浙江大学 20 14 –20 15 学年 秋冬 学期

《面向对象程序设计》课程期末考试试卷

课程号： 211C0010 ，开课学院： 计算机科学与技术

考试试卷：A 卷√、B 卷（请在选定项上打√）

考试形式：闭√、开卷（请在选定项上打√），允许带 纸笔 入场

考试日期： 2015 年 01 月 20 日,考试时间： 120 分钟

诚信考试，沉着应考，杜绝违纪。

考生姓名：______学号：______所属院系：______

题序	一	二	三	四	五	六	七	八	总 分
得分									
评卷人									

1. Choices (10%)

Choose the only correct answer from the four choices.

1. Which is the correct way to define a pure virtual function? (C)
a. virtual void foo(); b. virtual void foo() {}
c. virtual void foo() = 0; d. void foo(){}

2. Which part of a class can be defined as virtual function? (D)
a. friend function b. constructor function
c. static member function d. destructor function

3. Which function will not be created by compiler if not defined explicitly? (D)
a. constructor function b. destructor function
c. copy constructor function d. inline function

4. Which of the following sentences is not correct (D)
a. Derived class object can be assigned to Base class object.
b. Base class object can be assigned to Derived class object.
c. Derived class object point can be assigned to Base class object point.
d. Base class object point can be assigned to Derived class object point.

5. What would be printed by the following statements? (C)

```
void main( )
{
        int i = 1;
        double x = 1.111;
        cout << i << " " << x << endl;
        {
                int x = 2;
                double i = 2.222;
                cout << i << " "<< x << endl;
        }
}
```

A. 1   1.111
   2   2

B. 1   1
   2.222   2

C. 1   1.111
   2.222   2

D. None of the above

2. Write the output of the code below (assuming all the header files are taken care of）(30%, 6% for 2nd question, 4% for the rest)

1）// 缺省构造函数

```
        class A {
                int i;
        public:
                A() : i(0) {}
                ~A() { cout << get(); }
                void set(int i) { this->i = i; }
                int get() { return i; }
        };

        int main()
        {
                A* p = new A[2];
                delete[] p;
                return 0;
        }
```

Answers:

00

2) // 构造函数与析构函数

```
class MyClass
{
public:
        MyClass(int id) { cout << "MyClass::Ctor\n"; }
        ~MyClass() { cout << "MyClass::Dtor\n"; }
private:
        int id;
};

class Base
{
public:
        Base(int _id):myclass(_id) { cout << "Base::Ctor\n"; }
        ~Base() { cout << "Base::Dtor\n"; }
        virtual void foo() { cout << "Base::foo\n"; }
private:
        MyClass myclass;
};

class Derived : public Base
{
public:
        Derived(int _id):Base(_id){ cout << "Derived::Ctor\n"; foo(); }
        ~Derived(){ cout << "Derived::Dtor\n"; foo(); }
        virtual void foo() { cout << "Derived::foo\n"; }
private:

};

int main()
{
        Base* p = new Derived(10);
        delete p;
        return 0;
}
```

Answers:

```
MyClass::Ctor
Base::Ctor
Derived::Ctor
Derived::foo
Base::Dtor
MyClass::Dtor
```

3) // 缺省构造函数

```
        class A {
                int i;
        public:
                A() : i(0) {}
                virtual void set(int i) { this->i = i; }
                virtual int get() { return i; }
        };

        class B : public A {
                int i;
```

```
public:
        B() : i(10) {}
        virtual void set(int i) { this->i = i; }
};

int main()
{
        B b;
        A* p = &b;
        p->set(30);
        cout << p->get();

        return 0;
}
```

Answers:
0

3) // 函数重载

```
class A {
public:
        void f(int a, int b=10) {cout << a+b; }
        void f(int a, int b=10) const {cout << a-b; }
};

int main()
{
        A a;
        A const * p = &a;
        p->f(25);
        return 0;
}
```

Answers:
15

4) // 虚函数

```
class A {
        int i;
public:
        virtual void set(int ii) { i = ii;}
        virtual int get() { return i; }
};

class B : public A {
        int i;
public:
        virtual void set(int ii) { i = ii;}
        virtual int get() { return i; }
    };

    int main()
    {
        B a;
        B b;
        a.set(10);
        b.set(20);
        A& p = a;
        p.set(30);
        p = b;
        p.set(40);
        cout << a.get();
        return 0;
    }
```

