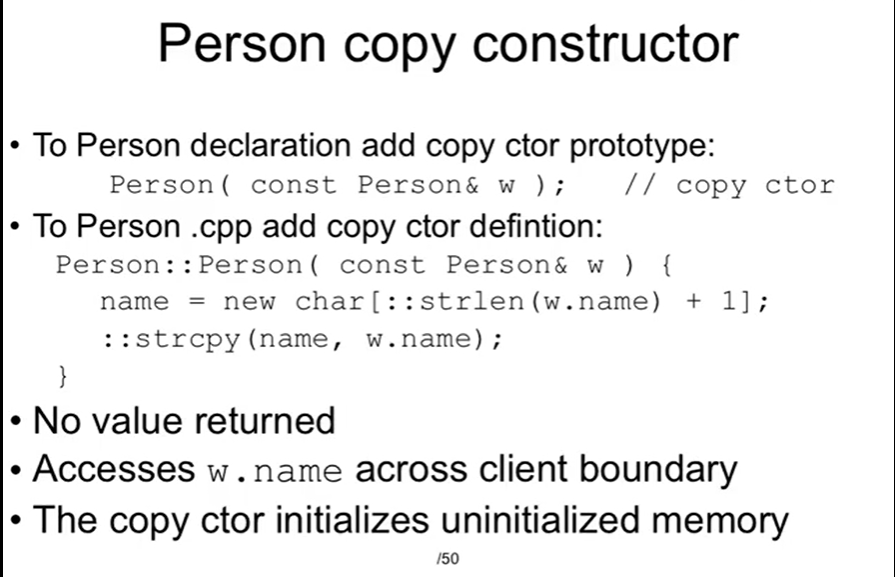
【这个是const还是 &使得避免陷入递归？】

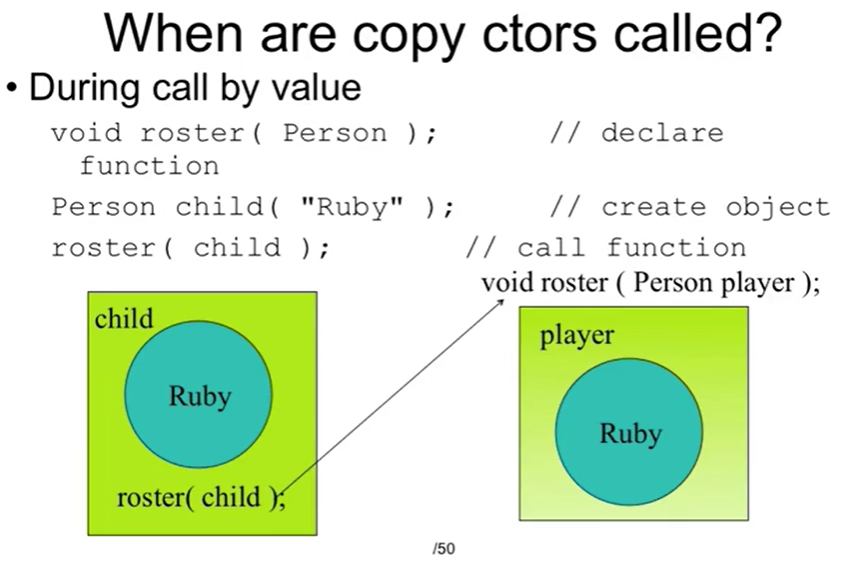
# # 26 拷贝构造Ⅱ

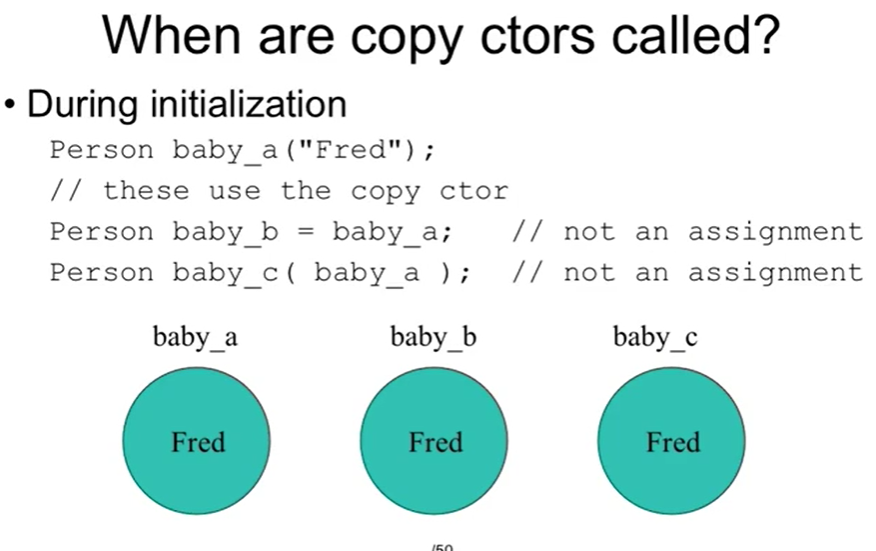




