

面向对象方法与C++程序设计

第2章

类与对象

大连理工大学
主讲人-赵小薇



拷贝构造函数



```
int i=100;
```

开辟一个整型变量空间同时置其值为100。

- 对于一个对象，也可以定义用一个已经存在的对象进行初始化。这种初始化的方式需要调用拷贝构造函数来实现。
- 若不定义拷贝构造函数，则系统自动生成默认的拷贝构造函数，把已经存在对象的数据按位复制到新生成对象的空间。



举例



```
Point p1("home",1.0,2.0),p2("school",3.0),p3;
```

类的成员函数空间

Point

Point(char *,double,double)

~Point()

disp()

对象空间

p1

x	1.0
y	2.0
name	

p2

x	1.0
y	2.0
name	

p3

x	0.0
y	0.0
name	

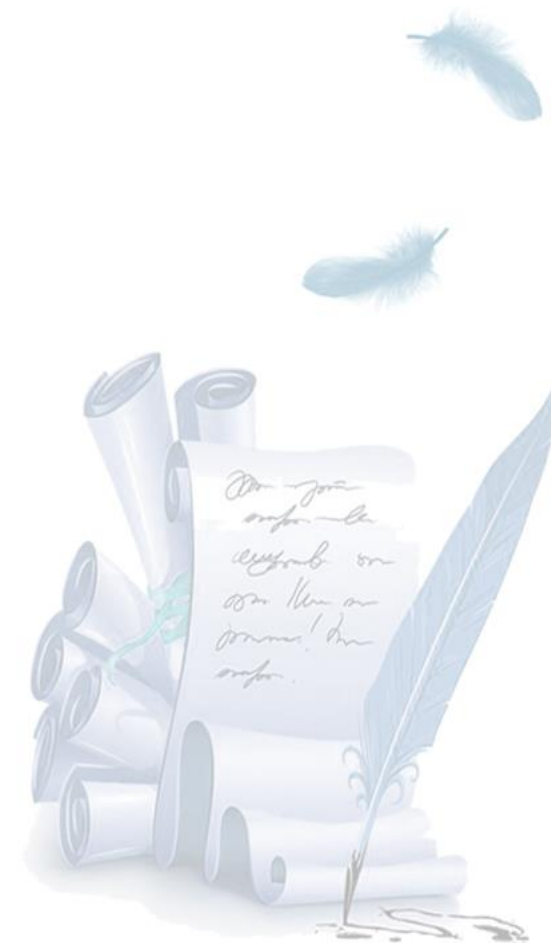
堆空间

h
o
m
e
\0

s
c
h
o
o
l
\0

n
o

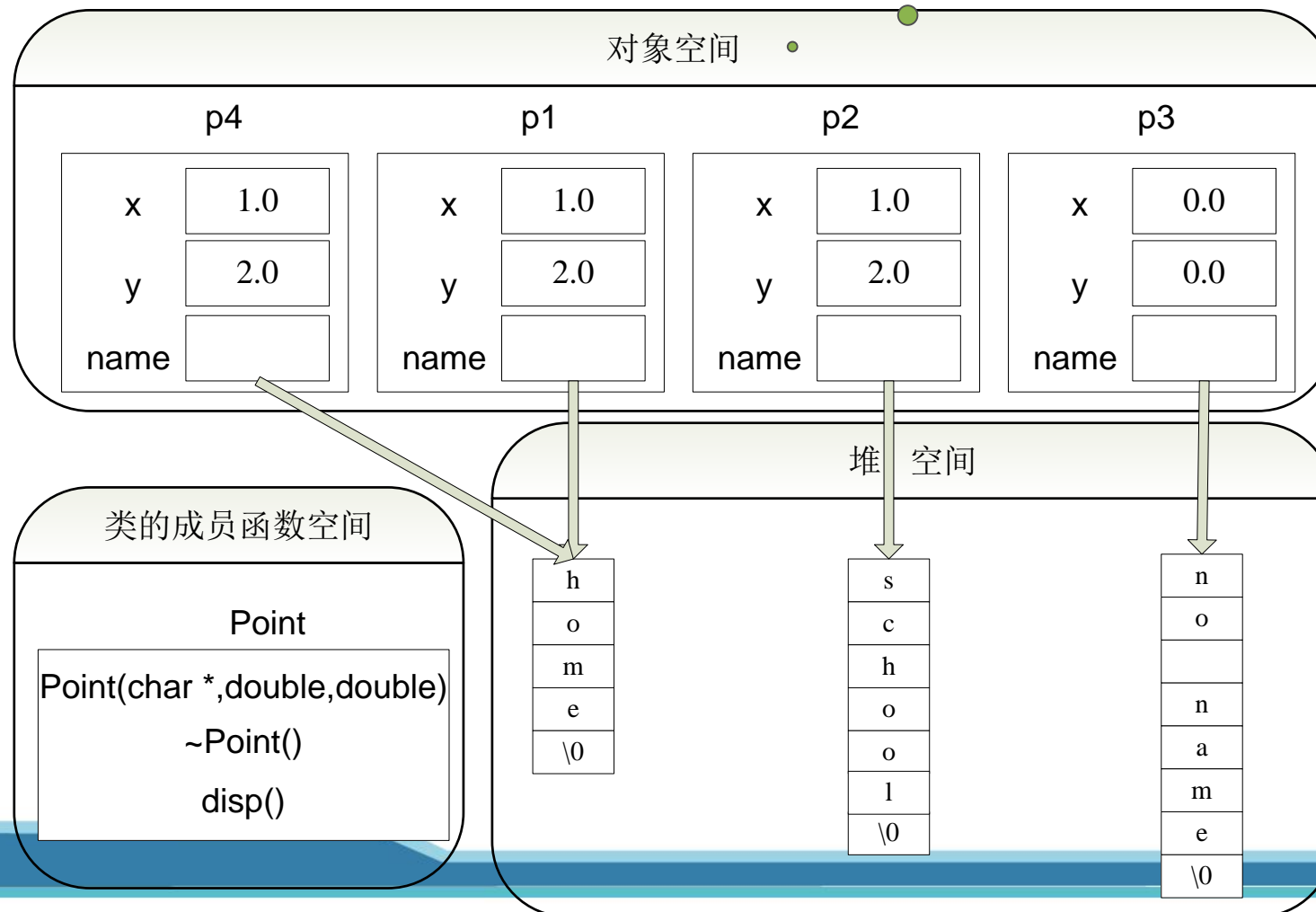
n
a
m
e
\0





```
Point p1("home",1.0,2.0),p2("school",3.0),p3;  
Point p4=p1 ;
```

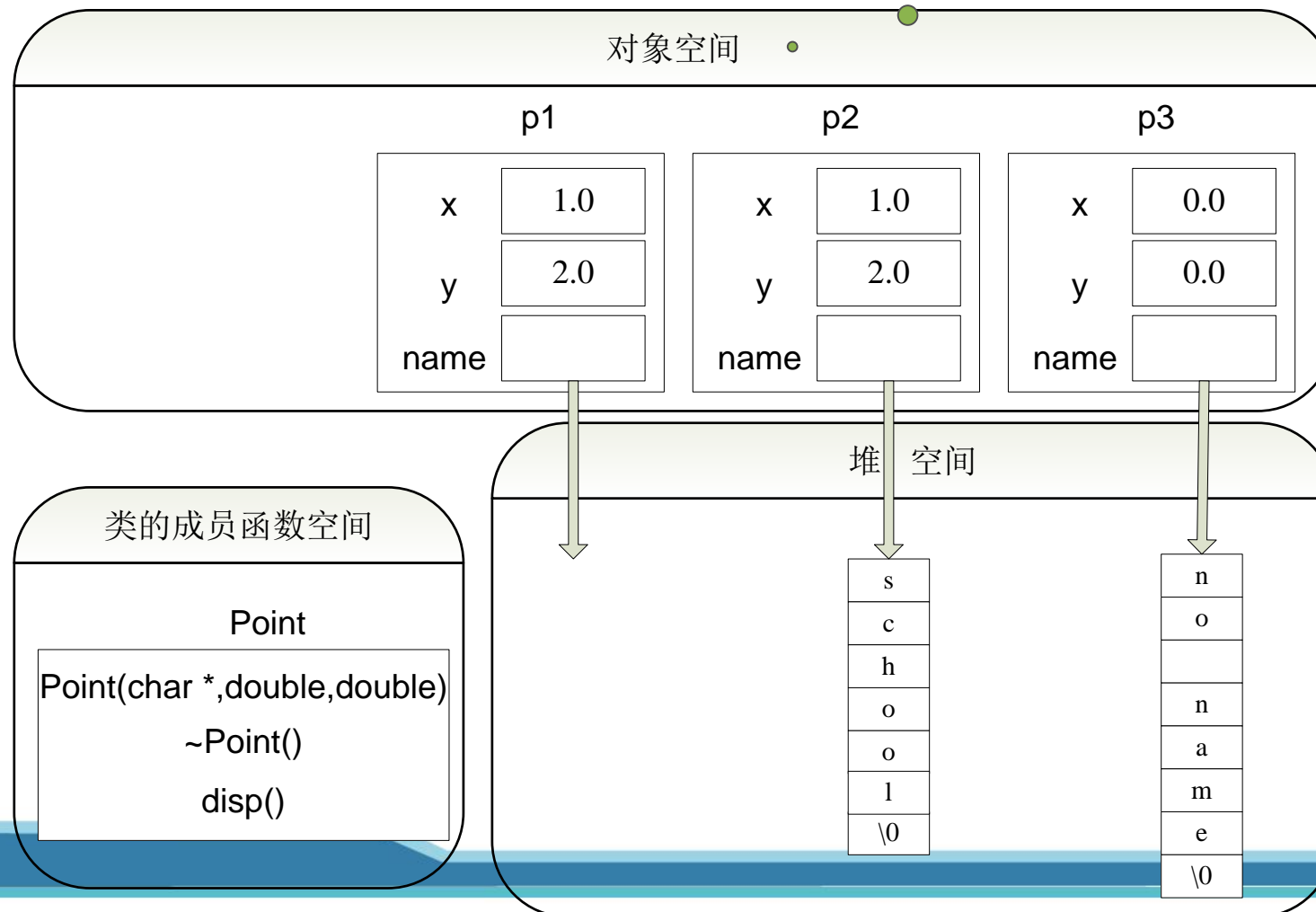
怎样析构?





```
Point p1("home",1.0,2.0),p2("school",3.0),p3;  
Point p4=p1 ;
```