Answers:
40

5) // 继承

```
class A{
public:
    int f(){ return 1; }
    virtual int g(){ return 2; }
};

class B : public A{
public:
    int f(){ return 3; }
    virtual int g(){ return 4; }
};

class C : public A{
public:
    virtual int g(){ return 5; }
};

int main(){
A *pa;
    A a;
    B b;
    C c;

    pa = &a;
    cout << pa->f() << endl;
    cout << pa->g() << endl;
    pa = &b;
    cout << pa->f() + pa->g() << endl;
```

```
        pa = &c;
        cout << pa->f() << endl;
        cout << pa->g() << endl;

        return 0;
}
```

Answers:
1
2
5
1
5

6) // 继承与虚函数

```
class Base{
public:
        int Bar(char x)
        {
                return (int)(x);
        }
        virtual int Bar(int x)
        {
                return (2*x);
        }
};

class Derived : public Base{
public:
        int Bar(char x)
        {
                return (int)(-x);
        }
        virtual int Bar(int x)
        {
                return (x/2);
        }
};

int main()
{
        Derived obj;
        Base* pObj = &obj;

        cout<<pObj->Bar((char)100)<<endl;
        cout<<pObj->Bar(100)<<endl;

        return 0;
}
```

Answers:

```
100
50
```

3. Please correct the following programs（point out the errors and correct them. Please state the reasons if necessary (10%)

1) Please cross out all the lines that will not compile successfully **in the main function** of the following program

```
#include<iostream>
using namespace std;

class A{
public:
        A(){}
        virtual int output() = 0;
private:
        int i;
};

class B : public A{
private:
        int j;
};

class C{
public:
        int f(int a){ return x*a; }
protected:
        void setX(int a){ x = a; }
        int getX(){ return x; }
private:
        int x;
};

class D : public C{
private:
        int z;
};

int main(){
        A* pA = NULL;
        A objA;
        B objB;
        C objC;
        D objD;

        objC.setX(2);
        cout << objC.getX();

        objC.setX(1);
        objC.f(3);

        return 0;
}
```

Answers:

```
int main(){
        A* pA = NULL;
        ~~A objA;~~
        ~~B objB~~;
        C objC;
        D objD;

        ~~objC.setX(2);~~
        ~~cout << objC.getX();~~

        ~~objC.setX(1);~~
        objC.f(3);

        return 0;
}
```

2)

```
        class A {
        virtual void f() { cout << "lala"; }
        public:
                void f(int a, int b=10) {cout << a+b; }
                void f(int a, int b=10) const {cout << a-b; }
        };

        class B : public A {
        public:
                void f() { cout << "lili"; }
        };

        int main()
        {
                B a;
                A * p = &a;
                p->f();
                return 0;
        }
```

Answers:

A 中的 f()不能是 private 的。

3)
```
        class A {
                static int k;
        public:
                A():k(0) { cout << k; }
        };

        int main()
        {
                A a;
                return 0;
        }
```

答案：
1 去掉 static。或；
2 增加一行全局的：int A::k; 并且去掉初始化列表

4. Fill in the blanks（10%）

```
class Employee
{
public:
        Employee( const char * const, const char * const );
        ~Employee();
        const char  *getFirstName() const;
        const char *getLastName() const;
        static int getCount();
private:
        char *firstName;
        char *lastName;
        static int count;
};

int Employee::count = 0;
int Employee::getCount()
{
        return count;
}

Employee::Employee( const char * const first, const char * const last )
{
        firstName = new char[ strlen( first ) + 1 ];
        strcpy( firstName, first );

        lastName = new char[ strlen( last ) + 1 ];
        strcpy( lastName, last );

        count++;

        cout << "Employee constructor for " << firstName
```

```
    << ' ' << lastName << " called." << endl;
}
Employee::~Employee()
{
        cout << "~Employee() called for " << firstName
    << ' ' << lastName << endl;

        delete [] firstName;
        delete [] lastName;

        count--;
}
const char *Employee::getFirstName() const
{
        return firstName;
}
const char *Employee::getLastName() const
{
        return lastName;
}
int main()
{
        cout << "Number of employees: "
    << Employee::getCount() << endl;

        Employee *e1Ptr = new Employee( "Susan", "Baker" );
        Employee *e2Ptr = new Employee( "Robert", "Jones" );
        cout << "Number of employees: "
    << e1Ptr->getCount();

        cout << "\n\nEmployee 1: "
    << e1Ptr->getFirstName() << " " << e1Ptr->getLastName()
    << "\nEmployee 2: "
    << e2Ptr->getFirstName() << " " << e2Ptr->getLastName() << "\n\n";

        delete e1Ptr;
        e1Ptr = 0;
        delete e2Ptr;
        e2Ptr = 0;
        cout << "Number of employees: "
    << Employee::getCount() << endl;
        return 0;
}
```

```
Number of employees: 0
Employee constructor for Susan Baker called.
Employee constructor for Robert Jones called.
Number of employees: 2

Employee 1: Susan Baker
Employee 2: Robert Jones

~Employee() called for Susan Baker
~Employee() called for Robert Jones
Number of employees: 0
```