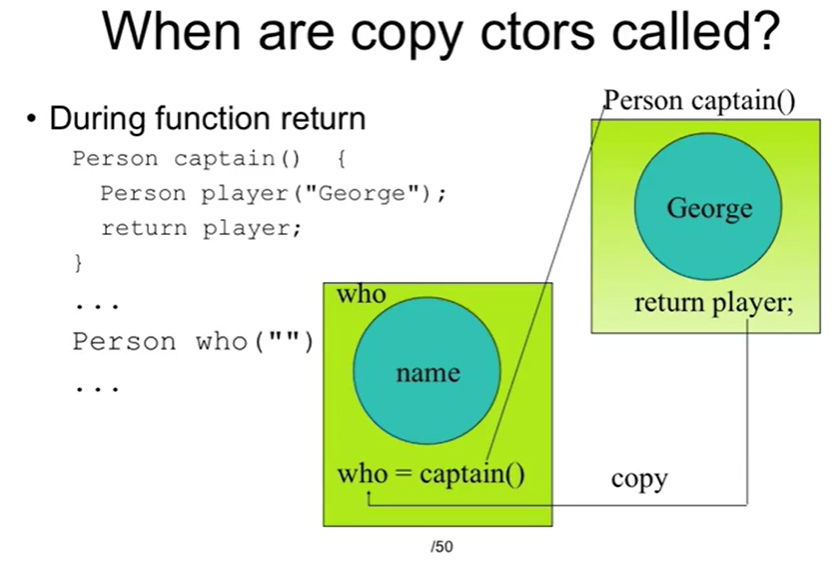








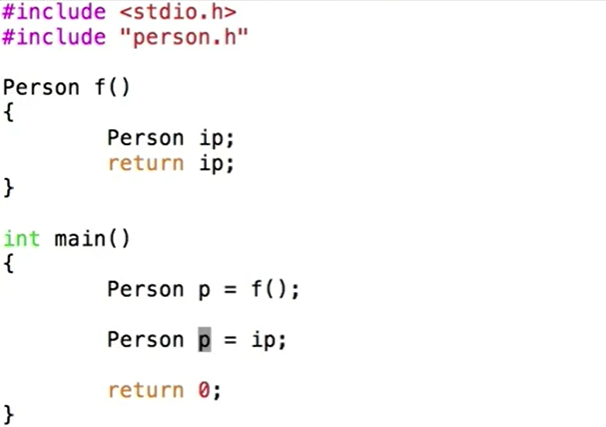
* 对象A = 对象B， 同一个类型则进行的是**copy ctr**拷贝构造
* C A(B), copy ctr

