《面向对象程序设计》练习题

班级：　　姓名：　　学号：

一、　单选题

1、下面函数声明正确的是（　）。

A、 f (int x , int y);　　B、 void (x , y);

C、 void f(int x , y);　　D、 void f (int , int);

2、通常拷贝构造函数的参数是（　）。

A、某个对象名　　B、某个对象的成员名

C、某个对象的引用名　　D、某个对象的指针名

3、关于成员函数特征的下列描述中，错误的是（　）。

A、成员函数一定是内联函数　　B、成员函数可以重载

C、成员函数可以设置缺省参数　　D、成员函数可以是静态函数

4、将 string 类型字符串 s1 和 s2 进行连接的语句是（　）。

A、s3=s1+s2　　B、s3=s1.insert(1,s2)

C、s3=s1.replace(1,1,s2)　　D、s3=s1.substr(1,1)

5、在 C++中，函数签名不包括（　）。

A、函数的返回类型　　B、函数参数的个数

C、函数参数类型　　D、函数的名称

6、 C++中，关于下列设置函数参数默认值的描述中，（　）是正确的。

A、不允许设置函数参数的默认值

B、设置参数默认值时，应该全部参数都设置

C、设置参数默认值时，应按照从左向右的顺序设置

D、设置参数默认值时，应按照从右向左的顺序设置

7、下面说法正确的是（　）。

A、内联函数是在运行时将该函数的目标代码插入调用该函数的地方

B、内联函数是在编译时将该函数的目标代码插入调用该函数的地方

C、类的内联函数必须在类体内定义

D、类的内联函数必须在类体外通过加关键字 inline 定义

8、（　）不是构造函数的特征。

A、构造函数的函数名与类名相同　　B、构造函数可以重载；

C、构造函数可以设置缺省参数　　D、构造函数必须指定返回值类型

9、虚基类的主要作用是（　）。

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

10、关于 this 指针的说法错误的是（　）。

A、this 指针的值不可改变　　B、this 指针存放的是对象的地址信息

C、 this 是一个关键字　　D、静态成员函数拥有 this 指针

11、定义 int m = 3, &r = m;，则表达式++m, r--, m + r 的值为（　　）。

A. 4,4,8　　B. 4,3,7　　C. 6　　D. 7

12、定义函数如下：

```
int f(int a, int b = 1, int c = 2) {
    return a + b + c;
}
```

则下列计算结果**错误**的是（　）。

A. 表达式 f(1)的值为 4。　　B. 表达式 f(1, f(1))的值为 7。

B. 表达式 f(1, f(1), f(1))的值为 9。　　D. 表达式 f(f(1), f(1))的值为 11。

13、定义类 MyClass 如下：

```
#include<iostream>
using  namespace  std;
class MyClass {
  public:
    MyClass(int i) {}
};
```

则 main 函数中引发编译错误的程序行是（　　）。

```
int main() {
    MyClass m(1);           //A
    MyClass *p = &m;        //B
    MyClass ma[3];          //C
    ma[0] = MyClass(2);     //D
}
```

14、CInt 为自定义类，则下列分析**错误**的有（　　）。

```
CInt a, b(a);
CInt d = b;
CInt f();
void g(CInt c);
```

A. 定义对象 b 时需要调用该类的复制构造函数。

B. 定义对象 d 时需要调用该类的赋值运算符函数。

C. 调用函数 f 时需要调用该类的复制构造函数。

D. 在给函数 g 传递参数的过程中会调用该类的析构函数。

15、如下定义类 Var 描述变量，则下列分析**错误**的有（　　）。

```
#include<iostream>
using  namespace  std;
class Var {                 //变量类
    string name;            //变量的名称
public:
    Var(const char* str) :name(str) {}
    ~Var() {cout << name << "\t";}
};
Var global("gobal");
int main() {
    Var a("a");
    Var b("b");
}
```

A. 程序输出结果是 b　a　global。

B. 对象 a 最先构造。

C. 对象 b 最先析构。

C. 对象析构的时机与其作用域有关。

16、下列程序段中多处用到 const 关键字，main 函数中引发错误的有（　　）。

```
#include<iostream>
using  namespace  std;
class Object {
    int n;
public:
```

```
    Object(int m) :n(m) {}
    void inc() {++n;}
    void set(int m) {n = m;}
    void print() const {cout << n << endl;}
    void copy(const Object& other) {n = other.n;}
};
int main() {
    Object a(1);
    const Object b(2);
    a.inc();                //A
    b.set(1);               //B
    b.print();              //C
    a.copy(b);              //D
}
```

17、MyClass 为自定义类，则下列分析中错误的有（　　）。

```
int main() {
    MyClass *a = new MyClass;               //1
    MyClass *b = new MyClass(a);            //2
    delete a;                               //3
    delete b;                               //4
}
```

A. 第 1 行程序需要调用类 MyClass 的构造函数。

B. 第 2 行程序需要调用类 MyClass 的复制构造函数。

C. 第 3 行程序需要调用类 MyClass 的析构函数。

D. 对象 a 在对象 b 之后析构。

18、a 和 b 是自定义的有理数类 Rational 的两个对象，该类对某些算术运算符进行了重载。下列说法中**错误**的是（　　）。

A. 表达式 a+b 中的运算符“+”一定重载为成员函数形式。

B. 表达式 a+b 可以用 operator+(a,b)形式调用运算符函数。

C. 表达式 3+a 中的运算符“+”必须重载为类的友元函数。

D. 表达式 cout<<a 中的运算符“<<”必须重载为类的友元函数。

19、对自定义类型 T 重载赋值运算符“=”时，可以（　　）。

A. 把参与赋值的两个操作数设置为运算符函数的参数

B. 把该运算符函数声明为 const 成员函数

C. 把该运算符函数声明为类的友元函数

E. 在类定义体外定义该运算符函数

20、定义 STL 容器类 vector 的对象 v 如下，则下列操作**有误**的是（　　）。

```
vector<int> v(10);
```

A. v.push_front(1)

B. v[2] = 2

C. v.back() = 10

D. cout << v.size()

21、假定 MyClass 为一个类，则执行 MyClass a, b(2), *p;语句时，自动调用该类构造函数(　　)次。

A. 2　　B. 3　　C. 4　　D. 5

22、以下选项中没有 this 指针的函数是(　　)。

A. 内联成员函数　　B. 构造函数

C. 静态成员函数　　D. 析构函数

23、下列不是类的成员函数的是(　　)

