

XI`AN TECHNOLOGICAL UNIVERSITY

实验报告

实验课程名称 虚函数及其应用

专 业：计算机科学与技术

班 级： 16060104

姓 名： 杨 旭

学 号： 16060104117

实验学时： 2

指导教师： 刘白林

成 绩：

2017 年 6 月 18 日

西安工业大学实验报告

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 专业 | 计算机科学与技术 | 班级 | 计科四班 | 姓名 | 杨旭 | 学号 | 16060104117 |
| 实验课程 | C++ | 指导教师 | 刘白林 | 实验日期 |  | 同实验者 |  |
| 实验项目 | 虚函数及其应用 | | | | | | |
| 实验设备及器材 | 微型 | | | | | | |

一、实验目的

1．理解虚函数与运行时（动态）多态性之间的关系，掌握虚函数的定义及应用；

2．理解纯虚函数与抽象类的概念，掌握抽象类的定义及应用；

3．理解虚析构函数的概念及作用。

二、实验学时

课内实验：2课时 课外练习：2课时

三 本实验涉及的新知识

㈠ 虚函数与动态多态性

在C++中，如果将基类与派生类的同名成员函数定义为虚函数，就可以定义一个基类指针，当基类指针指向基类对象时访问基类的成员函数，当基类指针指向派生类对象时访问派生类的成员函数，实现在运行时根据基类指针所指向的对象动态调用成员函数，实现动态多态性。换句话说，虚函数与派生类相结合，使C++能支持运行时（动态）多态性，实现在基类中定义派生类所拥有的通用“接口”，而在派生类中定义具体的实现方法，即“一个接口，多种方法”。

㈡ 虚函数的定义

1．在基类中定义

在定义函数的前面加上“virtual ”。即：

virtual 返回类型 函数名(参数表)

{ …… }

2．在派生类中定义

函数的返回类型、函数名、参数的个数、参数类型及顺序必须与基类中的原型完全相同。

3．说明：

⑴ 在派生类中定义虚函数时，可用“virtual”也可不用“virtual”（最好都使用）。

⑵ 虚函数在派生类中重新定义时，其原型必须与基类中相同。

⑶ 必须用基类指针访问虚函数才能实现运行时（动态）多态性；当用普通成员函数的调用方法（即用圆点运算符）调用虚函数时，为静态调用；

⑷ 虚函数在自身类中必须声明为成员函数（不能为友元函数或静态成员函数），但在另一个类中可以声明为友元函数。

⑸ 虚函数可以公有继承多次，其虚函数的特性不变。

⑹ 构造函数不能定义为虚函数，但析构函数可以定义为虚函数。

⑺ 虚函数与重载函数的关系

① 普通函数重载是通过参数类型或参数的个数不同实现的；重载一个虚函数时，其函数原型（返回类型、参数个数、类型及顺序）完全相同。

② 当重载的虚函数只有返回类型不同时，系统将给出错误信息；如果定义的虚函数只有函数名相同，而参数个数或类型不同时，则为普通函数重载。

㈢ 纯虚函数与抽象类

1．纯虚函数定义

格式：

virtual 返回类型 func\_name(参数表)=0；

[ { …… } ]

2．抽象类

⑴ 抽象类的概念

当一个类中定义了一个或多个纯虚函数，则该类称为抽象类。

⑵ 说明：

① 抽象类中包含没有功能的纯虚函数，用来提供派生类的公共接口函数。因此，抽象类只能作为其它类的基类，不能定义其对象。

② 可以定义抽象类的指针或引用，用于指向派生类而实现多态性。

③ 如果在派生类中未重新定义纯虚函数，则派生类只是继承了基类的纯虚函数，这时，派生类仍是抽象类。

㈣ 虚析构函数

1．虚析构函数的定义

在析构函数名前加上“virtual”。即：

virtual ~ 类名（void）

{ …… }

2．说明

当类中需要显式定义析构函数，而在基类中定义了虚函数时，应将析构函数定义为虚析构函数，以实现动态调用析构函数。

四、实验内容

㈠ 验证及认知实验

按要求调试下列程序，并回答相关问题。

程序1（exp\_701.cpp）

#include <iostream.h>

class Base

{ int a,b;

public :

Base(int x,int y)

{a=x;b=y;}

void show()

{ cout <<" a="<<a<<" b="<<b; }

};

class Derived:public Base

{ int c;

public:

Derived(int x,int y,int z):Base(x,y)

{c=z;}

void show()

{ Base::show();

cout <<" c="<<c;

}

};

void main()

{Base mb(50,50),\*mp;

Derived md(10,20,30);

mp=&mb; mp->show();

cout<<endl;

mp=&md; mp->show();

cout<<endl;

