**2017C++复习资料**

**一、单项选择题**

1．对重载函数形参的描述中，错误的是 D 重载是重点

A. 参数的个数可能不同

B. 参数的类型可能不同

C. 参数的顺序可能不同

D. 参数的个数、类型、顺序都相同，只是函数的返回值类型不同

2.通常的拷贝构造函数的参数是 C

A． 某个对象名 B．某个对象成员名

C． 某个对象的引用 D．某个对象的指针名

3、关于构造函数特点的描述中，错误的是 A

A. 定义构造函数必须指出类型 （构造函数特点是无返回类型）

B. 构造函数的名字与该类的类名相同

C. 一个类中可定义0至多个构造函数 （不写构造函数，系统会自动生成一个空的构造函数）

D. 构造函数是一种成员函数

4、下面程序的运行结果为 C

#include <iostream.h>

int add(int a,int b);

void main()

{

extern int x,y;

cout<<add(x,y)<<endl;

}

int x=20,y=5;

int add(int a,int b)

{

int s=a+b;

return s;

}

A．20 B．5 C．25 D．编译会提示出错信息

5、关于常成员的描述中，错误的是 B

A. 常成员包含常数据成员和常成员函数两种

B. 常数据成员必须是公有的

C. 常数据成员要使用构造函数成员初始化列表进行初始化

D. 常对象只能调用它的常成员函数，不能调用其它的成员函数

6、关于友元函数的描述中，错误的是 B

A. 友元函数不是成员函数

B. 友元函数只能访问类中私有成员

C. 友元函数破坏隐藏性，尽量少用

D. 友元函数说明在类体内，使用关键字friend

7、对于动态分配内存空间描述正确的是 D

A．使用new运算符分配的内存空间的长度必需是常量

B．delete运算符可以释放动态的存储空间和静态的存储空间

C．由new分配的内存空间是不连续的

D．delete运算符只能释放由new分配的动态存储空间

8、能够释放对象所占资源的是 （ ） A

A．析构函数 B．数据成员

C．构造函数 D．静态成员函数

9、虚函数的定义是在基类中进行的，定义时需要冠以关键字 C

A．static B．frend

C．virtual D．public

10、在C++中串流类是在头文件strstrea.h中定义的，下列不属于串流类的是 ） C

A．strstream B．ostrstream

C．ofstream D．istrstream

11、下列特性中不是面向对象程序设计语言所特有的是（ D ）。

（A）数据封装 （B）继承（C）多态性 （D）模板

12、（ B ）是给变量取一个别名，它引入了变量的同意词。

（A）指针 （B）引用 （C）枚举 （D）结构

13、类成员的访问权限中，（ C ）只能被本类的成员函数和其友元函数访问。

（A）share （B）public （C）private （D）protected

14、关于构造函数，下列说法不正确的是（ D ）。

（A）构造函数名字和类名相同

（B）构造函数在创建对象时自动执行

（C）构造函数无任何函数返回类型

（D）构造函数有且只有一个

15、派生类可以直接访问其基类的（ D ）。

|  |  |
| --- | --- |
| （A）公有成员 | （B）保护成员 |
| （C）私有派生 | （D）公有和保护成员 |

16、下列关于运算符重载的描述中，错误的是（ B ）。

|  |
| --- |
| （A）运算符重载不可以改变操作数的个数 |
| （B）运算符重载不可以改变运算符的功能 |
| （C）运算符重载不可以改变结合方向 |
| （D）运算符重载不可以改变运算优先级 |

17、C++语言是从早期的C语言逐渐发展演变而来的.与C语言相比，它在求解问题方法上进行的最大改进是（ B ）

（A）面向过程　　（B）面向对象　　（C）安全性　　（D）复用性

18、对于类中定义的成员，其隐含访问权限为（ C ）。

A．public B．protected

C．private D．static

19、下面有关重载函数的说法中正确的是（ C ）

（A）重载函数必须具有不同的返回值类型；　　（B）重载函数形参个数必须不同；

（C）重载函数必须有不同的形参列表　　 （D）重载函数名可以不同；

20、有如下的对类“CSample”的说明，其中（ B ）是错误的。

　　class CSample {

　　A．int a=23;

　　B．CSample();

　　　　public:

　　C．CSample(int val);

　　D．~ CSample();

21、每个类（ C ）构造函数。

（A）只能有一个 （B）只可有公有的

（C）可以有多个 （D）只可有缺省的

22、在一个类中可以对一个操作符进行（ D ）重载。

（A）1 种 （B）2 种以下 （C）3 种以下 （D）多种

23、在公有继承的情况下，基类非私有成员在派生类中的访问权限（ B ）

（A） 受限制 （B）保持不变 （C）受保护 （D）不受保护

24、应在下列程序划线处填入的正确语句是（ C ）

#include <iostream.h>

class Base

{public:

void fun(){cout<<"Base::fun"<<ENDL;}

};

class Derived:public Base

{ void fun()