A. 构造函数　　B. 析构函数　　C. 友元函数　　D. 复制构造函数

24、对下列语句正确的是(　　)。

```
const int* x;// 1
int* const x;// 2
```

A. 语句 1 的含义是指针变量 x 不能更改　　B. 语句 2 的含义是指针变量 x 所指向的值不能更改

C. 语句 2 的含义是指针变量 x 不能更改　　D. 语句 1 和 2 是相同含义的不同定义方式

25、假定 Example 是一个类，“Example* function( )const;”是该类中一个成员函数的原型，若该函数返回 this 值，当用 x.function ()调用该成员函数后，x 的值(　　)。

A. 已经被改变　　B. 可能被改变

C. 不变　　D. 受到函数调用的影响

26、有如下面程序, 其运行结果是(　　) 。

```
#include<iostream>
using  namespace  std;
int  main(){
     int x; int &y=x;
      x=5; y=10;
      cout<<x<< ‘,’ <<y<<endl;
     return  0;
  }
```

A. 5,5　　B. 5,10

C. 10,5　　D. 10,10

27、有如下类声明：

```
class  MyClass{ int  x; };
```

在执行下列程序段后:

```
MyClass  a[2], b, *p=&b;
```

其构造函数被调用了(　　)次.

A. 1　　B. 2　　C. 3　　D. 4

28、下面哪个是析构函数的特性？ (　　)

A. 一个类仅只有一个析构函数　　B. 析构函数可以重载

C. 析构函数必须在类声明中定义。　　D. 析构函数可以有一个或多个参数

29、关于类的静态（static）成员的描述中，错误的是(　　)。

A. 静态成员分静态数据成员和静态成员函数两种

B. 静态数据成员初始化不用构造函数

C. 静态成员函数中不能直接访问非静态成员

D. 静态数据成员初始化必须在类声明中

30、关于类的成员函数，下面描述错误的是 (　　)。

A. 成员函数必须是内联（inline）函数。

B. 成员函数可以重载。

C. 除了构造函数和析构函数，成员函数必须指定返回类型。

D. 成员函数的参数可以有默认参数值。

31、下面哪个不是函数签名?(　　)

A. 形式参数的个数　　B. 形式参数的数据类型

C. 形式参数的顺序　　D. 函数返回值类型

32、构造函数的初始化段不能用于初始化(　　) 。

A. const 数据成员　　B. 引用数据成员

C. static 数据成员　　D. 对象成员

33、如有下面的声明：

```
string s1; char s2[20];
```

则不正确的语句是（　　）。

A. s1=s1+s2;　　B. s2=s1;

C. s1.append(s2);　　D. s1=s2;

34、下面哪个是 const 成员函数的声明?(　　)

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

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

35、基类的那些成员可以通过派生类对象直接访问?(　　)

A. 公有继承方式基类的公有成员

B. 保护继承方式基类的公有成员

C. 公有继承方式基类的保护成员

D. 保护继承方式基类的保护成员

36、抽象类应该含有(　　)。

A. 至多一个纯虚函数　　B. 至少一个纯虚函数

C. 至多一个虚函数　　D. 至少一个虚函数

37、关于虚函数的下述描述中，正确的是(　　)。

A. 虚函数不是成员函数　　B. 虚函数是非静态成员函数

C. 虚函数不能继承　　D. 派生类的虚函数与基类的虚函数在参数上不能相同

38、下列运算符中，不能重载的运算符是(　　)。

A. ->　　B. [ ]

C. &&　　D. ::

39、下列描述中，正确的是(　　)。

A. C++语言只支持单一继承，不支持多重继承。

B. 派生类不能作为基类派生出新类。

C. 构造函数和析构函数都不能被继承。

D. 派生类的析构函数中不会调用直接基类的析构函数。

40、已知类模板定义如下：

```
template<class ST>
class  A{
      public:
         A(int i)
            {//...}
       //...
      };
```

下列关于模板类对象的定义，正确的是(　　)。

A. A<ST> a(5);　　B. A<class ST> a(5);

C. A<int> a(5);　　D. A<class int> a(5);

41、下列程序段中有（　　）处错误。

```
#include <iostream>
int main(); {
    int x;
    std::cin >> x >> std::endl;
    std::cout << "x = 3" << "\n";
    }
```

A. 2　　B. 3

C. 4　　D. 5

42、对下列程序段的描述中正确的是（　　）。

```
int f(int x);
int main() {double n = f(3);}
int f(int x) {return x;}
```

A．函数定义正确，函数调用错误。　　B．函数调用正确，函数定义错误。

B．函数定义和函数调用都正确。　　D．函数原型错误，函数调用正确。

43、下列函数参数的默认值定义错误的是(　　)。

A．int f(int x, int y, int z = 0);

B．int f(int x = 0, int y = 0, int z);

C．int f(int x, int y = 0, int z = 0);

C．int f(int x = 0, int y = 0, int z = 0);