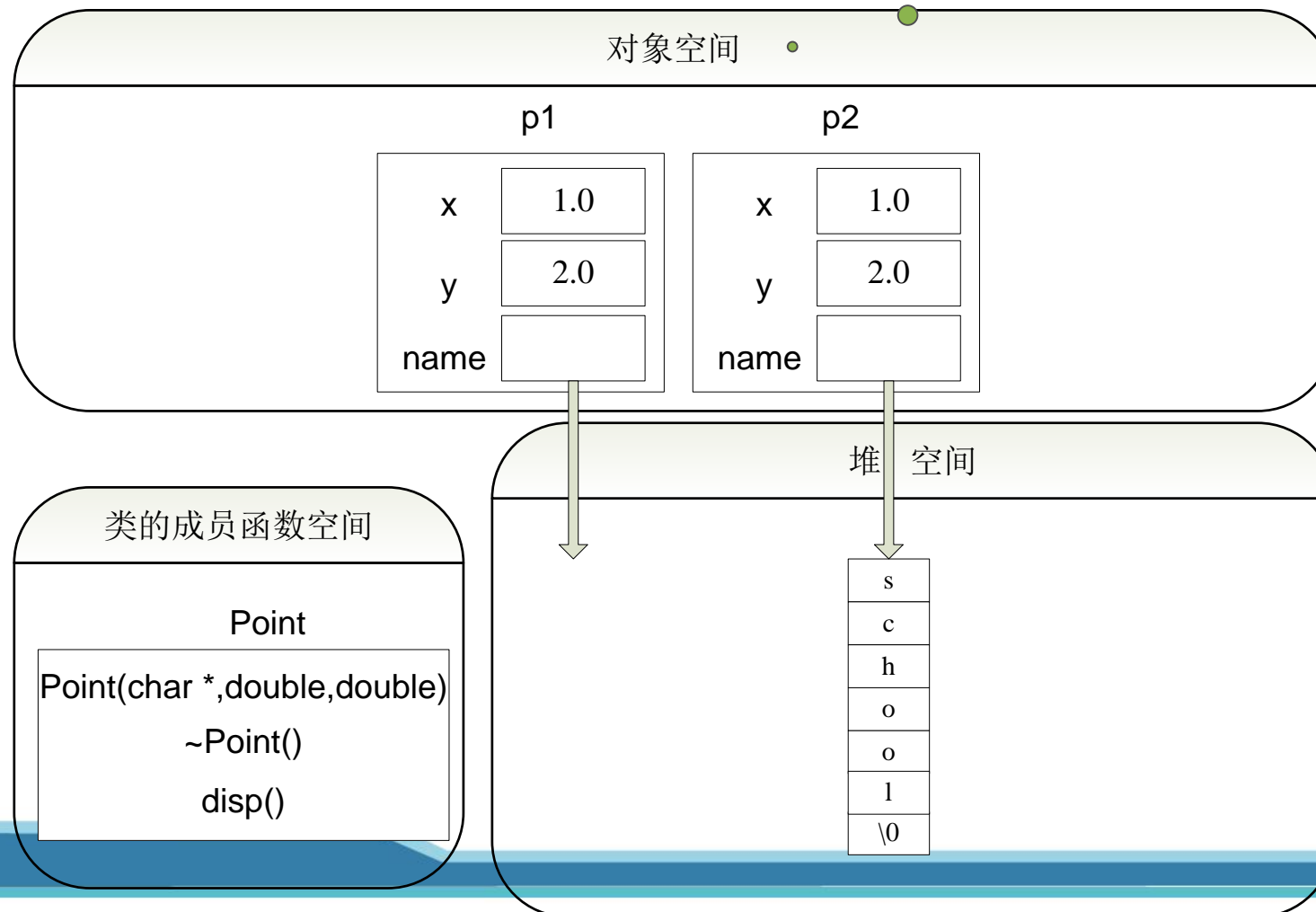
p4析构





```
Point p1("home",1.0,2.0),p2("school",3.0),p3;  
Point p4=p1 ;
```

