**《面向对象编程技术》试卷**

**注：1）请将答案写在答题纸上，写在试卷上不算分。答题纸在试卷的最后页。**

**2）交卷时,试卷和答题纸一起交。**

**一、单选题 (每小题1.5分，共30分)**

1. C++中，以下有关构造函数的叙述不正确的是 \_\_\_\_\_\_ 。

A. 构造函数名必须和类名一致 B. 构造函数在定义对象时自动执行

C. 构造函数无任何函数类型 D. 在一个类中构造函数有且仅有一个

2．以下叙述不正确的是 \_\_\_\_\_\_ 。

A. 在类的定义中，通常是成员变量描述对象的属性；用成员函数描述对象的行为

B. 类的一个成员只能具有一种访问控制属性

C. 构造函数和析构函数是特殊的成员函数，因此不允许重载

D. 通过对象只能访问类的公有成员

3. 以下关于虚函数的叙述不正确的是 \_\_\_\_\_\_ 。

A. 虚函数属于成员函数 B. 虚函数不允许说明成静态的

C. 凡是虚函数必须用virtual说明 D. 虚函数可以被继承

4. cout是I0流库预定义的 \_\_\_\_\_\_ 。

  A．类  B. 对象  C. 包含文件  D. 常量

5．面向对象程序设计中的数据隐藏指的是 \_\_\_\_\_\_ 。

A．输入数据必须输入保密口令       B．数据经过加密处理

C. 对象内部数据结构上建有防火墙   D．对象内部数据结构的不可访问性

6．拷贝（复制）构造函数的作用是 \_\_\_\_\_\_ 。

A．进行数据类型的转换   　  B．用对象调用成员函数

C．用对象初始化对象   D．用一般类型的数据初始化对象

7. 下列不是描述类的成员函数的是 \_\_\_\_\_\_ 。

A.构造函数      B.析构函数

C.友元函数      D.拷贝构造函数

8. 如果类A被说明成类B的友元，则 \_\_\_\_\_\_ 。

A. 类A的成员即类B的成员        B. 类B的成员即类A的成员

C. 类A的成员函数不得访问类B的成员 D. 类B不一定是类A的友元

9. 对于任何一个类，析构函数最多有 \_\_\_\_\_\_ 个。

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

10. 下列特性中，C与C++共有的是 \_\_\_\_\_\_ 。

A.继承 B.封装 C.多态性 D.函数定义不能嵌套

11. 在公有继承的情况下，基类公有和保护成员在派生类中的访问权限 \_\_\_\_\_\_ 。

A. 受限制 B. 保持不变 C. 受保护 D. 不受保护

12. 通过 \_\_\_\_\_\_ 调用虚函数时，采用动态束定。

A. 对象指针 B. 对象名 C. 成员名限定 D. 派生类名

13. C++ 类体系中，不能被派生类继承的有 \_\_\_\_\_\_ 。

A. 成员转换函数 B. 构造函数 C. 虚函数 D. 静态成员函数

14. 假定 ab 为一个类，则执行 ab x； 语句时将自动调用该类的 \_\_\_\_\_\_ 。

A. 有参构造函数 B. 无参构造函数 C. 拷贝构造函数 D. 赋值构造函数

15. 静态成员函数不能说明为 \_\_\_\_\_\_ 。

A. 整型函数 B. 浮点函数 C. 虚函数 D. 字符型函数

16. 在 C++ 中，数据封装要解决的问题是 \_\_\_\_\_\_ 。

A. 数据规范化排列 B. 数据高速转换

C. 避免数据丢失 D. 保证数据完整性

17．关于友元函数的描述中，错误的是 \_\_\_\_\_\_\_\_ 。

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

18. 实现运行时的多态性用 \_\_\_\_\_\_ 。

A. 重载函数 B. 构造函数 C. 析构函数 D. 虚函数

19. 关于 new和new[] 运算符的下列描述中，\_\_\_\_\_\_ 是错误的。

A． 它可以用来动态创建对象和对象数组；

B．使用它创建的对象或对象数组可以使用运算符 delete和delete[] 删除；

C．使用它创建对象时要调用构造函数；

D．使用它创建对象数组时必须指定初始值；

20．下列关于虚函数的说明中，正确的是 \_\_\_\_\_\_ 。

A． 从虚基类继承的函数都是虚函数。

B． 虚函数不得是静态成员函数。

C． 只能通过指针和引用调用虚函数。

D． 抽象类中的成员函数都是虚函数。

**二、写出下列程序的运行结果，如您觉得有错,写明出错原因；如你觉得无输出，请明确写出。（每小题5分，共40分）**

1．

#include <iostream>

using namespace std;

class A{

protected:

int a;

public:

A(int p1) { a=p1; }

int inc1(void) { return ++a; } };

class B1:public A{

protected: int b1;

public: B1(int p1,int p2):A(p1) { b1=p2; }

int inc1(void) { return A::inc1(); }

void display(void) { cout<<"a="<<a<<" b1="<<b1<<"\n"; } };

class B2:public A{

private: int b2;

public: B2(int p1,int p3):A(p1) { b2=p3; }

int inc1(void) { A::inc1(); A::inc1(); return A::inc1(); }

void display(void) { cout<<"a="<<a<<" b2="<<b2<<"\n"; } };

class C:private B1,public B2{

private: int c;

public: C(int p11,int p12,int p21,int p22,int p):B1(p11,p12),B2(p21,p22)

{ c=p; }

int inc1(void) { return B2::inc1(); }

void display(void) { B1::display(); B2::display(); cout<<"c="<<c<<"\n";} };

int main(void){

C d(1,2,3,4,5); cout<<"\n"; d.display(); cout<<"\n";

d.inc1(); d.display(); return 0; }

2．

#include <iostream>

using namespace std;

template<class T>

class Tclass{

T x,y;

public:

Tclass (T a,T b):x(a) { y=b; }

Tclass (T a) { y=(T)0, x=a; }

void pr( ) { char c; c=(y>=(T)0)? '+':'-';

cout<<x<<c<< (y>(T)0 ? y : -y) <<"i"<<endl; } };

int main( ){

Tclass<double> a(10.5,-5.8); a.pr( );

Tclass<int> b(10); b.pr( ); return 0; }

3.

#include <fstream>

#include <iostream>

#include <vector>

using namespace std;

int main ()

{ vector<vector<int> > datas;

Ifstream ifs("test.dat");

int m, n;

ifs >> m >> n;

int i, j;

for (i = 0; i < m; i ++) {

vector<int> V;

for (j = 0; j < n; j ++ ) { int x; ifs>>x; V.push\_back(x); }

datas.push\_back (V); }

for (i = 0; i < m; i ++) {

for (j = 0 ;j < n; j ++) { cout << datas [i][j]<< " ";}

cout << endl; }

}

文件Test.dat内容如下 ：

3  4  1  2  3  4  5  6  7  8  9  10 11 12

4.

#include <iostream>

using namespace std;

class CAT

{ public: CAT(){ itsAge=new int; \*itsAge=5;}

~CAT(){ delete itsAge; itsAge=NULL;}

int GetAge(){ return \*itsAge; }

void SetAge(int age) { \*itsAge=age; }

protected: int \*itsAge; };

int main()

{ CAT a; cout<<"a's age:"<<a.GetAge()<<endl; a.SetAge(6);

CAT b(a); cout<<"a's age:"<<a.GetAge()<<endl;

cout<<"b's age:"<<b.GetAge()<<endl; a.SetAge(7);

cout<<"a's age:"<<a.GetAge()<<endl; cout<<"b's age:"<<b.GetAge()<<endl; }

5．

#include <iostream>

using namespace std;

#include <string>

class Base

{ private: char msg[30];

protected: int n;

public: Base(char s[],int m=0):n(m) { strcpy(msg,s); }

void output(void) { cout<<n<<endl<<msg<<endl; } };

class Derived1:public Base

{ private: int n;

public: Derived1 (int m=1): Base("Base",m-1) { n=m; }

void output(void) { cout<<n<<endl; Base::output();} };

class Derived2:public Derived1

{ private: int n;

public: Derived2(int m=2):Derived1(m-1) { n=m; }

void output(void) { cout<<n<<endl; Derived1::output();} };

int main()

{ Base B("Base Class",1); Derived2 D; B.output(); D.output();}

6．

#include <iostream>

using namespace std;

class Rr

{ public: Rr(int a){ data=a;}

operator int(){ return data;}

operator double(){ return double(data);}

private: int data; };

int main()

{ Rr r1(2),r2(4); int x=int(r1)+int(r2);

double y=double(r1)/double(r2); cout<<"x="<<x<<",y="<<y<<endl;

return 0; }

7.