{\_\_\_\_\_\_\_\_\_\_\_\_\_//显示调用基类的函数fun()

cout<<"Derived::fun"<<ENDL;

}

};

（A）fun(); （B）Base.fun(); （C）Base::fun(); （D）Base->fun();

25、执行下面的程序将输出（ A ）

#include <iostream.h>

class BASE{

char c;

public:

BASE(char n):c(n){}

virtual ~BASE(){cout<<c;}

};

class DERIVED:public BASE{

char c;

public:

DERIVED(char n):BASE(n+1),c(n){}

~DERIVED(){cout<<c;}

};

int main()

{DERIVED a('X');

return 0;

}

（A）XY  （B）YX  （C）X  （D）Y

26下面描述中，表达错误的是（ B ）

（A）公有继承时基类中的public成员在派生类中仍是public的

（B）公有继承是基类中的private成员在派生类中仍是private的

（C）公有继承时基类中的protected成员在派生类中仍是protected的

（D）私有继承时基类中的public成员在派生类中是private的

27、定义析构函数时，应该注意（ C ）

（A）其名与类名完全相同 （B）返回类型是 void 类型

（C）无形参，也不可重载 （D）函数体中必须有 delete 语句

28、对于在类中定义的静态数据成员，下面正确的说法是（ C ）

A.该类的每个对象中都有一个静态数据成员

B.对象创建时产生

C.在类外进行初始化

D.既可以在类外初始化，也可以在类内初始化

29、C++中声明常量的关键字是（ A ）。

A. const B. extern C. public D. enum

30. 设置虚基类的目的是( B )

A.简化程序 B.消除二义性 C.提高运行效率 D.减少目标代码

31. 关键字（ C ） 说明对象或变量初始化后不会被修改。

A. static B. public C. const D. inline

32. 如果调用带有默认参数的函数时缺少一个参数，则（ B ）参数就作为这一参数。

A. 第一个 B. 最后一个 C. 中间一个 D. 以上都不是

33. 成员函数可声明为静态的，条件是它不访问 （ C ）类成员。

A. 静态 B. 常数 C. 非静态 D. 公共

34. 内联函数执行起来比标准函数（ B ）。

A. 更慢 B. 更快 C. 次数更多 D. 以上都不是

35. 默认参数的值由（ C ）提供。

A. 该函数 B. 调用程序 C. 上述二者 D. 以上都不是

36. 要让一个类中的所有对象具有共同的数据，请使用（ C ）。

A. 常数成员变量 B. 私有数据成员

C. 静态数据成员 D. 以上都是

37. 表达式 long(intVar) 也可表示为（ C ）。

A. intvar = long; B. intVar(long) C. (long)intVar D. 以上都不是

38. 如果表达式++i\*k中的“++”和“\*”都是重载的友元运算符，若采用运算符函数调用格式，则表达式还可以表示为（ B ）。

A. operator\*(i.operator++(),k) B. operator\*(operator++(i),k)

C. i.operator++().operator\*(k) D. k.operator\*(operator++(i))

39、下面对模板的声明，正确的是（ C ）。

A. template<T> B. template<class T1, T2>

C. template<class T1, class T2> D. template<class T1; class T2>

40、下列的各类函数中，（ C ）不是类的成员函数

A. 构造函数 B. 析构函数 C. 友元函数 D. 复制构造函数

41、C++是（ C ）。

A. 面向对象的程序设计语言

B. 面向过程的程序设计语言

C. 既支持面向对象的程序设计又支持面向过程的程序设计的混合型语言

D. 非结构化的程序设计语言

42、面向对象程序设计思想的主要特征中不包括（ D ）。

A. 封装性 B. 多态性 C. 继承性 D. 功能分解，逐步求精

43、若定义：string str; 当语句cin>>str; 执行时，从键盘输入：

Microsoft Visual Studio 6.0!

所得的结果是str=（ B ）。

A. Microsoft Visual Studio 6.0! B. Microsoft

C. Microsoft Visual D. Microsoft Visual Studio 6.0

44、考虑下面的函数原型声明：void testDefaulParam(int a,int b=7,char z='\*');

下面函数调用中，不合法的是（ C ）。

A. testDefaulParam(5); B. testDefaulParam(5,8);

C. testDefaulParam(5,'#'); D. testDefaulParam(0,0,'\*');

45、下列语句中，将函数int sum(int x, int y)正确重载的是（ C ）。

A. float sum(int x, int y); B. int sum(int a, int b);

C. float sum(float x, float y); D. double sum(int y, int x);

46、下列表示引用的方法中，（ A ）是正确的。

已知：int a=1000;

