浙江理工大学 2017 —2018 学年第二学期

《面向对象程序设计 A》期末试卷（B）卷

本人郑重承诺：本人已阅读并且透彻地理解《浙江理工大学考场规则》，愿意在考试中自觉遵守这些规定，保证按规定的程序和要求参加考试，如有违反，自愿按《浙江理工大学学生违纪处分规定》有关条款接受处理。

承诺人签名：_____ 学号：_____ 班级：_____

一．选择题（每小题 2 分，共 20 分）

1.下列的各类函数中，( )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

2.下列语句中，将函数int sum(int x, int y)正确重载的是( )。

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);

3.下面关于友元的描述中，错误的是( )。

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

4.设置虚基类的目的是( )。

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

5.下面( )的叙述不符合赋值兼容规则。

A. 基类的对象可以赋值给派生类的对象

B. 派生类的对象可以赋值给基类的对象

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

6.下面描述中，表达错误的是( )

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

7.关于虚函数的描述中，( )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

8. 表达式*class car : public vehicle*， 基类是(　　)

A. car　　B. vehicle

C. public　　D. class

9. 下列对模板的声明，正确的是(　　)

A. template<T>

B. template<class T1,T2>

C. template<typename T1, typename T2>

D. template<class T1; class T2>

10. 对于下面的类MyClass，在函数f()中将MyClass对象的数据成员n的值修改为50的语句应该是(　　)

```
class MyClass
{
public:
    MyClass(int x) {n = x;}
    void SetNum(int n) {this->n = n;}
private:
    int n;
}
int f( )
{
    MyClass *ptr = new MyClass(45);
    ____________________
}
```

A. MyClass(50);　　B. SetNum(50);

C. ptr->n = 50;　　D. ptr->SetNum(50) ;

二．阅读程序，写出程序的运行结果(共 29 分)

1．(4 分)

```
#include<iostream>
using namespace std;
void f(int i) { cout << "f(int)" <<i <<endl; }
void f(float i) { cout << "f(float)" <<i <<endl; }
template <class T>
```

```
void f(T i) { cout << "f(T)" <<i <<endl; }
void main() {
    f(1);
    f(1.1);
}
```

2.（8分）

```
# include <iostream>
using namespace std;
class MyClass{
public:
    MyClass();
    MyClass(int m);
    void print();
    ~MyClass();
private:
    int m;
    static int n;
};
int MyClass::n = 0;
MyClass::MyClass()
{
    cout <<"This is a constructor"<<endl;
    n += 10;
    this->m = 10;
}
MyClass::MyClass(int m)
{
    cout <<"This is a constructor"<<endl;
    this->m = m + 10;
}

void MyClass::print(){
    cout<<"The value of n is "<<n<<endl;
    cout<<"The value of m is "<<m<<endl;
```

```
}
MyClass::~MyClass(){
    cout<<"This is a destructor!"<<endl;
}
int main(){
    MyClass first;
    MyClass* p_sec = new MyClass(10);
    first.print();
    p_sec->print();
    delete p_sec;
    return 0;
}
```

3．(4 分)

```
#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;
}
```

4．(5 分)

```
#include <iostream>
using namespace std;
class Base{
public:
    int f(){return 11;}
    virtual int g(){return 12;}
};
class DerivedA:    public Base{
public:
    int f(){return 21;}
    virtual int g(){return 22;}
};
class DerivedB:    public Base{
public:
    virtual int g(){return 32;}
};
int main(){
       Base * pBase;
       Base objBase;
       DerivedA objDerivedA;
```

```
    DerivedB objDerivedB;
    pBase =& objBase;
    cout<< pBase -> f()<<endl;
    cout<< pBase -> g()<<endl;
    pBase =& objDerivedA;
    cout<< pBase -> f() + pBase -> g()<<endl;
    pBase =& objDerivedB;
    cout<< pBase -> f()<<endl;
    cout<< pBase -> g()<<endl;
    return 0;
}
```

5．(8 分)

```
#include <iostream>
using namespace std;
void MyFunc( void );

class CTest
{
public:
    CTest(){};
    ~CTest(){};
    const char *ShowReason() const { return "Exception in CTest class."; }
};

class CDtorDemo
{
public:
    CDtorDemo();
    ~CDtorDemo();
```

```
};

CDtorDemo::CDtorDemo(){
	cout << "Constructing CDtorDemo." << endl;
}
CDtorDemo::~CDtorDemo(){
	cout << "Destructing CDtorDemo." << endl;
}
void MyFunc()
{
	CDtorDemo D;
	cout<< "In MyFunc(). Throwing CTest exception." << endl;
	throw CTest();
}

int main()
{
	cout << "In main." << endl;
	try
	{
		cout << "In try block, calling MyFunc()." << endl;
		MyFunc();
	}
	catch( CTest E )
	{
		cout << "In catch handler." << endl;
		cout << "Caught CTest exception type: ";
		cout << E.ShowReason() << endl;
	}
	catch( char *str )
	{
		cout << "Caught some other exception: " << str << endl;
	}
	cout << "Back in main. Execution resumes here." << endl;
```

```
    return 0;
}
```