44、关于消息，下列说法中不正确的是(　　)。

A．发送消息的对象请求服务，接受消息的对象提供服务

B．消息的发送者必须了解消息的接收者如何相应消息

C．在C++中，消息的发送具体体现为对接收消息的对象的某个函数的调用

D．每个对象只能接收某些特定格式的消息

45、对类的构造函数和析构函数描述正确的是(　　)。

A．构造函数可以重载，析构函数不能重载

B．构造函数不能重载，析构函数可以重载

C．构造函数可以重载，析构函数也可以重载

D．构造函数不能重载，析构函数也不能重载

46、在C++语言中，作用域：：运算符的功能是(　　)。

A．标识作用域的级别的；　　B．指出作用域的范围的；

B．给定作用域的大小的；　　D．标识某个成员是属于哪个类的。

47、假定MyClass为一个类，则执行MyClass a, b(2), *p;语句时，自动调用该类构造函数(　　)次。

A．2　　B．3　　C．4　　D．5

48、以下选项中没有this指针的函数是(　　)。

A．内联成员函数　　B．构造函数　　C．静态成员函数　　D．析构函数

49、下列不是类的成员函数的是(　　)

A．构造函数　　B．析构函数　　C．友元函数　　D．复制构造函数

50、对下列语句正确的是(　　)。

```
const int* x;    // 1
int* const x;    // 2
```

A．语句1的含义是指针变量x不能更改

B．语句2的含义是指针变量x所指向的值不能更改

C．语句2的含义是指针变量x不能更改

C．语句1和2是相同含义的不同定义方式

51、假定Example是一个类，“Example* function( )const;”是该类中一个成员函数的原型，若该函数返回this值，当用x.function ()调用该成员函数后，x的值(　　)。

A．已经被改变　　B．可能被改变

B．不变　　D．受到函数调用的影响

52、对静态成员的不正确描述为(　　)。

A．静态成员不属于对象，是类的共享成员

B．静态数据成员要在类外定义和初始化

C．调用静态成员函数时要通过类或对象激活，所以静态成员函数拥有this指针

D．只有静态成员函数可以操作静态数据成员

53、关于 delete 运算符的下列描述中，(　　)是错的。

A. 它必须用于 new 返回的指针

B. 它也适用于空指针

C. 对一个指针可以使用多次该运算符

D. 指针名前只用一对方括号符，不管所删除数组的维数

54、有函数 f 定义如下：

```
int f (int a  = 1,  int b = 2,  int c=3){
      return a + b + c;
    }
```

下列说法中错误的是(　　)。

A.函数调用 f()的结果为 6。

B. 函数调用 f(f(f()))的结果为 1 6。

C. 函数调用 f(f(), f())的结果为 1 5

D. 函数调用 f(f(), f(f()))的结果为 2 1。

55、下列程序的输出结果是(　　)。

```
#include<iostream>
using  namespace  std;
void  f(int&  i,  int&  j)  {
    int  t(i);
    i = j,  j = t;
    cout  <<  i  <<  "\t"  <<  j  <<  "\t";
  }
int  main()  {
    int  a=  5,  b=  8;
    f(b,  a);
    cout  <<  a  <<  "\t "  <<  b  <<  endl;
    return 0;
   }
```

A. 8 5　5　8　　B.　5　8　8　5

C. 5 8　5　8　　D.　8　5　8　5

56、如下是某同学定义的类 A，则其中有(　　)处错误。

```
class A{
    int  a=  1;
    void A();
    ~A (void);
    };
```

A.2　　B. 3　　C. 4　　D. 5

57、针对下列类型定义，可以出现在 main 函数中的语句有(　　)。

```
class Date  {
    private:
    int year,  month,  day;
    };
```

A. Date d;

B. Date: :year = 1;

C. Date d = {2009, 1, 1};

D. cout<<month;

58、有程序如下，下列说法中错误的是(　　)。

```
#include<iostream>
using  namespace  std;
class RMB  {
     int  yuan,  jiao,  fen;
  public:
      RMB (int  y= 0,  int  j =0, int  f=0){
            yuan  =  y;  jiao  =  j;  fen  =  f;
         }
       ～RMB()  {cout << yuan << “:” <<  jiao << “:” << fen ;}
  };
int main()  {
      RMB a[2],  b(10,  20,  30),  c(b);
      return 0;
  }
```

A. 20:10:30 不可能是程序的输出结果。

B. 析构对象 c 时输出结果是 0:0:0。

c. 0:0:0 会在输出结果中出现两遍。

D. 对象 a[0]最后析构。

59、类 A 是自定义类型，下列说法中错误的是(　　)。

```
A* a= new A;
A*b= new A(a);
delete  a;
delete  b;
```

A. 构造堆对象 a 时会调用类 A 的默认构造函数。

B. 构造堆对象 b 时会调用类 A 的复制构造函数。

C. 堆对象 a 先构造，所以它后析构。

D. 释放指针 b 所指内存时，会调用类 A 的析构函数。

60、a、b、C 是类 T 的对象，为计算表达式 c=a+b，必须(　　)。

A. 以成员函数形式重载加法运算符+。

B. 以友元函数形式重载赋值运算符=。

c. 以友元函数形式重载加法运算符+。

D. 以 T 作为加法运算符函数的返回类型。

61、函数模板 u2v 的功能是把 u 类型的数据转换为 v 类型的数据，它的定义如下则下列说法中错误的是(　　)。

```
template <class V,  class U>
    V u2v(U u)  {
        return V(u);
    }
```

A. 表达式 u2v (3.14)的值为 3。

B. 表达式 u2v<int> (1.2)的值为 1。

C. 表达式 u2v<int>(‘a’)的类型为 char。

D. 必须显式实例化模板参数 v。

62、s 是 STL 数据类型 string 的对象，下列表达式错误的是(　　)。

A. S+=’S’　　B.　s=”Hello”

C.　s-‘a’　　D.　s>”abc”

63、下列程序中，访问出错的是(　　)。

```
#include<iostream>
using  namespace  std;
class A{
      private:     int x;
      protected:    int y()  {  x = 1 ;}        //A
      public:     int  Z;
  };
class  B  :   public A  {
      public:
        void u()  {cout << x;}                  //B
 };
int  main()  {
         B b;     b. z  =  1;                    //C
         b. u();                                //D
         return 0;
  }
```

