姓名_____

学号_____

院系_____

班级_____

课序号_____

任课教师_____

装

订

线

大 连 理 工 大 学

课程名称：面向对象方法与C++程序设计　试　卷：A　是否开卷 否

授课院（系）：软件学院　考试日期：2016 年 5 月 8 日　试卷共 14 页

	一	二	三	四	五	六	七	八	九	十	总分
标准分	30	20	20	20	10						100
得　分											

一、单项选择题（30 分，每题 2 分）

1. 在类定义的外部，可以被访问的成员有（C）

【A】所有类成员

【B】private 或 protected 的类成员

【C】public 的类成员

【D】public 或 private 的类成员

2. 类 clase C1 的说明如下，错误的语句是（B）

```
class  C1{
int a;              //(A)
void                //(B)
public:
C1(int val);        //(C)
~C1( );             //(D)
};
```

3. 下面对析构函数的正确描述是（A）

【A】系统可以提供默认的析构函数

【B】析构函数必须由用户定义

【C】析构函数可以有参数

【D】析构函数可以设置默认参数

4. 下面对友元的错误描述是（D）

【A】 关键字 friend 用于声明友元 ✓

【B】 一个类的成员函数可以是另一个类的友元函数

【C】 友元函数访问类对象的成员不受访问特性影响

【D】 友元函数通过 this 指针访问对象成员

5. 以下关于 this 指针的叙述中正确的是（D）

【A】任何与类相关的函数都有 this 指针

【B】类的成员函数都有 this 指针

【C】类的友员函数都有 this 指针

【D】类的非静态成员函数才有 this 指针

6. 下列函数中，不能重载运算符的函数是（B）

【A】成员函数　　【B】构造函数

【C】普通函数　　【D】友员函数

7. 关于C++中的重载机制，以下描述正确为（A）。
【A】可以对普通函数进行重载，也可以重载类的成员函数
【B】运算符只可以重载为成员函数 ✗
【C】类的构造函数与析构函数都可以被重载 ✗
【D】普通类和类模板都可以进行重载

8. 派生类采用（A）方式可以使基类中的保护数据成员成为自己的私有数据成员。
【A】私有继承　　【B】保护继承
【C】公有继承　　【D】私有、保护、公有均可

9. 在如下继承层次下，实例化派生类 Derived 的对象时，调用构造函数的顺序为（　）。

```
class Base{…};
class Base1: virtual Base{…};
class Base2: virtual Base{…};
class Derived : public Base2, public Base1{…};
```

【A】Base(), Base1(), Base(), Base2(), Drived()
【B】Base(), Base2(), Base(), Base1(), Drived()
【C】Base(), Base1(), Base2(), Drived()
【D】Base(), Base2(), Base1(), Drived()

10. 关于多重继承二义性的描述，错误的是（D）。
【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直接基类派生来的，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作用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

11. 下面描述中，正确的是（D）
【A】虚函数是没有实现的函数 ✗
【B】纯虚函数不必在派生类中实现
【C】抽象类是没有纯虚函数的类 ✗
【D】抽象类指针可以指向不同的派生类对象

12. 在C++中，容器是一种（D）。
【A】标准类　　【B】标准对象
【C】标准函数　　【D】标准类模板

13. 有如下函数模板定义：

```
template <class T>T fun(T x,T y){return x*x+y*y;}
```

在下列对 fun 的调用中，错误的是（C）。
【A】fun(2,8)　【B】fun(2.0,8.2)　【C】fun(2.3,8)　【D】fun('2','8')

14. 关于异常描述不正确的是（D）。
【A】c++的异常处理机制通过 3 个保留字 try、catch、throw 实现
【B】任何需要检测的语句必须放在 try 语句中，并 trow 语句抛出异常
【C】throw 语句抛出异常后，catch 语句利用数据类型匹配进行异常捕获
【D】一旦 catch 捕获异常后，不能将异常用 throw 语句再次抛出

15. istream 类中不能实现输入字符串的成员函数为（C）。
【A】get()　　【B】getline()
【C】<<　　【D】read()

二、写出下面程序的运行结果。（20 分，每题 5 分，共 4 题）

1. （5 分）

```
#include <iostream>
using namespace std;
class CExample {
private:
    int a;
public:
    CExample(int b){
        a=b;
        cout<<"构造函数"<<endl;
    }
    CExample(const CExample& C){
        a=C.a;
        cout<<"拷贝构造函数"<<endl;
    }
    void Show ()
    {
        cout<<a<<endl;
    }
    CExample add(){
        a++;
        return *this;
    }
};
int main()
{
    CExample A(100);
    CExample B=A;
    B.Show();
    B=A.add();
    B.Show();
    return 0;
}
```

构造函数 < 拷贝构造函数.
100.
拷贝构造函数.
101.