#include <iostream>

using namespace std;

class sz { int intT1;

public: sz() { intT1=5;}

void Set1(int t) {intT1=t;}

int Get1() { return intT1; } };

class sz2

{ public: sz2() { mszT=new sz(); mszT2=new sz();}

sz \*f1() { mszT2=new sz(); mszT2->Set1(1); return mszT2; }

sz \*f2() const { return mszT; }

const sz \*f3() { return mszT; }

const sz \*f4() const {return mszT2;}

private:

sz \*mszT;

sz \*mszT2; };

int main()

{ sz t; sz2 f;

sz \*s1=f.f1(); cout<<s1->Get1()<<endl;

s1->Set1(10); cout<<s1->Get1()<<endl;

s1=f.f2(); cout<<s1->Get1()<<endl;

s1->Set1(20); cout<<s1->Get1()<<endl;

const sz \*s2=f.f3(); t=\*s2; cout<<t.Get1()<<endl;

s2=f.f1(); t=\*s2; cout<<t.Get1()<<endl;

sz \* const s3=f.f1(); t=\*s3; cout<<t.Get1()<<endl;

s3->Set1(30); cout<<s3->Get1()<<endl;

return 0; }

8.

#include <iostream>

using namespace std;

class A { public: virtual void print(int x) {cout <<"A:"<< x << '\t' ;} };

class B:public A { public: virtual void print(float x) {cout <<"B:"<< x << '\t';} };

class C:public A{ public: virtual void print(int x) {cout <<"C:"<< x <<endl;} };

void show(A &a) { a.print(3); }

void main( )

{ A a; B b; C c;

a.print(1); b.print(2); c.print(3);

show(a); show(b); show(c); }

**三、 程序填空题(每空2.5分，共25分)**

1． 下面程序定义了一个圆类，其数据成员为圆心坐标和半径，成员函数area()计算圆的面积，应用程序生成两个圆类实例，若是同心圆则计算圆环的面积。

#include <iostream>

using namespace std;

const double pi=3.14159;

class circle

{ int x,y; //圆心坐标

double r; //圆半径

public:

circle( \_\_\_\_\_\_\_\_ ) // 参数缺省值均为0

{ x=i; y=j; r=a; }

int getx() { return x; }

int gety() { return y; }

double area(); };

\_\_\_\_\_\_\_\_ area() { return pi\*r\*r;}

void main()

{ circle cmin(5,5,10), cmax(5,5,17);

if ( \_\_\_\_\_\_\_\_ ) //圆心坐标相同

cout<<"ring area="<< \_\_\_\_\_\_\_\_ <<endl; //输出圆环的面积

}

2．下列程序的输出结果为012。

#include <iostream>

using namespace std;

class Test{

public: \_\_\_\_\_\_\_\_\_\_\_\_\_

~Test() {}

void print(){ cout<< data; }

private:

int data; };

int main(){

Test t[3]= \_\_\_\_\_\_\_\_;

for(int i=0;i<3;i++) t[i].print();

return 0; }

3．下列程序的输出结果为：-1/2

#include <iostream>

using namespace std;

class Fraction{ //“分数”类

public: Fraction(double a,double b):num(a),den(b){}

//重载取负运算符“-”，返回一个分数对象，其分子是原来分子的相反数。

Fraction operator-(){ \_\_\_\_\_\_\_\_ ; return f;}

void print(){ cout<<num<<'/'<<den<<endl; }

private:

double num; //分子

double den; //分母

};

int main(){

Fraction f1(1,2);

(-f1).print(); return 0; }

4．函数模板Min的功能是返回具有n个元素的数组array中的最小值。

#include <iostream>

using namespace std;

template \_\_\_\_\_\_\_\_

T Min(T\* array,int n) {

T min=array[0];

for(int i=0;i<n;i++) \_\_\_\_\_\_\_\_ ;

return min;

}

int main(){

double arr[8]={1.2, 2.3, 31.21, 42, 51.5, 6.24, 74.12, 82};

cout<< \_\_\_\_\_\_\_\_ ;

return 0;

}

**答题纸**

一、单选题 (每小题1.5分，共30分)

1~5 6~10 11~15 16~20

二、看程序写运行结果。（每小题5分，共40分）

三、 程序填空题(每空3分，共30分)