64、有类型定义如下，则下列说法中错误的是(　　)。

```
class Object  {};
class A  { Object oa;};
class  B  :  public A  { Object ob;};
```

A. 构造类 B 的对象时会首先调用类 Obj ect 的构造函数。

B. 对语句 B b; 的执行会先后引发 6 次函数调用。

C. 析构类 B 的对象时，类 B 的析构函数在类 A 的析构函数之前调用。

D. 析构类 A 的对象时会最后调月类 Obj ect 的析构函数。

65、下列(　　)行的输出结果是 B::v 。

```
#include<iostream>
using  namespace  std;
class A{
    public:
      void  f ( )   { cout<< "A: : f"<<endl; }
      virtual void v()  {cout<<"A: :v"<< endl; }
     };
 class B : public A {
    public :
       void f ()  {cout<<"B: : f"<<endl; }
      virtual void v()  {cout<<"B: :v"<<endl; }
 };
 int main()  {
     A  a,  *pa  =  &a;
     Pa->f();                          //A
     Pa->V();                          //B
     B  b;     pa  =  &b;
     Pa->f();                          // C
     Pa->V();                          // D
     return  0;
 }
```

66、下列程序段中，ofs 是 ofstream 类的对象，ifs 是 ifstream 类的对象，c 是 char 型变量，则执行结果是(　　)。

```
while (ifs .get(c))
    ofs. put(c);
```

A. 把一个文件的第 1 个字节写入到另一个文件中
B. 把一个文件的内容全部写入到另一个文件
C. 仅仅把字符 c 写入到一个文件中
D. 把两个文件的内容合并到一起

67、下列(　　)循环能够把 2 6 个小写字母以文本方式写入文件中。

```
#include<iostream>
using  namespace  std;
int main()  {
    int  a;
    ofstream  ofs("data. txt");
    a= ' a' ;   while  (a  <=  ' z' )  ofs << a++;          //A
    a = ' a' ;   while  (a  <=  ' z' )  ofs.put(a++);        //B
    a= ' a' ;   while  (a  <=  'z' )  {                      //C
        ofs .write((const  char*)&a,  sizeof (a));
         ++a;
        }
    a=  ' a' ;  while  (a  <=  ' z' )  cout  <<char(a++);     //D
    ofs. close();
 }
```

68、在执行下列程序段时，从键盘输入 Hello, World! ，则(　　)是正确的。

```
char  str [20]
string s;
cin >>str;
getline ( cin,s);
```

A. 字符数组 str 的内容为 Hello。
B. 对象 s 的内容为 Hello, World ! 。
C. 对象 s 的内容为空字符串。
D. 字符数组 str 的内容不包括空格。

分析下列程序，回答 69，70 题。

```
#include<iostream>
using  namespace  std;
class A {
private:
    int x = 1;                              //1
public:
    A(int n) {this->x = n;}                 //2
    A(const A& a) {x = a.x;}                //3
    void Set(int n) const {x = n;}          //4
    void Print() const {cout << x;}         //5
    friend void f(A a) {a.x++;}             //6
};
int main() {
    A a;    a.Set(3);                       //7
    A b(1);  f(b);                          //8
    A c(b);  c.Print();                     //9
```

```
    return  0;
}
```