A. int &x=a; B. char &y; C. int &z=1000; D. float &t=&a;

47、在一个函数中，要求通过函数来实现一种不太复杂的功能，并且要求加快执行速度，选用（ A ）。

A. 内联函数 B. 重载函数 C. 递归调用 D. 嵌套调用

48、下列有关C++类的说法中，不正确的是（ D ）。

A. 类是一种用户自定义的数据类型

B. 只有类中的成员函数或类的友元函数才能存取类中的私有成员

C. 在类中，如果不做特别说明，所有成员的访问权限均为私有的

D. 在类中，如果不做特别说明，所有成员的访问权限均为公用的

49、已知X类，则当程序执行到语句：X array[3];时，调用了（ D ）次构造函数。

A. 0 B. 1 C. 2 D. 3

50、下面说法中，正确的是（ B ）

A. 一个类只能定义一个构造函数，但可以定义多个析构函数

B. 一个类只能定义一个析构函数，但可以定义多个构造函数

C. 构造函数与析构函数同名，只要名字前加了一个求反符号（~）

D. 构造函数可以指定返回类型，而析构函数不能指定任何返回类型，即使是void类型也不可以

51、已知：print( )函数是一个类的常成员函数，它无返回值，下列表示中，（ A ）是正确的。

A. void print( ) const; B. const void print( );

C. void const print( ); D. void print(const);

52、下面描述中，表达错误的是（ B ）

A. 公用继承时基类中的public成员在派生类中仍是public的

B. 公用继承时基类中的private成员在派生类中仍是private的

C. 公用继承时基类中的protected成员在派生类中仍是protected的

D. 私有继承时基类中的public成员在派生类中是private的

53、设置虚基类的目的是（ B 　）。

A. 简化程序 B. 消除二义性 C. 提高运行效率 D. 减少目标代码

54、下面（B ）的叙述不符合赋值兼容规则。

A. 派生类的对象可以赋值给基类的对象

B. 基类的对象可以赋值给派生类的对象

C. 派生类的对象可以初始化基类的对象

D. 派生类的对象的地址可以赋值给指向基类的指针

55、关于虚函数的描述中，（ C ）是正确的。

A. 虚函数是一个static类型的成员函数

B. 虚函数是一个非成员函数

C. 基类中说明了虚函数后，派生类中与其对应的函数可不必说明为虚函数

D. 派生类的虚函数与基类的虚函数具有不同的参数个数和类型

56、下面关于友元的描述中，错误的是（ D ）。

A. 友元函数可以访问该类的私有数据成员

B. 一个类的友元类中的成员函数都是这个类的友元函数

C. 友元可以提高程序的运行效率

D. 类与类之间的友元关系可以继承

57、下列关于静态数据成员的说法，不正确的是（ C ）。

A. 类中定义的公用静态数据成员，可以通过类的对象来访问

B. 类中定义的所有静态数据成员，都必须在类外初始化

C. 静态数据成员不是所有对象所共用的

D. 普通的成员函数可以直接访问类中的静态数据成员

58、运用运算符delete删除一个动态对象时（ c ）。

A．系统首先为该动态对象调用构造函数，再释放其占用的内存

B．系统首先释放该动态对象占用的内存，再为其调用构造函数

C．系统首先为该动态对象调用析构函数，再释放其占用的内存

D．系统首先释放动态对象占用的内存，再为其调用析构函数

59、可以在类外用p.a的形式访问派生类对象p的基类成员a，其中a是（ D ）。

A．私有继承的公用成员 B．公用继承的私有成员

C．公用继承的保护成员 D．公用继承的公用成员

60、不能说明为虚函数的是（ B ）。

A．析构函数 B．构造函数 C．类的成员函数 D．以上都不对

61、如果一个类至少有一个纯虚函数，那么就称该类为（ A ）。

A．抽象类 B．派生类 C．纯基类 D．以上都不对

62、关于C++与C语言关系的描述中，（ D ）是错误的。

A．C语言是C++语言的一个子集 B．C语言与C++语言是兼容的

C．C++语言对C语言进行了一些改进 D．C++语言和C语言都是面向对象的

63、已知：int m=10; 下列表示引用的方法中，（ A ）是正确的。

A．int &x=m; B．int &y=10; C．int &z; D．float &t=&m;

下列运算符中，（ C ）运算符在C++中不能被重载。

A．&& B．[] C．:: D．new

64、模板的使用实际上是将类模板实例化成一个（ C ）。

A．函数 B．对象 C．类 D．抽象类

65、假定MyClass为一个类，则该类的拷贝构造函数的声明语句为（ C ）。

A．MyClass(MyClass x) B．MyClass&(MyClass x)

C．MyClass(MyClass &x) D．MyClass(MyClass \*x)