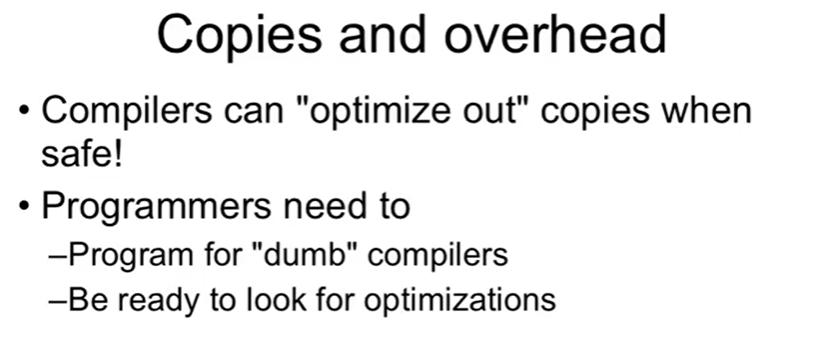


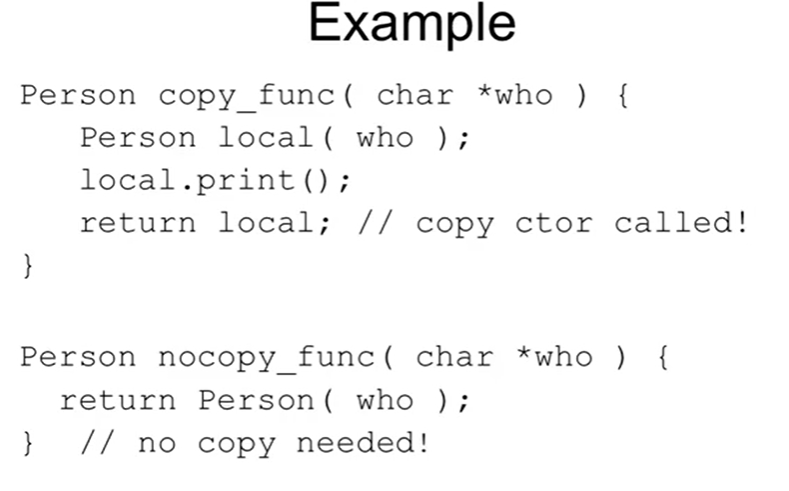
拷贝构造发生的场景（时机）

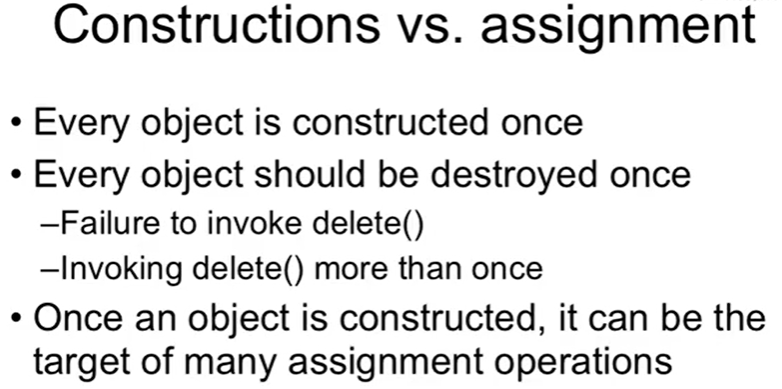
1. **函数调用（传参时）**
2. **函数返回值**
3. **变量初始化**的时候，以一个对象初始化一个新的对象

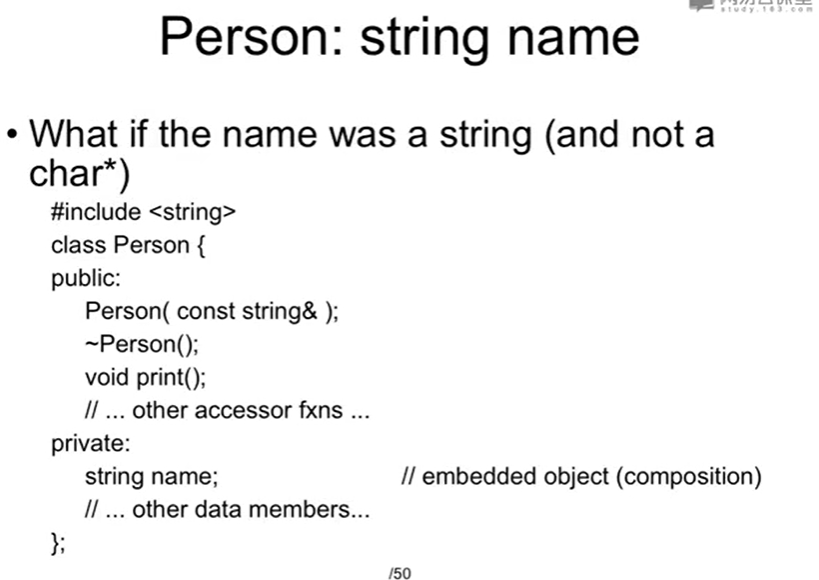
拷贝构造**发生几次**？



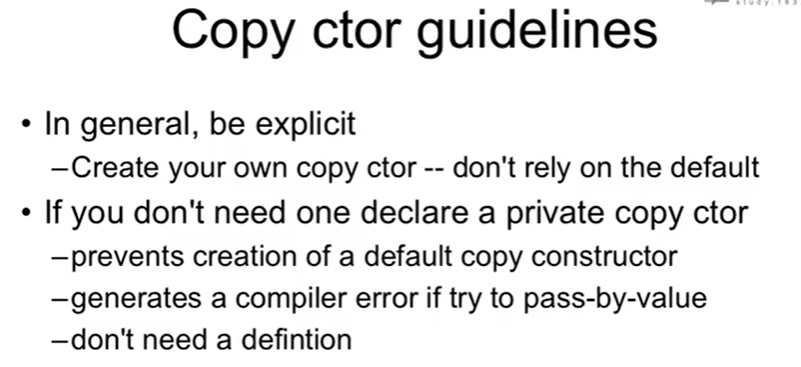








碰到string 对象，会自动完成string对象的拷贝。



写一个CPP类时，应该一开始就添加以下**三个函数**：

1. default construct
2. virtual destructor
3. copy constructor

构造/析构/拷贝构造