69、下面关于几个关键字的用法中，说法（　　）是错误的。

A．第 2 行用 this 来访问数据成员 x，也可以这样用：(*this).x=n；

B．第 4 行中 const 用法有误，不能在 const 成员函数中修改数据成员；

C．第 3、5 两行中的 const 都会保护当前对象的数据成员不被修改；

D．第 6 行定义友元函数的目的是使得函数 f 能够访问类的私有数据成员；

70、下列关于类的定义和成员的调用中，说法（　　）是正确的。

A．定义类的同时可以初始化数据成员，如第 1 行所为；

B．由于类 A 中没有默认构造函数，所以第 7 行构造对象 a 错误；

C．第 8 行对友元函数的调用是错误的，应该这样：b.f()；

D．第 9 行调用了类的复制构造函数，且 Print 的输出结果为 2；

二、　判断题

1、构造函数可以声明为虚函数。（　　）

2、一个类的友元函数不是这个类的成员函数。（　　）

3、赋值操作符=重载时只能重载成类的成员函数。（　　）

4、派生类只继承了基类的公有成员和保护成员。（　　）

5、一个对象只能属于一个类。（　　）

6、对于任意一个类，析构函数的个数可以为任意多个。（　　）

7、类的静态数据成员是该类的所有对象共享的数据。（　　）

8、模板类可以直接用来定义对象。（　　）

9、运算符重载可以改变运算符的优先级（　　）

10、类的析构函数在对象消亡时自动调用。（　　）

11、若 T 代表任意数据类型，则语句 T t;定义了 T 类的一个对象。（　　）

12、在类定义体以外对私有成员的访问都是不允许的。（　　）

13、若 T 为自定义类型，则执行 T* p; 语句时不会调用 T 的构造函数。（　　）

14、定义对象数组时一般需要类的默认构造函数。（　　）

15、若以值形式传递函数参数，则需要调用该参数类型的复制构造函数。（　　）

16、赋值运算符函数需要声明为类的 const 成员函数。（　　）

17、在定义 const 对象之后，可通过赋值设定它的值。（　　）

18、执行程序段 T a; T b = a;时，需要调用类 T 的赋值运算符函数。（　　）

19、析构函数可以声明为类的 const 成员函数。（　　）

20、在输出 T 类的对象 t 时 cout<<t，需要把运算符<<定义为类 T 的成员函数。（　　）

21、关于类和对象，类是对象的抽象。（　　）

22、在类声明外不允许直接访问其私有成员。（　　）

23、一个类的拷贝构造函数可定义如下（其中 AB 是一个类名）：

```
AB::AB(const AB &){//…} （　　）
```

24、可定义有默认参数值的函数如下：　（　　）

```
int fun(int a,int b=3,int c){//…}  ;
```

25、类的构造函数的函数名与类名相同并可以重载。（　　）

26、有如下声明语句：

```
const char* ptr;
```

则 ptr 是一个指向 C 风格字符串的常量指针。（　　）

27、不能实例化抽象基类的对象。（　　）

28、有如下类声明：　（　　）

```
class BC{
```

```
    int  x;
   };
```
则可以声明其派生类：
```
  class DC: public BC{
    int z;
  public:
    void Set( int a, int c)   { x=a; z=c;}
    int  Sum() { return x+z;}
  };
```
29、 转型构造函数可以有一个或两个参数。（　　）

30、友元函数是类的成员函数，所以可以直接访问或修改其私有成员。（　　）

31、函数原型中形参的名字可以省略，但是类型一定不能省略。（　　）

32、函数 void f(const T& t);的参数类型 T 只能表示自定义类型。（　　）

33、static 数据成员需要在构造函数中初始化。（　　）

34、根据定义 int n=1，&r=n;，则表达式++r，n--的值为 1。（　　）

35、在构造类 A 的数组 A* a [2]时会调用构造函数 2 次。（　　）

36、执行程序段 T a; T b=a; 时，需要调用类 T 的赋值运算符函数。（　　）

37、析构函数可以声明为类的 const 成员函数。（　　）

38、在输出 T 类的对象 t 时 cout<<t，需要把运算符<<定义为类 T 的成员函数。（　　）

39、类 B 从类 A 公有继承，则程序段 A a; B &b=a; 会引发编译错误。（　　）

40、类 A 中定义有纯虚函数，则函数原型 void f(A a); 是错误的。（　　）

41、仅设置字宽为 4，填充字符为’#’，则表达式 cout<<-10 的输出结果为-#10。（　　）

42、istream 类成员函数 get 既可以输入单个字符，也可以输入整行字符串。（　　）

三、　程序分析

下列各题源代码均已正确包含相关头文件。

1、有如下代码：
```
#include <iostream>
using namespace std;
class Clock{
public: Clock(int k) {     }
};
void f(Clock ck) { // …    }
void main(){
  int a=100;
  f(a);
}
```
请问上述代码是否正确？并解释其原因。

2、分析下面的类体系。
```
#include <string>
using namespace std;
class A{
public:
   void  f(string s) { m=s;}
   string m;
};
```

```
class B: public A{
public:
    void f(int t){m=t;}
    int m;
};
B  b;
```

回答下列问题：

(1)当用语句 b.m 访问数据成员 m 是否会引起二义性？

(2)写出通过对象 b 调用函数 f(string s)和 f(int t)的语句。

3、现定义一个 Triangle 类如下：

```
class Triangle{
    private:
     int side1,side2,side3;
  public:
     Triangle ( ){ side1=side2=side3=1;}
     Triangle (int s1,int s2,int s3){side1=s1; side2=s2; side3=s3; }
  };
  void main()
  {
   Triangle  a(3,4,5);    //(1)
   Triangle  b(a);       //(2)
   Triangle  c;          //(3)
  }
```

请回答（1)、(2)、(3）依次调用了哪个构造函数？

4、分析程序，写出运行结果。

```
#include <iostream>
using namespace std;
void swap(int &x, int y)
{
    x=x+10;  y=y-5;
}
int main(  )
{
    int a=10,b=20;
    cout<<"a="<<a<<"b="<<b<<endl;
    swap(a,b);
    cout<<"a="<<a<<"b="<<b<<endl;
    return 0;
}
```

5、阅读下面的程序，分别写出 const 的作用。

```
class c{
  public:
    void set (const string &n) {name=n;}
    const string & get(  ) const (return name;}
  private:
      string name;
};
```

6、分析下面代码，指出错误并说明原因。

```
class AC{
  public:
      AC(int a){  x=a;  }
  private:
      int x;
};
class CD:public AC{
  public:
      CD(  ){ //… }
   };
```

7、阅读程序，写出运行结果。

```
double arr[5];
double& select(int num) {
    int n = num % 5;
    return arr[n];
}
int main() {
    for(int i = 11; i <= 15; ++i) {
        select(i) = i;
    }
    for(int i = 0; i <= 4; ++i) {
        cout << arr[i] << "\t";
    }
}
```

8、阅读程序，写出运行结果。

```
int fun(char *, char *);
int main()
{
    cout << fun("abcfgy", "abcdhu") << endl;
    cout << fun("abc", "fgy") << endl;
    cout << fun("abcfgy", "abc") << endl;
```

```
}
int fun(char *s1, char *s2) {
    while(*s1 && *s2 && *s1++ == *s2++);
    s1--;
    s2--;
    return *s1 - *s2;
}
```

9、阅读程序，写出运行结果。

```
    class Point {
    private:
        int x, y;
    public:
        Point(int xx = 0, int yy = 0) {
            x = xx; y = yy;
        }
        Point(const Point &p) {
            x = p.x;  y = p.y;
        }
        int GetX() {return x;}
        int GetY() {return y;}
    };
    void f(Point p) {
        cout << p.GetX() << "\t" << p.GetY() << endl;
    }
    Point g() {
        Point a(7, 33);
        return a;
    }
    int main() {
        Point a(15, 22);
        Point b(a);   f(b);
        b = g();      f(b);
    }
```