66、下列关于类和对象的叙述中，错误的是( A )。

A)一个类只能有一个对象 B)对象是类的具体实例

C)类是对某一类对象的抽象 D)类和对象的关系是一种数据类型与变量的关系

67、对于语句 cout<<x<<endl;错误的是描述是( D )。

A)“cout”是一个输出流对象 B)“endl”的作用是输出回车换行

C)“x”是一个变量 D)“<<”称作提取运算符

68、有如下类声明，则类MyDERIVED中保护的数据成员和成员函数的个数是( B )。

class MyBASE

{ private: int k;

public: void set(int n){ k=n;}

int get( )const{ return k;} };

class MyDERIVED: protected MyBASE

{ protected: int j;

public: void set(int m, int n){ MyBASE::set(m); j=n;}

int get( ) const { return MyBASE::get( )+j; } };

　A)4 B)3 C)2 D)1

69、已知在一个类体中包含如下函数原型: Volume operator-(Volume)const;，下列关于这个函数的叙述中，错误的是( B )。

A)这是运算符-的重载运算符函数 　B)这个函数所重载的运算符是一个一元运算符

C)这是一个成员函数 　D)这个函数不改变类的任何数据成员的值

70、执行如下的程序段后，输出结果是( A )。

cout.fill('\*'); cout.width(6); cout.fill('#'); cout << 123 << endl;

A)###123 B)123### C)\*\*\*123 D)123\*\*\*

71、在下列函数原型中，可以作为类AA构造函数的是( D )。

A)void AA(int); B)int AA(); C)AA(int)const; D)AA(int);

72、以下( C )成员函数表示纯虚函数。

A)virtual int vf(int); B)void vf(int)=0;

C)virtual void vf()=0; D)virtual void vf(int){}

73、有如下类声明：class XA

{ private: int x;

public: XA(int n){ x=n;} };

class XB: public XA

{ private: int y;

public: XB(int a,int b); };

在构造函数XB的下列定义中，正确的是( B )。

A)XB::XB(int a,int b): x(a),y(b){ } B)XB::XB(int a,int b): XA(a),y(b){ }

C)XB::XB(int a,int b): x(a),XB(b){ } D)XB::XB(int a,int b): XA(a),XB(b){ }

74、在表达式 x+y\*z中， + 是作为成员函数重载的运算符，\* 是作为非成员函数重载的运算符。则 operator+ 有 个参数，operator\* 有 参数。( C )

A)2、2 B)2、1 C)1、2 D)1、1

75、应在下列程序划线处填入的正确语句是( C )。

class Base

　{ public:

void fun(){cout<<"Base::fun"<<endl;} };

　class Derived:public Base

{ void fun()

{ //显示调用基类的函数fun()

cout<<"Derived::fun"<<endl; } };

A)fun(); B)Base.fun(); C)Base::fun(); D) Base->fun();

76.下列类中（D）不是输入流类istream的派生类。

A. ifstream  B.istrstream C. iostream  D. ofstream

要实现动态联编，必须通过( A )调用虚函数。

A)对象指针 B)成员名限定 C)对象名 D)派生类名

77. 关于多继承二义性的描述，（D）是错误的。

A．派生类的多个基类中存在同名成员时，派生类对这个成员访问可能出现二义性

B．一个派生类是从具有共同的间接基类的两个基类派生来的，派生类对该公共基类的访问可能出现二义性

C．解决二义性最常用的方法是域运算符对成员进行限定

D．派生类和它的基类中出现同名函数时，将可能出现二义性

78. 关于输入/输出流，下面描述中正确的是( )

A. cin和cout是C++语言中用于输入输出的流类

B.使用cin和cout进行文件读写，必须首先打开文件

C. cin和cout可以对任何类型的数据或类对象进行输入输出

D. cin和cout以运算符重载的方式实现数据的输入输出

**二、填空题**

1、函数重载时，编译系统会根据\_\_形参的类型\_\_ 或\_\_形参的个数\_\_ 来区分。与返回类型无关

2、若有char a[80];则cin>>a表示从键盘上缓冲区中读入一个\_\_字符串\_\_ 到数组a中。

3、若希望用一个已有对象来构造另一个同类型的对象，可以使用\_\_拷贝构造函数\_\_来实现。

4、静态成员（static）属于\_\_类\_\_ ，而不属于\_\_任何一个对象\_\_ ，它被同一个类的所有对象共享。

5、类的继承方式有\_\_ public（或公有）\_\_ 继承、\_\_ private（或私有）\_\_ 继承和\_\_ protected（或保护）\_\_ 继承。