CExample B;
B=A;

2. （5 分）

```
#include<iostream>
using namespace std;
class Internet
{
public:
    Internet(char *name,char *address) {
        strcpy(Internet::name,name);
        strcpy(Internet::address,address);
        count++;
    }
    static void Internet::Sc() {
```

```
        cout<<count<<endl;
    }
    Internet &Rq();
public:
    char name[20];
    char address[20];
    static int count;
};
Internet& Internet::Rq(){
    return *this;
}
int Internet::count = 0;
void vist() {
    Internet a1("abc","www.cndev-lab.com");
    Internet a2("123","www.cndev-lab.com");
}
void fn(Internet s) {
    cout<<s.Rq().count<<endl;
}
void main() {
    cout<<Internet::count<<endl;
    vist();
    Internet::Sc();
    Internet b("sina","www.cndev-lab.com");
    Internet::Sc();
    fn(b);
    fn(b);
}
```

3. （5 分）

```
#include<iostream>
using namespace std;
class BASE {
public :
    void get (int l, int K ){
        a=l;x=K;
    }
     void print(){cout<<"a="<<a<<'\t'<<"x="<<x<<endl;
    }
     int a ;
protected:
    int x;
};
class A :public BASE{
public:
     void get (int l, int k){
          BASE obj3,obj4;
          obj3.get(50,60);
          obj3.print();
          BASE::a=l;x=k;
          a=BASE::a+obj3.a;
```

```
            obj4=*this;
            obj4.print();
}
      void print(){
            cout<<"a="<<a<<endl;
            BASE::print();
      }
private:
      int a;
};
void main(){
      BASE   Obj1;
      A Obj2;
      Obj1.get(10,20);
      Obj2.get(30,40);
      Obj1.print ();
      Obj2.print ();
}
```

4. （5 分）

```
#include<iostream>
using namespace std;
void XHandler(int test){
      try {
         if(test) throw test;
      }
      catch(int i){
            cout << "Caught exception #: " << i << endl;
      }
}
void main(void){
      cout << "Start: " << endl;
      XHandler(1);
      XHandler(2);
      XHandler(0);
      XHandler(3);
      cout << "End";
 }
```

三、程序填空（20 分，每题 5 分，共 4 题）

1. 每个横线处填写一条语句，使得程序输出如下结果。

```
constructor
3  abc
copystructor
3  abc
destructor
destructor
```

```
#include <iostream>
#include <string>
using namespace std;
class student
{
public:
    student(char *na);
    ~student();
    student(const student &);
    void print();
private:
    int num;   //记载字符个数
    char *name;    //字符起始地址
};
void student::print()
{
    cout<<num<<"  "<<name<<endl;
}
student::student(char *na)
{
   num=strlen(na);
   name = new char[num+1];
   strcpy(name,na);
   cout<<"constructor\n";
}
student::~student()
{
   delete[] name;
   cout<<"destructor\n";
}
student::student(const student &s)
{
//深度复制，使得每个对象有各自的堆空间
```

```
        num=s.num;
        name=new char[num+1];
        strcpy(name,s.name);
        cout<<"copystructor\n";
    }
    void main()
    {
       student s1("abc");
       s1.print();
    S2=student(const s1); student S2=S1;.
       s2.print();
    }
```

2. 程序填空，使得输出为：1243

```
#include <iostream>
using namespace std;
class A{
private:
      int X;
protected:
      int Y;
public:
      int Z;
      A(int a, int b, int c) {
           X=a; Y=b; Z=c;
      }
      int GetX() {
           return X;
      }
};
class B :public A{
private:
      int K;
public:
      B(int a, int b, int c, int d): X(a), Z(c). 不存在默认构造.
           K=d;                                  A(a,b,c).
      }
      void Show( ) {
           cout << GetX() << Y << K;
      }
};
void main( ){
      B b(1,2,3,4).
      b.Show();
      cout<<b.Z;
}
```

