C++类模板

1、类模板的格式为：

template<class 形参名，class 形参名，…> class 类名

{ ... };

如

template<class T>

class A

{

public:

T a;

T b;

T hy(T c, T &d);

};

类模板中有默认参数类型的情况

template<class T=int>

class A

{};

类模板中有两种参数的情况

template<class T1,class T2>

class myclass

{};

2.类模板对象的创建：

A<int> m;在类A后面跟上一个<>尖括号并在里面填上相应的类型，这样的话类A中凡是用到模板形参的地方都会被int 所代替。当类模板有两个模板形参时创建对象的方法为A<int, double> m;类型之间用逗号隔开。

3.类模板对象的继承：

继承时候要明确类型

1)模板类继承模板类

template<class T1>

class A

{

public:

A(T1 t1):x(t1)

{};

T1 x;

};

template<class T2>

class B:public A<T2> //类模板类型会传递，T1，T2为一个类型

{

public:

B(T1 t1,T2 t2):A(t1),y(t2)

{};

T2 y;

};

2)普通类继承模板类

template<class T1>

class A

{

public:

A(T1 t1):x(t1)

{};

T1 x;

};

class B:public A<int> //这里A必须实例化

{

public:

B(int t1,int t2):A<int>(t1),y(t2) //

{};

int y;

};

3)模板类继承普通类

class A

{

public:

A(aa):a(aa)

{};

int a;

}

template<class T1>

class B:public A

{

public：

B(T1 t1,int aa):A(aa),b(t1)

{};

T1 b；

}

4.类模板作为函数的参数：

函数的形参可以使类模板对象或是类模板对象的引用，实参则是类模板实例化的模板类对象。

当一个函数拥有类模板参数的时候，这个函数也必须是函数模板。

如：

template<class T> //类模板

class A

{

public:

T x;

A(T xx):x(xx)

{};

}

template<class T>

void funcA(A &a) //类模板的函数

{

std::cout << a.x << std::endl;

}

5.

6.含有虚函数的模板类：

联系虚函数的概念，没什么特别

template<class T1>

class A

{

public:

T1 x;

A(T1 t1):x(t1)

{};

virtual func()

{};

};

7.模板抽象类：

含有纯虚函数的模板类