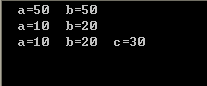
((Derived\*)mp)->show();

cout<<endl;

}

问题：

1. 编译运行程序的输出结果为：



⑵ 根据程序的输出结果可知：

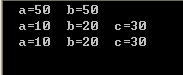
执行“mp=&mb; mp->show();”时，调用的是 Base 类的show() ；

执行“mp=&md; mp->show();”时，调用的是 Base 类的show() ；

执行“((Derived\*)mp)->show();”时，调用的是 Derived 类的show() ；

其中“((Derived\*)mp)”是将Base类“mp”指针强制转换为 Derived 类的指针。

⑶ 在基类“Base”中的成员函数“void show()”改为“virtual void show()”，再重新编译运行程序，输出结果为：



⑷ 当执行“mp=&md; mp->show();”时，调用的是 Base 类的show() ；函数“show()”称为 虚函数 ，实现了 运行时 多态性。

程序2（exp\_702.cpp）

#include <iostream.h>

class Base

{ public :

virtual void show()

{ cout<<"Base该类无计算"<<endl; }

};

class Derived1:public Base

{ int a ;

public:

Derived1(int x)

{a=x;}

void show()

{ cout <<"a="<<a<<endl; }

};

class Derived2:public Base

{ int b ;

public:

Derived2(int x)

{b=x;}

void show()

{ cout <<"b="<<b<<endl; }

};

void main()

{Base mb;

Base \*mp;

Derived1 md1(10);

Derived2 md2(20);

mp=&mb; mp->show();

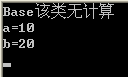
mp=&md1; mp->show();

mp=&md2;mp->show();

}

问题：

⑸ 编译运行程序的输出结果为：



⑹ 将“Base”类中的“virtual void show()”改为“virtual void show()=0”，重新编译程序会出现 错误 ，其中的“show() ”称为 纯 虚函数，该类称为 类，出错原因是 纯虚函数没有函数体 。

⑺ 去掉main()函数中的“Base mb;”及“mp=&mb; mp->show();”，重新编译运行程序的输出结果为：



㈡ 知识应用实验

1．分析下列程序，写出程序的输出结果，再上机运行程序验证其正确性，如果不正确，请认真分析出错原因。

程序3（exp\_703.cpp）

#include <iostream.h>

class parent

{protected :

char version;

public:

parent()

{version='A';}

virtual void print()

{cout <<"\n The parent. version "<<version;}

};

class derived1:public parent

{private:

int info;

public:

derived1(int number)

{info=number;

version='1';}

void print()

{cout<<"\n The derived 1 info:"<<info<<" version "<<version;}

};

class derived2:public parent

{private:

int info;

public:

derived2(int number)

{info=number; }

void print()

{cout<<"\n The derived 2 info:"<<info<<" version "<<version;}

};

void main()

{parent ob,\*op;

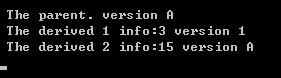
op=&ob;

op->print();

derived1 d1(3);

op=&d1;

op->print();



derived2 d2(15);

op=&d2;

op->print();

The parent. Version A

The derived 1 info 3 version 1

The derived 2 info 15 version A

cout <<endl;

}

程序4（exp\_704.cpp）

#include <iostream.h>

class base1

{ public:

virtual void fun()

{cout <<"----base1----"<<endl;}

};

class base2

{ public:

virtual void fun()

{cout <<"----base2----"<<endl;}

----base1----

----base2----

----derived----

----derived----

};

class derived:public base1,public base2

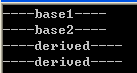
{ public :

void fun()

{cout <<"----derived----"<<endl;}

};

void main()



{ base1 ob1,\*p1;

base2 ob2,\*p2;

derived ob3;

p1=&ob1; p1->fun();

p2=&ob2; p2->fun();

p1=&ob3; p1->fun();

p2=&ob3; p2->fun();

}

程序5（exp\_705.cpp）

#include <iostream.h>

class figure

{ protected :

double x,y;

public :

figure(double a,double b)

{x=a;y=b;}

Triangle with height 10.0 and base 6.0

Has an area of 30.0

Square with dimension 10.0\*6.0has an area of60.0

Circle with radius 10.0has an area of100



virtual void show\_area( )

{ cout <<"No area computation defined";

cout <<" for this class.\n";

}

};

class triangle:public figure

{ public:

triangle(double a,double b):figure(a,b)

{ } ;

void show\_area()

{ cout<<"Triangle with height "<<x;

cout<<" and base "<<y<<" has an area of";

cout<<0.5\*x\*y<<endl;}

};

class square:public figure

{ public :

square(double a,double b):figure(a,b)