10、阅读程序，写出运行结果。

```
class A {
private:
    int x;
public:
    A(int n = 0) : x(n) {
        cout << "ctor: x = " << x << endl;
    }
    A(const A& a) {
        x = a.x;
```

```
        cout << "copy ctor: x = " << x << endl;
    }
    ~A() {
        cout << "dtor: x = " << x << endl;
    }
};
int main() {
    A a, b(1),c(b);
}
```

11、阅读程序，写出运行结果。

```
    class Tree {
    public:
        Tree(int height = 0) {
            h = height;
            cout << "ctor: h = " << h << endl;
        }
        ~Tree() {
            cout << "dtor: h = " << h << endl;
        }
    private:
        int h;
    };
    int main() {
        Tree *pdj = new Tree[2];
        pdj[1] = Tree(20);
        delete [] pdj;
    }
```

12、阅读程序，写出运行结果。

```
    class Clock {
    private:
        int hour, minute, second;
    public:
        Clock(int h = 0, int m = 0, int s = 0) {
            hour = h; minute = m; second = s;
        }

        void ShowTime() const {
            cout << hour << ": " << minute
                   << ": " << second << endl;
        }
        Clock& operator ++ () {
            second++;
            if (second >= 60) {
```

```
            second -= 60;
            minute++;
            if (minute >= 60) {
                minute -= 60;
                hour++;
                if (hour >= 24)
                    hour -= 24;
            }
        }
        return *this;
    }
 Clock operator ++ (int) {
        Clock temp(*this);
        ++(*this);
        return temp;
    }
 };
 int main() {
    Clock c1(23, 59, 59), c2(21, 58, 59);
    (++c1).ShowTime(); c1.ShowTime();
    (c2++).ShowTime(); c2.ShowTime();
 }
```

13、阅读程序，写出运行结果。

```
#include<iostream>
using  namespace  std;
class BC{
  public:
       BC(){cout<<"BC  constructor\n";}
      ~BC(){cout<<"BC  destructor\n";}
};
class DC:public BC{
  public:
      DC(){cout<<"DC  constructor\n";}
      ~DC(){cout<<"DC  destructor\n";}
};
class DDC:public DC{
  public:
       DDC(){cout<<"DDC  constructor\n";}
       ~DDC(){cout<<"DDC  destructor\n";}
 };
int main(){
   DDC  d1;
   return  0;
}
```

14、阅读程序，写出运行结果。

```
#include<iostream>
using  namespace  std;
class C{
  public:
    C() {p=new  int;}
    void  set (int  a){*p=a;}
    int get() const  {return  *p;}
  private:
    int  *p;
};
int  main(){
   C  c1,c2;
   c1.set(100);
   c2.set(200);
   cout<<c1.get()<<endl;
   cout<<c2.get()<<endl;
   c2=c1;
   cout<<c1.get()<<endl;
   cout<<c2.get()<<endl;
   c2.set(-999);
   cout<<c1.get()<<endl;
   cout<<c2.get()<<endl;
   return  0;
}
```

15、阅读程序，写出运行结果。

```
#include<iostream>
using  namespace  std;
template<class T>
T sum(T *s,int n){
    T  sum=0;
    for(int  i=0;i<n;i++)
      if(s[i]>0) sum+=s[i];
    return sum;
  }
int  main(){
   int  Iarray[]={2, -3,4,-6,8,7};
   double Darray[]={-1.2,4.3,4.2,-6.5, 8.6, 9.8};
   int  Isum=sum(Iarray,6);
   double Dsum=sum(Darray,6);
   cout<<"Isum="<<Isum<<", Dsum="<<Dsum<<endl;
return  0;
}
```

16、阅读程序，写出运行结果。

```
#include <iostream>
using namespace std;
class C{
public:
C(int k) { x=k; cout<<"C(int) firing "<<endl;}
int  get(){return x;}
private:
int x;
};
void fun(C  c ) {  cout<<c.get()<<endl; }
void main(){
  int a=10;
  fun(a);
}
```

17、阅读程序，写出运行结果。

```
#include <iostream>
using namespace std;
void fun(int m, int& p, int& s) {
p = 1, s = 0;
for (int i = 1; i <= m; i++) {
    p *= i;
    s += p;
}
}
int main() {
int a, b, n;

n = 3;
fun(n, a, b);
cout << a << "\t" << b << endl;
n = 5;
fun(n, a, b);
cout << a << "\t" << b << endl;
}
```

18、阅读程序，写出运行结果。

```
#include <iostream>
int fun(char *, char *);
int main()
{
std::cout << fun("abcfgy", "abcdhu") << std::endl;
std::cout << fun("abc", "fgy") << std::endl;
std::cout << fun("abcfgy", "abc") << std::endl;
}
int fun(char *s1, char *s2) {
while(*s1 && *s2 && *s1++ == *s2++);
s1--;
s2--;
return *s1 - *s2;
}
```

19、阅读程序，写出运行结果。

```
#include<iostream>
class MyClass {
int x, y;
public:
MyClass() { x = y = 0;}
MyClass(int i, int j) {x = i; y = j;}
void copy(MyClass& s);
void setxy(int i, int j) {x = i; y = j;}
void print() {
   std::cout << "x = " << x << ", y = " << y << "\n";
}
};
void MyClass::copy(MyClass &s) {
x = s.x; y = s.y;
}
void func(MyClass s1, MyClass &s2) {
s1.setxy(10, 20);
s2.setxy(30, 40);
}
int main()
{
MyClass p(6, 8), q;
q.copy(p);
func(p, q);
p.print(); q.print();
}
```