p3析构





```
Point p1("home",1.0,2.0),p2("school",3.0),p3;  
Point p4=p1 ;
```

p2析构

对象空间

p1

x	1.0
y	2.0
name	

堆 空间

类的成员函数空间

Point

Point(char *,double,double)

~Point()

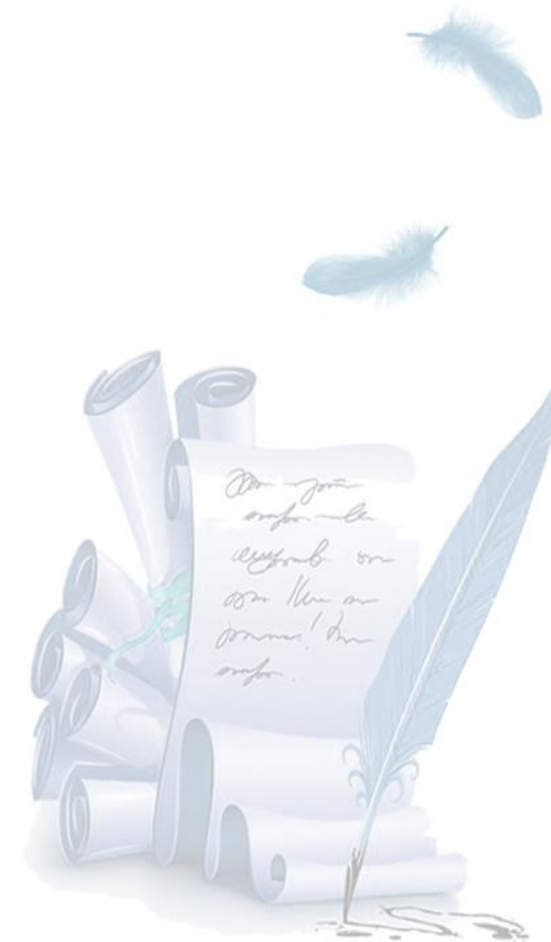
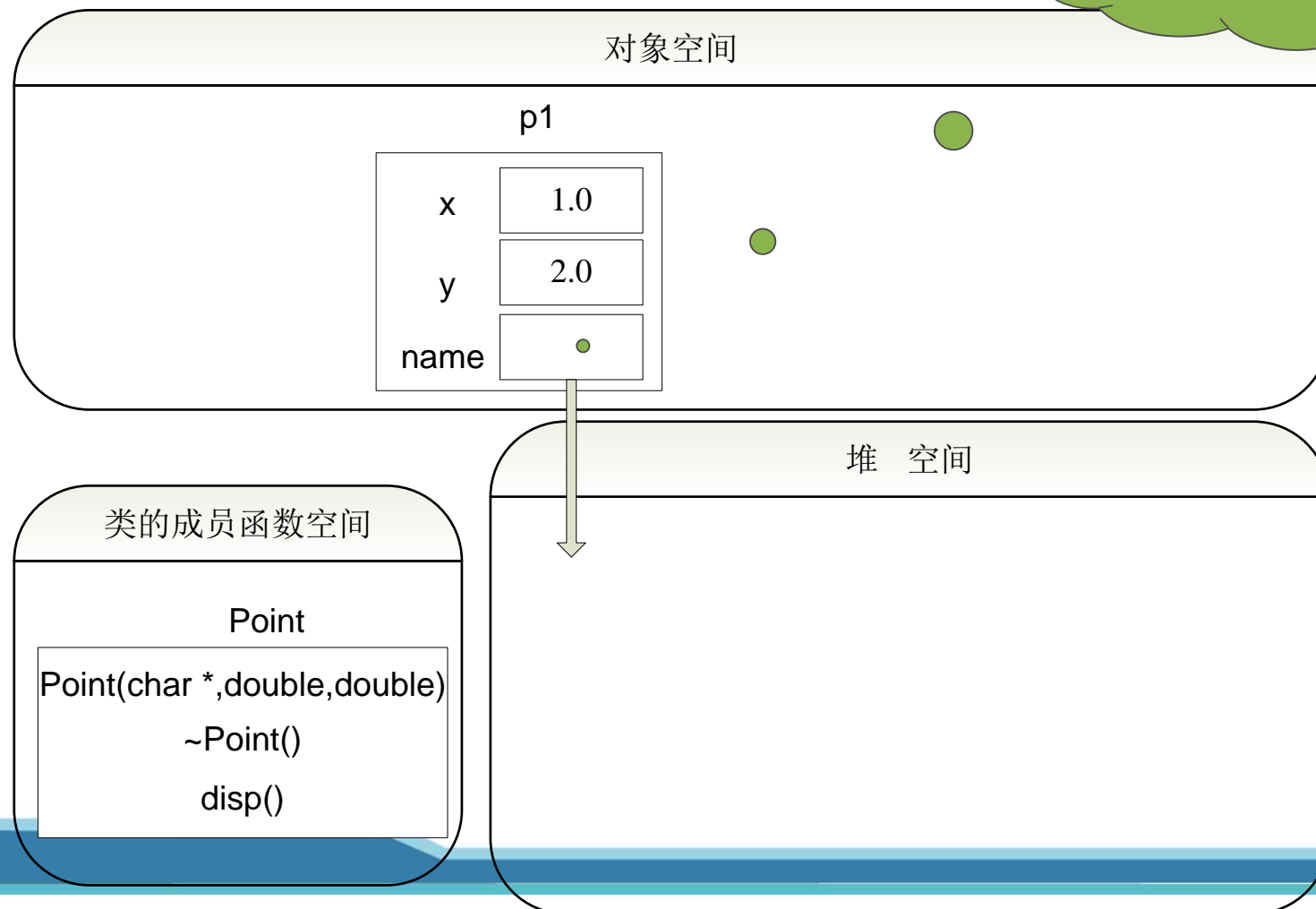
disp()