{ };

void show\_area()

{ cout <<"Square with dimension "<<x;

cout <<" \* "<<y<<" has an area of ";

cout <<x\*y<<endl;

}

};

class circle:public figure

{ public:

circle(double a):figure(a,a)

{ };

void show\_area()

{cout <<"Circle with radius "<<x;

cout <<" has an area of ";

cout <<x\*x\*3.141593<<endl;

}

};

void main()

{figure \*p;

triangle t(10.0,6.0);

square s(10.0,6.0);

circle c(10.0);

p=&t; p->show\_area();

p=&s; p->show\_area();

p=&c; p->show\_area();

}

2．完善、调试通过下列程序，并按所要求回答问题。

程序6（exp\_706.cpp）

//求求圆、圆的内切正方形及外切正方形的面积及周长

#include <iostream>

#include <math.h>

using namespace std;

class shape

{ protected :

double r;

public:

shape(double x)

{r=x;}

virtual double area()=0; //定¡§义°?纯ä?虚¨¦函¡¥数ºy

virtual double perimeter()=0; //定¡§义°?纯ä?虚¨¦函¡¥数ºy

};

class circle:public shape

{ public:

circle(double x):shape(x)

{ }

double area()

{return 3.14\*r\*r;}

double perimeter()

{return 2\*3.14\*r;}

};

class in\_square:public shape

{ public:

in\_square(double x):shape(x)

{ }

double area()

{return 2\*r\*r;}

double perimeter()

{return 4\*sqrt(r\*r+r\*r)\*r;}

};

class ex\_square:public shape

{ public:

ex\_square(double x):shape(x)

{ }

double area()

{return 4\*r\*r;}

double perimeter()

{return 8\*r;}

};

void main()

{ shape \*p;

circle ob1(5.0);

in\_square ob2(5.0);

ex\_square ob3(5);

p=&ob1;

cout <<"Circle :"<<" area="<<p->area(); //输º?出?圆2的Ì?面?积y

cout<<" perimeter="<<p->perimeter()<<endl;//输º?出?圆2的Ì?周¨¹长¡è

p=&ob2;

cout <<"Internal :"<<" area="<<p->area();//输º?出?内¨²切D正y方¤?形?的Ì?面?积y

cout<<" perimeter="<<p->perimeter()<<endl;//输º?出?内¨²切D正y方¤?形?的Ì?周¨¹长¡è

p=&ob3;

cout <<"External :"<<" area="<<p->area();//输º?出?外ªa切D正y方¤?形?的Ì?面?积y

cout<<" perimeter="<<p->perimeter()<<endl;//输º?出?外ªa切D正y方¤?形?的Ì?周¨¹长¡è

};

问题：

程序中（1）处应改为 0 ，（2）处应改为 r\*r+r\*r p=&ob1

（3）处应改为 p=&ob2 ，（4）处应改为 p=&ob3

㈢ 程序设计实验

用动态联编机制处理学生信息：

1．定义抽象类process：只提供一个纯虚函数print()；

2．定义person类，公有继承process类，定义print()成员函数显示相关信息（用前面已用过的类）；

3．定义student类，公有继承person类，定义print()成员函数显示相关信息（用前面已用过的类）；

4．在main()函数中，定义process指针变量调用person 对象及student对象。

说明：可将process类的定义及person类的定义放在同一个头文件中；测试程序存放在exp\_707.cpp中。

#include <iostream>

#include <string>

using namespace std;

class process

{

public:

virtual void print()=0;

};

class person:public process

{

public:

string name;

int age;

string sex;

person(string na="0",int a=0,string s="0")

{

name=na;

age=a;

sex=s;

}

virtual void print()

{

cout<<"姓?名?：êo"<<name<<endl;

cout<<"年¨º龄¢?：êo"<<age<<endl;

cout<<"性?别Àe：êo"<<sex<<endl;

}

};

class student:public person

{

private:

int num;

string Class;

public:

student (string na="0",int a=0,string s="0",int n=0,string C="0"):person(na,a,s)

{

num=n;

Class=C;

}

virtual void print()

{

cout<<"姓?名?：êo"<<name<<endl;

cout<<"年¨º龄¢?：êo"<<age<<endl;

cout<<"性?别Àe：êo"<<sex<<endl;

cout<<"学¡ì号?:"<<num<<endl;

cout<<"班ã¨¤级?：êo"<<Class<<endl;

}

};

void main()

{

process \*pn;

person ob1("陈?飞¤¨¦",19,"女?");

student ob2("程¨¬菲¤?",26,"女?",15,"经-管¨¹1班ã¨¤");

pn=&ob1;pn->print();

pn=&ob2;pn->print();

}