20、阅读程序，写出运行结果。

```
#include <iostream>
#include <cassert>
using namespace std;
class String {
private:
char *str;
public:
String(char* s = "") {
    str = new char[strlen(s) + 1];
    assert(str != NULL);
    strcpy(str, s);
    cout << "ctor: " << str << endl;
}
~String() {
    cout << "dtor: " << str << endl;
    delete str;   str = NULL;
}
};
int main() {
String* s1 = new String("Hello");
String* s2 = new String("World");
delete s1;
delete s2;
}
```

21、阅读程序，写出运行结果。

```
#include <iostream>
using namespace std;
class B {
public:
  B(int i,int j){b1 = i, b2 = j;}
  void print() { cout << b1 << ',' << b2 << endl;}
  void print() const { cout << b1 << ':' << b2 << endl;}
private:
  int b1,b2;
};
int main() {
B b1(5,10);
b1.print();
const B b2(2,8);
b2.print();
}
```

22、阅读程序，写出运行该程序后的输出结果。

```
bool f (int n)  {return n% 2 == 0; }
void g(int& n)  {  if  (f (n)   n *= 10; }
void h(int n)  {cout<< n<<"\t"; }
int  main ( )   {
int a[]={2,1,4,3,6,7,5}
int n  = sizeof (a)  / sizeof (*a) ;
reverse (a,  a + 3) ;
reverse (a + 3,  a + n - 1) ;
sort(a + 2, a + n - 2);
for _each (a,  a  +  n,  g) ;
for _each (a,  a  +  n,  h) ;
return 0;
 }
```

23、阅读程序，写出运行该程序后的输出结果。

```
class Object {
    string name;
public :
    Object ( char  *s= "NoName" ) :name (s) {cout<<name<<endl; }

     Object (const  Object&  o) : name (o .name)   { cout<< name<<endl;}
   ~Object ( )   {cout<< name<<endl;}
};
Object  MakeObject ( char * s )   {
    Object obj (s) ;
    return obj;
}
Object  global ( "Global" )
int main ()  {
    MakeObject ( "Local" ) ;
 return 0;
}
```

24、阅读程序，写出运行该程序后的输出结果。

```
class Rational  {
     int nurn, den;
public :
    Rational (int  n  =  0,  int  d  =  1)   :nurn (n) ,  den (d)   {}
    Rational& operator ++ () {
         num += den;
        return  *this;
     }
    Rational  operator ++ (int)  {
         Rational  r ( *this ) ;
         ++(*this)
        return r;
    }
     friend ostream& operator<<(ostream& os,  const Rational& rhs){
        return os<<rhs.num<<'/'<<rhs.den;
    }
};
int main ( )   {
    Rational  ra,  rb(2,  7) ;
   cout<<ra<<"\t"<<rb<<"\n";
ra  = ++rb;
cout<<ra<<"\t"<<rb<<"\n";
   ra  = rb++;
cout<<ra<<"\t"<<rb<<"\n";
return 0;
}
```

25、阅读程序，写出运行该程序后的输出结果。

```
class Base {
   int n;
public :
    Base (int i  = 0) :n(i)  {cout<<"n = "<<n<< endl;}
~Base()  {cout  <<  "n  =  "  <<  n  <<  endl  ;}
};
class Derived : public Base{
    int n;
public :
    Derived (int  i = 1) :n(i)   {cout<< "n  =  "<< n<< endl; }
~Derived ( )   { cout<< "n  =  "<< n<< endl;  }
};
int  main ( )   {
    Derived a,  b (1) ;
    return 0;
}
```

26、阅读程序，写出运行该程序后的输出结果。

```
class Shape {
protected :
     double area;
public :
     double  GetArea ( )  const    return  area; }
    virtual  void  CalcArea ( )  =  0;
    virtual  const  char* WhoAmI ()  = 0;
class Point : public Shape {
public :
     void  CalcArea ( )   { area  =  0; }
     const   char*  WhoAml ( )    { return   " Point " ; }
class Square : public Shape {
     double length;
public :
    Square (double  d)   :length (d)  { }
     void CalcArea ()  { area = length* length; }
     const  char*  WhoAmI( )    return  "Square"; }
};
class Circle : public Shape {
     double radius;
public :
     Circle (double  d)   : radius (d)   { }
     void  CalcArea ( )   {area = 3.14 *radius *radius; }
     const   char*   WhoAml ( )    { return   " Circle "; }
};
int main ()  {
    vector<Shape* >   VS;
     Point  p;        vs.push_back (&p) ;
    Square  s (10) ; vs.push_back (&s) ;
     Circle  c (10) ; vs.push_back (&c) ;
     for  (size_t  i  =  0;  i  <  vs . size () ;  ++i)   {
           cout<<vs[i] ->WhoAml ()<< "\t";
              vs[i] ->CalcArea ( ) ;
           cout<<vs [i] ->GetArea ()<<endl;
    return 0;
 }
```

27、阅读程序，写出运行该程序后的输出结果。

```
bool f (int i)  {return  ! (i %3) ; }
int main ()  {
    ofstrearn  ofs ( "nurns . dat" ) ;
     int n  = 10;
     while (--n) ofs<<n<<" t";
     ofs . close ( ) ;
int x. y = 0, z = 0;
ifstrearn  ifs ( "nurns . dat" )
while  ( !ifs.eof ()) {
     ifs >>x;
     if(f(x))   {
     ++z,  y+=x;
cout<< x <<”\t”;
    }
}
Cout<< y / z<<endl;
ifs . close ( ) ;
}
```

四、程序设计

1、 编写程序，用 vector 存放 10 个整型数值，并对数据按升序排序。

2、 定义一个学生类，其中有 3 个数据成员：学号、姓名、年龄，以及若干成员函数。同时编写 main 函数使用这个类，实现对学生数据的赋值和输出。

3、现有如下类：

```
class Point {
  private:
        int  x;
        int  y;
          };
```