6、重载运算符的含义必须清楚，不能有\_\_二义性\_\_ 。

7、按照联编进行阶段的不同，联编方法可分为两种： \_\_动态联编\_\_ 和\_\_静态联编\_\_ 。

8、当基类中的某个成员函数被声明为虚函数后，此虚函数就可以在一个或多个派生类中被重新定义，在派生类中重新定义时，其函数原型，包括返回类型、\_\_函数名\_\_ 、\_\_参数个数\_\_ 、参数类型以及参数的顺序都必须与基类中的原型完全相同。

9、C++的I/O是以 字节流 的形式实现的，每个C++编译系统都带有一个面向对象的输入/输出软件包，这就是 I/O流类库 。

10、观看以下程序：

class point

{

public:

void show() {cout<<”point”<<endl;}

};

void main()

{

point p1;

point \*p;

p=&p1;

\_\_\_p1.show();\_\_\_\_\_\_\_\_\_\_\_\_//通过对象p1访问show函数

\_\_\_p->show();\_\_\_\_\_\_\_\_\_\_\_//通过指针p访问show函数

}

11、在C++类中可以包含\_\_\_\_\_public\_\_\_\_\_\_\_\_\_、\_\_\_\_\_private\_\_\_\_\_\_\_\_\_和\_\_\_\_\_protected\_\_\_\_\_\_\_\_\_三种具有不同访问控制权的成员。

1. 定义类时，在类头中将使用关键字\_\_\_\_class\_\_\_\_\_\_\_\_\_来表示类定义的开始。

13、如果一个类中至少有一个纯虚函数，则该类称为\_\_\_\_抽象类\_\_\_\_\_\_\_\_\_\_\_\_。

15、C++类的组成包括数据成员和\_\_\_成员函数\_\_\_\_\_\_\_\_\_\_\_，友元\_不是\_\_\_\_（是、不是）该类的成员函数。

16、友员函数是用关键字\_\_\_\_\_friend\_\_\_\_\_\_\_修饰的非成员函数。

17、若有：

int i;

int &j=i;

i=10;

j=i+1;

则i=\_\_\_11\_\_\_\_\_\_\_，j=\_\_\_\_\_11\_\_\_\_\_。

18、 new的功能是\_动态申请一段连续内在\_，delete的功能是\_\_释放由new申请的内存\_\_。

19、假定A是一个类名，则该类的拷贝构造函数的原型说明语句为： A(const A &); 。

20、后置自增运算符“++”重载为类的成员函数（设类名为A）的形式为：

A operator++ (int); 。

21、类和对象的关系可表述为：类是对象的 抽象 ，而对象则是类的 实例 。

22、静态成员函数没有隐含的 this指针 ，所以，在C++程序中，静态成员函数主要用来访问静态数据成员，而不访问非静态成员。

23、在图1中，A，B，C，D，E，F均是类，其中属于单继承的派生类有 E ，属于多继承的派生类有 D F ，类F的基类有 D E ，类A的派生 A、B、C、D、E类有 D F 。

A

B

C

D

E

F

图1 类的继承层次图

24、如果只想保留公共基类的一个复制，就必须使用关键字 virtual 把这个公共基类声明为虚基类。

25、从实现的角度来讲，多态性可以划分为两类： 静态多态性 和 动态多态性 。

26、如果一个类包含一个或多个纯虚函数，则该类称为 抽象类 。

27、若要把void fun()定义为类A的友元函数，则应在类A的定义中加入语句 friend void fun(); 。

28、列出C++中的两种代码复用方式： 继承 和 组合或模板 。

29、析构函数的作用是 在对象被系统释放之前做一些内存清理工作 。

**三、判断题**

1、C++语言支持封装性和继承性，不支持多态性。×

2、关系运算符可以用来比较两个字符的大小，也可用来比较两个字符串的大小。×

3、字符串"sdfgh"的长度是5。√ 占用空间是6，字符串有个结尾符号占空间。

4、引用和指针都可作函数参数。√

5、友元函数说明在类体内，它是一种成员函数。×说明（定义）在类外，声明在类里面

6、友元类必须被说明为公有成员。×

7、纯虚函数是在基类中说明的虚函数，它在该基类中没有定义具体的操作内容。√

8、析构函数不能被继承。√

9、运算符重载以后，其优先级和结合性都不能改变。√

10、在C++中，函数名实际上是一个指针，它指向函数的入口。√

11、友元函数用于允许一个函数访问不相关类的私有部分。√

12、构造函数可以被继承。×

13、动态绑定的多态性是通过虚函数实现的。√

14、在c++中，只能重载已有的运算符。√

15、重载函数必须有不同的参数列表。√

16、可以用delete释放不是用new运算符分配的内存。×

17、类成员的默认访问模式是private。√

18、在类Time中的析构函数可以声明为：void ~Time(int); ×

19、const对象必须初始化。√

1、C++语言支持封装性和继承性及多态性。( √ )