3.

```
#include <iostream>
using namespace std;
template <class T>
```

```
class Array1D{
public:
    Array1D(int size =0);
    //深复制
    Array1D(const Array1D<T>& v){
      size = v.size;
      element = new T[size];
      for(int i = 0;i <size;i++)
        element[i]=v.element[i];
    }
    ~Array1D(){                    delete [] element
      delete T                                ;
      element =0;                 }
    T& operator[](int i)const;
    int Size(){return size;}
private:
      int size    //数组的大小
    T* element;//一维数组的起始地址
};
      templete < class T >
Array1D<T>::Array1D(int sz ){
    if(sz <0) {
        cout << "argument error" << endl;
        exit(0);
    }
    size = sz;
    element =      new T [size ]
}
template <class T>
T& Array1D<T>::operator[](int i )const{
    if(i<0 || i>=size) i = 0;
    return element[i];
}
int main(){
     Array1D<int> array1(10);//实例化成整型一维数组
    int i;
    for(i =0; i< 10;i++)
        array1[i] = i;//调用[]函数
    Array1D<int> ____________________
    for(i =0; i< 10;i++)
        cout << array2[i] << ' ';   //输出 0 1 2 3 4 5 6 7 8 9
    cout << endl;
   return 0;
}
```

4.

```
#include<iostream>
#include<fstream>
using namespace std;
class Student
{
public:
    char id[10];
    char name[10];
    float score;
};
```

```
void main()
{
    //键盘输入写到 stud1.dat
    Student s[5];
    int i;
    for(i=0;i<=4;i++)
    {
      cin>>s[i].id>>s[i].name>>s[i].score;
    }
    ofstream outf(_____, ios::binary|ios::out) ;
    outf.write(_____ ___, sizeof(Student)*5 ) ; //写入五条记录
    outf.close();
    //读出每人成绩加 10
    ifstream inf( "stud1.dat",_____) ;
    inf._____ ( (char *) s,  sizeof(Student)*5 ) ; //读出五条记录
    for(i=0;i<=4;i++)
        s[i].score+=10;
    inf.close();
    //再次写入文件 stud1.dat
    outf._____( "stud1.dat",ios::binary|ios::out) ;
    outf.write( (char *) s,  sizeof(Student)*5 ) ;
    outf.close();
}
```

四、编程题（20 分，共 3 题）

1. （8 分）设计一个线段基类（Line），包括的属性为长度，当创建无参数对象时要求用户输入长度，当创建有参数对象时无需用户输入长度。其派生的矩形类在产生对象时要求输入两个相邻边（其中一条是从线段继承来的）的长度。完成线段类和矩形的设计与实现，使得下面程序运行结果如下：

```
输入线段的长度：
3
输入另一条相邻边线段长度：
4
矩形，相邻边分别为：A: 3, B: 4
```

```
#include < iostream >
#include < cmath >
using namespace std;
class Line{      //线段基类
    ┘
};
```

```
class Rectangle : public Line{     //矩形类




};
void main(){
    Rectangle *t = new Rectangle();
    t->printSize();
}
```

2. （6 分）完成下面类成员函数运算符重载+和-（取反）的声明与实现，使得运行结果为：

```
c1 + c2 = (-2, -6)
- c1 = (3, -4)
```

```
#include <iostream>
using namespace std ;
class Complex                                          //定义 Complex 类
{
public:
    Complex( ){dReal=0;dImag=0; }                      //默认构造函数
    Complex(double r, double i){ dReal=r;   dImag=i; } //重载构造函数
    Complex operator + ( const  Complex &c );          //重载运算符+
    Complex   operator -                               //重载运算符-
    void print( )const;
private:
    double dReal;                                      //实部
    double dImag;                                      //虚部
};
// 重载加号
Complex Complex :: operator + (const  Complex &c)
{
```

```
// 重载取反符号




//输出复数
void Complex::print( )const {
    cout << '(' << dReal << ", " << dImag << ')' << endl;
}
// 测试函数
int main( ){
    Complex c0,c1(-3,4),c2(1,-10);
    c0 = c1 + c2;
    cout << "c1 + c2 = ";
    c0.print( );
    c0 = -c1 ;
    cout << "- c1 = ";
    c0.print( );
    return 0;
}
```

3. （6分）设计一个函数模板 max，返回给定数组中数据的最大值，并用 int 型和 double 型数组在 main 函数中使用模板。

```
template < class T>
T max (T arr[], int n)     //arr 为数据数组，n 为数组中数据个数。
{









}

void main()
{









}
```

五、面向对象程序设计（10 分）

利用面向对象思想，描述一个班级（ClassRoom）学生运动的程序框架（类结构）。班级学生包括男生（Male）、女生（Female）。男生运动为俯卧撑，女生运动为仰卧起坐。

要求：

（1）设计出所有的类，只需要写出类的声明（只列出属性和方法）。

（2）必须有一个班级类，并且该类有一个 sports 方法，当运行班级类的 sports 方法时，输出所有学生运动的信息，对班级类的 sports 方法写出实现。