三．程序填空题(每空 2 分，共 32 分)

1．下列程序在构造函数和析构函数中申请和释放类的私有成员，在横线处填上适当的语句，完成该类的实现。

```
class    ClassA{
public:
        ClassA(int a);
        ~ ClassA ();
private:
      int *data;
};
ClassA:: ClassA (int a){
      _____(1)_____;
}
ClassA::~ ClassA (){
      _____(2)_____;
}
```

2．在横线处填上适当的语句，完成下列程序，使得本程序的运行结果为 62 30。

```
#include <iostream>
using namespace std;
class Rectangle {
public:
    double getLength() { return length; }
    double getWidth() { return width; }
    double area() { return length*width; }
    Rectangle(double w, double len) :    {
    _____(3)_____
  }
```

```
protected:
        _____(4)_____
private:
        _____(5)_____
};
class Cube :public Rectangle {
public:
        double getHigh() { return high; }
        double area()
        {
                return width*getLength() * 2 + width*high * 2 + getLength()*high * 2;
        }
        double volume() { _____(6)_____ }
        Cube(double w, double len, double h) :_____(7)_____{ high=h;}
private:
        double high;
};
void main() {
        Cube cub1(2,3,5);

        cout << cub1.area() << " ";
        cout << cub1.volume() << endl;
}
```

3. 在横线处填上适当的字句，完成下面复数类的定义。

```
#include<iostream>
```

```
class Complex{
    double real;
    double image;
public:
    Complex(double r=0.0, double i=0.0){
        (8) ;
        image=i;
    }
    Complex( (9) com1){
        real=com1.real;
        image=com1.image;
    }
    (10) Complex operator+(Complex c1, Complex c2);
    Complex operator-( (11) ) const{
        return Complex((real-c.real), (image-c.image));
    }
    friend void print(Complex c);
};
Complex operator+(Complex c1, Complex c2){
    return (12) ;
}
```

4．下列程序用虚函数 print 和运行的多态性，把从键盘输入的一个 int 型数值 n，按八进制和十六进制输出，完善程序。

```
#include<iostream.h>
class OCT{
protected:
    int n;
public:
    OCT(int x){ n=x; }
    (13) { cout<<n<<"的八进制为："<<oct<<n<<endl; }
};
class HEX:public OCT{
public:
    HEX(int x):(14)
```

```
    void print(){ cout<<n<<"的十六进制为："<<hex<<n<<endl; }
};
void main()
{   int n;
    cout<<"请输入一个十进制：";
    cin>>n;
    OCT oc(n);
    HEX he(n);
    ____(15)____;
    p=&oc;
    p->print();
    ____(16)____;
    p->print();
}
```

四、编程题（19 分）

1、(6 分)下面是一个类的测试程序，设计出能使用如下测试程序的类。

```
void main()
{
    Test x(300,200);
    x.print();
}
```

输出结果：300-200=100

2、(13分)编写程序：定义抽象基类Shape，由它派生出3个派生类：Circle(圆形)、Rectangle(长方形)、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：(三角形面积计算公式： S=sqrt[p(p-a)(p-b)(p-c)],p=(a+b+c)/2)

```
class Shape
{
public:
    virtual double area() const =0;
};
```

主函数中定义如下对象

```
Circle circle(12.6);
Rectangle rectangle(4.5,8.4);
Triangle triangle(4, 4,5.65);          //三个参数为三角形的三条边
```

输出结果是 :

```
total of all areas=544.599
```

浙江理工大学 2017 —2018 学年第二学期

《面向对象程序设计 A》期末试卷（B）卷

本人郑重承诺：本人已阅读并且透彻地理解《浙江理工大学考场规则》，愿意在考试中自觉遵守这些规定，保证按规定的程序和要求参加考试，如有违反，自愿按《浙江理工大学学生违纪处分规定》有关条款接受处理。

承诺人签名：_____ 学号：_____ 班级：_____

答题纸

一	二	三	四	总分

一．选择题（每小题 2 分，共 20 分）

1	2	3	4	5	6	7	8	9	10

二．程序阅读题(共 29 分)

1.（4 分）

2.（8 分）

3. （4 分）

4. （5 分）

5. (8 分)

三．程序填空题（每空 2 分，共 32 分）

(1)_____ (2)_____

(3)_____ (4)_____

(5)_____ (6)_____

(7)_____ (8)_____

(9)_____ (10)_____

(11)_____ (12)_____

(13)_____ (14)_____

(15)_____ (16)_____

四．程序设计题（19 分）

1．（6 分）

2．（13 分）