1.静态的成员函数没有隐含的this指针，所以它们只能访问静态的数据成员。（√）

2.通过类对象可以访问类中所有的成员。（×）

3.构造函数是可以被派生类继承的。（×）

4. 构造函数和析构函数都可以是虚函数。（×）

5.只有类中全部函数都为纯虚函数时，该类才被称为抽象类。（×）

6.对于同一个运算符来说，重载的运算符的优先级比未重载的运算符的优先级高。（×）

7.一个类的友元一定是另一个类的成员函数。（×）

**四、分析程序，回答所提出的问题**

1、 #include <iostream>

using namespace std;

class B;

class A

{

public:

A(int i){ a=i; }

friend int F(A &f1,B &f2);

private:

int a;

};

class B

{

public:

B(int i){ b=i; }

friend int F(A &f1,B &f2);

private:

int b;

};

int F(A &f1,B &f2) \\可以访问类A，类B里的任何成员

{return (f1.a+f2.b)\* (f1.a-f2.b);}

void main()

{

A n1(10); //a=10

B n2(8); //b=8

cout<<F(n1,n2)<<endl;

}

① 写出程序的运行结果。

② 该程序中共有几个对象，分别属于哪个类? 为什么在程序的开头处通过语句class B; 对类B进行声明？

③ 程序中的哪两条语句会分别调用A、B的构造函数？写出这两个构造函数。

④ 函数F是什么函数？它的作用是什么？

① 36

② n1 和 n2 两个对象，n1属于A类，n2属于B类。

因为B类的定义在A类后，而A类中要引用B类。

③ 语句A n1(10); 和语B n2(8);

A类的构造函数：A(int i) { a=i; }

B类的构造函数：B(int i) { b=i; }

④ 函数F是友元函数。

它的作用是：提供了不同类或对象的成员函数之间、类的成员函数与一般函数之间共享数据的机制。（关键答出共享数据、或答出通过友元可以访问到封装与类A、B中的数据）

**五、阅读程序，写出运行结果**

1、 #include <iostream.h>

class X

{

public:

X(int i)

{ x=i; cout<<”int: ”<<i<<’,’<<”called\n”; }

X(double i)

{ x=(int)i\*2; cout<<” double: ”<<i<<’,’<<”called\n”; }

~X()

{ cout<<” ~X(): ”<<x<<’,’<<”called\n”; }

private:

int x;

};

void main()

{

int a=8;

double b=11.0;

X x1(55), x2(a\*5);

X x3=X(2\*a);

X x4=X(b);

}

输出结果为：int:55,called // X1被创建

int:40,called //x2被创建

int:16,called //x3被创建

double:11,called //x4被创建 x值为22，显示的是i值，不改变

~x():22,called //析构函数的调用顺序与构造函数调用顺序相反

~x():16,called

~x():40,called

~x():50,called

2、

#include<iostream.h>

class Sample

{ int n;

public:

Sample(int i){n=i;}

friend int add(Sample &s1,Sample &s2);

};

int add(Sample &s1,Sample &s2)

{

return s1.n+s2.n;

}

void main()

{

Sample s1(10),s2(20);

cout<<add(s1,s2)<<endl;

}

执行结果是：30

3、

#include<iostream.h>

int add(int x,int y)

{

return x+y+1;

}

double add(double x,double y)

{

return x+y-1;

}

void main()

{

int a=2,b=4;

double c=2.6,d=7.4;

cout<<add(a,b)<<","<<add(c,d)<<endl;

}

执行结果是：7，9

4、

#include<iostream.h>

class A

{

public:

int n;

};

class B:virtual public A{};

class C:virtual public A{};

class D:public B,public C

{

int getn(){return B::n;}

};

void main()

{

D d;

d.B::n=10;

d.C::n=20;

cout<<d.B::n<<","<<d.C::n<<endl;

}

执行结果是：20,20

5、

#include <iostream>

using namespace std;

class myclass

{

int a,b;

static int s;

public:

myclass(int x,int y)

{a=x;b=y;s++;}

void print()

{cout<<s<<endl;}

};

int myclass::s=0;

void main()

{

myclass m1(1,2),m2(4,5),m3(6,7);

m1.print();

m2.print();

m3.print();

}

执行结果是：3 3 3

**6、**

**#include <iostream>**

**using namespace std;**

**class A**

**{friend double count(A&);**

**public:**

**A(double t, double r):total(t),rate(r){}**

**private:**

**double total;**

**double rate;**

**};**

**double count(A& a)**

**{**

**a.total+=a.rate\*a.total;**

**return a.total;**

**}**

**int main(void)**

**{**

**A x(80,0.5),y(100,0.2);**

**cout<<count(x)<<','<<count(y)<<'\n';**

**cout<<count(x)<<'\n';**

**return 0;**

**}**

**执行结果：120，120 180**

**7、**

#include<iostream>

using namespace std;

class Count{

private:

static int counter;

int obj\_id;

public:

Count();

static void display\_total();

void display();

~Count();

};

int Count::counter;

Count::Count()

{

counter++;

obj\_id = counter;

}

Count::~Count()

{

counter--;

cout<<"Object number "<<obj\_id<<" being destroyed\n";

}

void Count::display\_total()

{

cout <<"Number of objects created is = "<<counter<<endl;

}

void Count::display()

{

cout << "Object ID is "<<obj\_id<<endl;

}

int main(void)

{

Count a1;

Count::display\_total();

Count a2, a3,a4;

Count::display\_total();

a2.display();

a4.display();

return 0;

}

执行结果：Number of objects created is =1

Number of objects created is =4

Object ID is 2

Object ID is 4

Object number being 4 destroyed

Object number being 3 destroyed

Object number being 2 destroyed

Object number being 1 destroyed

8、

#include <iostream >

using namespace std;

class BASE

{

char c;

public:

BASE(char n):c(n){}

virtual ~BASE(){cout<<c;}

};

class DERIVED:public BASE

{

char c;

public:

DERIVED(char n):BASE(n+1),c(n){}

~DERIVED(){cout<<c;}

};

int main(void)

{ DERIVED('X');

return 0;

}

执行结果：XY

9、

#include <iostream>

using namespace std;

class A {

public:

A( ) { cout << "A"; }

};

class B {

public:

B( ) { cout <<"B"; }

};

class C: public A {

public:

C( ) { cout << "C"; }

private:

B b; //包含子对象

};

int main ( ) {

C obj;

return 0;

}

运行结果：ABC

10、#include <iostream>

using namespace std;

class A

{public:

A(){cout<<"A::A() called.\n";}

virtual ~A(){cout<<"A::~A() called.\n";}

};

class B: public A

{public:

B(int i)

{ cout<<"B::B() called.\n";

buf=new char[i];

}

virtual ~B()

{ delete []buf;

cout<<"B::~B() called.\n";

}

private:

char \*buf;

};

int main()

{ A \*a=new B(15);

delete a;

return 0;

}

运行结果：A::A() called.

B::B() called.

B::~B() called.

A::~A() called.

11.

#include <iostream>

using namespace std;

class A

{public:

void f(){cout<<"Call A's function f()"<<endl;}

};

class B

{public:

void f() {cout<<"Call B's function f()"<<endl;}

void g() {cout<<"Call B's function g()"<<endl;}

};

class C: public A, public B

{public:

void g(){cout<<"Call C's function g()"<<endl;}

};

int main()

{ C cc;

cc.B::f();

cc.B::g();

cc.g();

return 0;

}

运行结果：Call B's function f()

Call B's function g()

Call C's function g()

**六、程序填空**

#include <iostream>

using namespace std;

class A

{

\_\_\_ public: (或protected:)\_\_\_\_（1）\_ \_\_\_\_

char name[80];

public:

A( \_const char \*n\_\_\_（2）\_\_\_\_\_\_ ) {\_strcpy(name,n);\_\_\_（3）\_\_\_\_\_\_ }

};

class B\_:public A\_(或protected A)\_\_\_（4）\_\_\_\_\_\_\_

{

public:

B(const char\*n)\_:A(n)\_\_\_\_（5）\_\_\_\_\_\_\_{}

void PrintName( ) {cout<<”name:”<<name<<endl;};

};

void main( )

{

B b1(“Ling Li”);

b1.PrintName( ) ;

} // 执行结果： name: Ling Li

**2、**以下程序是定义一个计数器类counter，对其重载运算符“+”，请填空。

class counter

{ private: int n;

public:

counter() {n=0;}

counter(int i){n=i;}

\_ counter operator + (counter c)\_\_\_ //运算符重载函数

{ counter t; t.n=n+c.n; return t; }

void disp() {cout<<"n="<<n<<endl;}

};

void main()

{ counter c1(5),c2(10),c3;

c3=c1+c2;

c1.disp(); c2.disp(); c3.disp(); }

3、下列程序的输出结果为2，请将程序补充完整。

class Base

{ public:

\_\_virtual\_\_ void fun()

{ cout<<1; }

};

class Derived: public Base

{ public:

void fun( ) { cout<<2; }

};

int main( )

{ Base \*p= new Derived;

p->fun( );

delete p;

return 0; }

|  |
| --- |
| **七、编写程序（共2题，第1题10分，第2题14分，共24分）**  1、定义一个日期类Date，包含年、月、日三个数据成员，以及一个求第二天日期的成员函数和输出日期的成员函数。运行情况如下图： |

#include <iostream.h>

class Date

{

private:

int year,month,day;

public:

Date(int y, int m, int d)

{year=y;month=m;day=d;}

void nextday();

void display()

{cout<<year<<"/"<<month<<"/"<<day<<endl;}

};

void Date::nextday()

{

int totaldays[2][12]={{31,28,31,30,31,30,31,31,30,31,30,31},

{31,29,31,30,31,30,31,31,30,31,30,31}};

day++;

int leap=(year%400==0||year%4==0&&year%100!=0);

if(day>totaldays[leap][month-1])

{

day=1; month++;

if(month>12)

{

month=1;year++;

}

}

}

void main()

{

int d,m,y;

cout<<"请输入年、月、日：\n";

cin>>y>>m>>d;

Date d1(y,m,d);

cout<<"今天是：";

d1.display();

d1.nextday();

cout<<"明天是：";

d1.display();

}

2. 测试一个名为rectangle的矩形类，其属性为矩形的左下角与右上角两个点的坐标，能计算矩形的面积。

3.定义一boat与car两个类，二者都有weight属性，定义二者的一个友元函数totalweight()，计算二者的重量和。

4. 编写程序：定义抽象基类Shape，由它派生出五个派生类：Circle（圆形）、Square（正方形）、

Rectangle（ 长方形）、Trapezoid （梯形）和Triangle （三角形），用虚函数分别计算各种图形的面积，并求出它们的和。要求用基类指针数组。使它的每一个元素指向一个派生类的对象。

注：主函数中定义如下对象

Circle circle(12.6);

Square square(3.5);

Rectangle rectangle(4.5,8.4);

Trapezoid trapezoid(2.0,4.5,3.2);

Triangle triangle(4.5,8.4);

#include <iostream>

using namespace std;

class Shape

{public:

virtual double area() const =0;

};

class Circle:public Shape

{public:

Circle(double r):radius(r){}

virtual double area() const {return 3.14159\*radius\*radius;};

protected:

double radius;

};

class Square:public Shape

{public:

Square(double s):side(s){}

virtual double area() const {return side\*side;}

protected:

double side;

};

class Rectangle:public Shape

{public:

Rectangle(double w,double h):width(w),height(h){}

virtual double area() const {return width\*height;}

protected:

double width,height;

};

class Trapezoid:public Shape

{public:

Trapezoid(double t,double b,double h):top(t),bottom(t),height(h){}

virtual double area() const {return 0.5\*(top+bottom)\*height;}

protected:

double top,bottom,height;

};

class Triangle:public Shape

{public:

Triangle(double w,double h):width(w),height(h){}

virtual double area() const {return 0.5\*width\*height;}

protected:

double width,height;

};

int main()

{

Circle circle(12.6);

Square square(3.5);

Rectangle rectangle(4.5,8.4);

Trapezoid trapezoid(2.0,4.5,3.2);

Triangle triangle(4.5,8.4);

Shape \*pt[5]={&circle,&square,&rectangle,&trapezoid,&triangle};

double areas=0.0;

for(int i=0;i<5;i++)

{areas=areas+pt[i]->area();}

cout<<"totol of all areas="<<areas<<endl;

return 0;

}

5.下列Shape类是一个表示形状的抽象类，area( )为求图形面积的函数，total( )则是一个通用的用以求不同形状的图形面积总和的函数。

class Shape

{public:

virtual double area()=0;

};

double total(Shape \*s[ ], int n)

{ double sum=0.0;

for(int i=0; i<n; i++) sum+=s[i]->area( );

return sum;

}

要求：（1）从Shape类派生圆类(Circle)、正方形类（Square），圆类新增数据成员半径（radius），正方形类新增数据成员边长（a），圆类和正方形类都有构造函数，修改、显示数据成员值的函数，求面积函数。

（2）写出main()函数，计算半径为5.5的圆和边长为9.9的正方形的面积和（必须通过调用total函数计算）。

参考程序：

class Circle:public Shape

{public:

Circle(double r){radius=r;}

void set()

{ cout<<"Please input the value of the circle:"<<endl;

cin>>radius;

}

void show()

{ cout<<"the radius of the circle="<<radius<<endl; }

double area() {return 3.14159\*radius\*radius;}

private:

double radius;

};

class Square:public Shape

{public:

Square(double a){this->a=a;}

void set()

{ cout<<"Please input the value of a:"<<endl;

cin>>a;

}

void show()

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

double area(){return a\*a;}

private:

double a;

};

int main()

{

Circle C(5.5);

C.show();

Square S(9.9);

S.show();

Shape \*s[]={&C,&S};

cout<<"total="<<total(s,2)<<endl;

return 0;

}

6.有两个矩阵a和b，均为2行3列。求两个矩阵之和。重载运算符“+”，使之能用 于矩阵相加。如c=a+b。初值自拟。