Point p1("home",1.0,2.0),p2("school",3.0),p3;
Point p4=p1 ;

p1析构
delete[] name?



拷贝构造函数一般形式



析构带来的问题，需要自定义拷贝构造函数，不能够用按位赋值。拷贝构造函数的语法格式为：

```
class 类名
{
public:
    类名( 类名& 形参 )
    { 拷贝构造函数的函数体 }
    .....
};
```

不定义拷贝构造函数，
系统会自动生成一个
默认的拷贝构造函数：
类名(类名& 形参)
{ }

函数名与类名同

引用必须，后面介绍



拷贝构造函数的调用



- 构造函数是当定义一个新对象时自动调用的无到有的过程;
- 拷贝构造函数利用一个已经存在的对象再生成一个新对象;
`Point p1("home",1.0,2.0);`//对象p1生成时没参照现有对象
`Point p2=p1;` //也可以写作 `Point p2 (p1);`

而对象p2生成时参照p1生成的，也称作用p1初始化p2。

`Point p3;`
`p3=p1;`



P3调用拷贝构造吗?





```
Point p1("home",1.0,2.0),p2("school",3.0),p3;  
Point p4=p1 ;
```

p4

x	1.0
y	2.0
name	

类的成员函数

```
Point  
Point(char *,double  
Point(Point &)  
~Point()  
disp()
```

```
Point::Point(Point &p){ //拷贝构造函数  
    x = p.x; y = p.y;  
    if(p.name) { //如果形参不是空值  
        name=new char[strlen(p.name)+1];  
        strcpy(name,p.name);  
    }else{  
        name=new char[8];  
        strcpy(name,"no name");  
    }  
    cout<<name<<" copy constructing";  
}
```

m
e
\0