请完成下列编程工作：(把本题程序写成一个完整的程序也可，不一定分步书写。)

（1)、添加合适的构造函数，完成数据成员的初始化；

（2)、重载操作符＝＝，实现两个 Point 对象是否全等的比较运算；

（3)、重载操作符<<和>>，实现 Point 对象的输入输出运算。

4、声明一个 Shape 抽象类，在此基础上派生出 Rectangle 和 Circle 类，二者都有虚函数 GetArea()函数计算对象的面积，在主函数中通过基类指针调用派生类的虚函数，计算不同形状的面积。

5、实现一个复数类 **Complex**，该类中包括实部 real 和虚部 imag 两个实型数据成员，并具有下述功能：(把本题程序写成一个完整的程序也可，不一定分步书写。)

（1） 添加构造函数，初始化其数据成员。

（2） 重载+算术运算符。

（3） 重载==关系运算符。

（4） 重载>>流提取运算符，从键盘输入复数的实部和虚部。

（5） 重载<<流插入运算符，以标准格式输出一个复数，并当虚部为零时，只输出实部。

测试程序如下：

```
int main() {
  Complex  a(1.5,2.5), b;
 cin>>b;// 需重载>>提取运算符
  if(a==b)   //需重载==关系运算符
     cout<<"a=b="<<b<<endl;//需重载<<插入运算符
  else
     cout<<"a+b="<<(a+b)<<endl;//需重载+算术运算符
 return  0;
   }
```

6、实现一个模型的类层次结构：声明一个公共基类 **Model** 并派生出立方体（**Cube**）和圆柱体（**Cylinder**）两个类。(把本题程序写成一个完整的程序也可，不一定分步书写。)

要求：

（1）声明 **Model** 为抽象基类，在其中声明一个求体积的纯虚函数（volume）,且该类中具有一个数据成员 m ，此数据可作为立方体的边长、圆柱体底面圆半径。

（2）在两个派生类中添加适当的数据成员和构造函数。

（3）两个派生类都要覆盖抽象基类 **Model** 的求体积的纯虚函数（体积公式：立方体 m^3、圆柱体 $\pi m^2 h$)。在主函数中声明两个派生类的对象以及 **Model** 类型的指针，并通过该指针调用 volume 函数计算不同模型的体积。

7、基本算法设计与应用。

(1)对 STL 算法 sort 的调用 sort(a，a+n，cmp)可以对区间[a，a+n)（即从 a 开始的 n 个元素）按照准则 cmp 排序。请自定义函数模板 mysort 实现这一功能，不能直接调用 sort 实现。

(2)现有区间[s，s+n)，其元素是如下定义的类 Student 的对象。请应用上述函数模板 mysort 对区间元素按照成绩 score 从高到低排序。

```
class Student  {
public：
string name;   / /姓名
int    id;     / /学号
double score;/ /成绩
);
```

8、类的定义与应用。

把本题程序写成一个完整的程序也可，不一定分步书写。

(1)三维空间中的一点可以用坐标(x，y，z)。定义类 Point 描述点，提供构造函数，并重载减法运算符“-”计算两点之间的距离。

(2)三维的球可以由圆心点 center 和半径 radius 决定。定义类 Sphere 描述琼，定义构造函数，并提供成员函数计算两球的距离。

(3)地球半径设为 re，月球半径设为 rm，某时刻它们分别处于点 pe 和 pm，计算它们的距离。

9、继承与多态性的应用。

把本题程序写成一个完整的程序也可，不一定分步书写。

(1)每个计算机配件 ComputerAccessory 都有制造商 manufaceurer 和价格 price 两种属性。请定义类 ComputerAccessory，并提供必要的操作。

(2)主板 MotherBoard、内存 Memory、显示器 Monitor 是典型的计算机配件。芯片组 chipset、内存容量 capacity、显示器类型 mtype 分别是这三种配件的重要特征。请定义类 MotherBoard、Memory、Monitor，并提供必要的操作。

(3) 主板、内存、显示器是计算机 Computer 的重要组成部分，请定义类 Computer，并提供必要的操作。现配置一台计算机（芯片组为 Intel 的主板，2G 内存，显示器类型为 LCD)，请计算其价格（暂不考虑其他配件的价格)。

10、基本算法设计与应用。

分别定义函数模板：(1）在数组 a 的前 n 个元素中查找某个元素 k，如果找到就返回其下标，否则返回-1 表示失败。(2）对于找到的元素统计它出现的次数。(3）在 main 函数中定义一组数据测试这两个函数模板。

11、类的定义与应用。(把本题程序写成一个完整的程序也可，不一定分步书写。)

（1）分数 Fraction 由分子 num 和分母 den 组成，请定义类描述分数，并在类中提供必要的操作。

（2）在 main 函数中定义两个分数对象，调用类中的成员函数输出这两个对象。

（3）在类中重载加、减、乘、除四个算术运算符，并对 main 函数中的两个对象分别计算和、差、积、商。

12、类的定义与应用。

每个学生有学号 id、姓名 name 和成绩 score 等属性。某班有 30 名学生，请计算他们的平均成绩，然后输出该班成绩最好的 10 个人的所有信息。

13、定义一组重载函数，分别对类型为 int、long 和 double 的两个数求和，要求函数返回所求出的和。

14、数组 a 有 n 个元素，定义一个函数模板将其中的元素逆转。

15、按照要求定义类及其成员函数

（1）定义一个试卷类 TestPaper，其属性有学生的姓名（name）和成绩（score)，在类中提供必要的成员函数。

（2）在 main 函数中实例化一个对象 t，并调用相应的成员函数输出学生的姓名和成绩。

（3）对于某班的若干个学生，计算其平均成绩。