5. Program Design（40%）

Please implement your string class with functions listed below like the string class in **<string>** and write a main function to test them.

```
/* you may need use following built-in functions

char * strcpy ( char * destination, const char * source );
size_t strlen ( const char * str );
int strcmp ( const char * str1, const char * str2 );

*/

class MyString{
        friend ostream& operator<< (ostream&, MyString&);  (4%)
        friend istream& operator>> (istream&, MyString&);  (4%)
public:
        MyString(const char* str = NULL);  (4%)
        MyString(const MyString &other);  (4%)
        MyString& operator=(const MyString& other);  (4%)
        bool operator==(const MyString&);  (4%)
        char& operator[](unsigned int);  (4%)
        size_t size();(4%)
        ~MyString();  (4%)
private:
            char *m_data;
};

int main ()    (4%)
```

KEY

```
class MyString{
            friend ostream& operator<< (ostream&, MyString&);  (1)
            friend istream& operator>> (istream&, MyString&); (5)
public:
            MyString(const char* str = NULL); (2)
            MyString(const MyString &other); (3)
            MyString& operator=(const MyString& other); (4)
            bool operator==(const MyString&); (6)
            char& operator[](unsigned int); (7)
            ~MyString(); (8)
private:
            char *m_data;
};
```

Main () (9)

```
MyString::MyString(const char* str)      //1 分
{
        if (!str)      //1 分
                m_data = NULL;
        else {
                int slen = strlen(str);
                m_data = new char[slen + 1];    //以上两行 1 分
                strcpy_s(m_data, slen+1, str);   //1 分
        }
}

MyString::MyString(const MyString &other)  //1 分
{
        if (!other.m_data)   //1 分
                m_data = NULL;
        else
        {
                int slen = strlen(other.m_data);
                m_data = new char[slen + 1];   //以上两行 1 分
                strcpy_s(m_data, slen+1, other.m_data); //1 分
        }
}

MyString::~MyString()   //2 分
{
        delete[] m_data;  //2 分
}

MyString& MyString::operator=(const MyString& other)
{
        if (this  != &other)  //1 分
        {
                delete[] m_data;    //0.5 分
                if (!other.m_data)   m_data = 0;  //0.5 分
                else   //1 分
                {
                        int slen = strlen(other.m_data);
                        m_data = new char[slen + 1];
                        strcpy_s(m_data, slen+1, other.m_data);
                }
        }
        return *this;  //1 分
}

bool MyString::operator==(const MyString &s)  //1 分
{
        if (strlen(s.m_data) != strlen(m_data))
                return false;
        return strcmp(m_data, s.m_data) ? false : true;  //3 分
}

char& MyString::operator[](unsigned int e)  //1 分
{
        if (e >= 0 && e <= strlen(m_data))  //1 分
                return m_data[e];   //2 分
}

ostream& operator<<(ostream& os, MyString& str)  //2 分
{
        os << str.m_data;  //1 分
        return os;    //1 分
}

istream &operator>>(istream &is, MyString &s)  //1 分
{
        char temp[255];
        is >> setw(255) >> temp;
        s = temp;
        return is;
}

int main()
{
        MyString str1 = "Aha!";
        MyString str2 = "My friend";
        MyString str3;
        if (!(str1 == str2))
        {
                str3 = str1 + str2;
        }
        cout << str3 << "\n" << str3.size() << endl;
        return 0;
}
```