假定 AB 引用是某个变量的别名，对引用的操作就是对被引用变量的操作。

 A. 对  
 B. 错



假定 AB 为一个类，则执行 AB x; 语句时将自动调用该类的无参构造函数。

 A. 对  
 B. 错



构造函数用来初始化类的新对象，构造函数与类同名，返回类型只能为void。

 A. 对  
 B. 错



用new动态分配的内存是在堆中分配的，而不是在栈中分配的。

 A. 对  
 B. 错



使用关键字class定义的类中缺省的访问权限是私有（private）的。

 A. 对  
 B. 错



在公有继承中，基类中的公有成员和私有成员在派生类中都是可见的。

 A. 对  
 B. 错



C++语言的标识符是不区分大小写的。

 A. 对  
 B. 错



作用域运算符（::）只能用来限定成员函数所属的类。

 A. 对  
 B. 错



声明对象时，类名前面不需要加class关键字。

 A. 对  
 B. 错



C++程序一般应当含有main函数，它是C++程序执行的入口。

 A. 对  
 B. 错



可以在类的构造函数中对静态数据成员进行初始化。

 A. 对  
 B. 错



C++语言中，既允许单继承，又允许多继承。

 A. 对  
 B. 错



派生类的继承方式有两种：公有继承和私有继承。

 A. 对  
 B. 错



多继承情况下，派生类的构造函数的执行顺序取决于定义派生类时所指定的各基类的顺序。

 A. 对  
 B. 错



类的私有成员只能被类中的成员函数访问，任何类以外的函数对它们的访问都是非法的。

 A. 对  
 B. 错



构造函数和析构函数都不能重载。

 A. 对  
 B. 错



在基类中被声明为虚函数的类的成员函数必须在每个派生类中显式声明为虚函数，才能具有多态的特征。

 A. 对  
 B. 错



关于运算符重载，运算符函数的返回类型不能声明为基本数据类型。

 A. 对  
 B. 错



将类的一个对象赋值给该类的另一个对象时，调用拷贝构造函数。

 A. 对  
 B. 错



模板可以使程序对任何数据类型进行同样方式的处理。

 A. 对  
 B. 错



1. 编译源文件后 产生的文件是（    ）。

 A. 目标文件  
 B. 可执行文件  
 C. 源文件  
 D. 头文件



2. 每个C++程序只能有一个（   ）。

 A. 全局变量  
 B. 函数  
 C. 头文件  
 D. 主函数



3. 使用内联函数的好处是（   ）。

 A. 增加函数调用次数  
 B. 提高代码执行效率  
 C. 减少代码量，增强代码可读性  
 D. 隐藏内部实现



4.下列不是函数重载所要求的条件是（   ）。

 A. 函数名相同  
 B. 函数返回值类型不同  
 C. 参数类型不同  
 D. 参数个数不同



5. 下列哪两个函数为重载函数（   ）。

 A. void a1(), void a2()  
 B. intb(int x), float b(int x)  
 C. char c(double y), char c(int y, double z)  
 D. bool d(double y), bool d(double y, int x = 0)



6. 在面对对象方法中，实现信息隐蔽依靠的是（   ）。

 A. 对象的封装  
 B. 对象的继承  
 C. 对象的多态  
 D. 对象的分类



7. 所谓数据封装就是将一组数据和与这组数据有关操作组装在一起，形成一个实体，这实体也就是（   ）。

 A. 类  
 B. 对象  
 C. 函数体  
 D. 数据块



8. 下列关于类定义的说法中，正确的是（   ）。

 A. 在类里定义的函数为内联函数  
 B. 类成员的默认访问权限是保护的  
 C. 数据成员必须被声明为私有的  
 D. 成员函数只能在类体外进行定义



9. 关于类和对象不正确的说法是（   ）。

 A. 类是一种类型，它封装了数据和操作  
 B. 对象是类的实例  
 C. 一个类的对象只有一个  
 D. 一个对象必属于某个类



10. 结构体中定义的成员默认的访问属性为（   ）。

 A. public  
 B. private  
 C. friend  
 D. protected



11.类中定义的成员默认的访问属性为（   ）。

 A. public  
 B. private  
 C. friend  
 D. protected



12. 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



13. 下列关于析构函数的描述中，错误的是（   ）。

 A. 析构函数可以重载  
 B. 析构函数由系统自动调用  
 C. 每个对象的析构函数只被调用一次  
 D. 每个类都有析构函数



14. 通常拷贝构造函数的参数是（   ）。

 A. 对象名  
 B.  对象的成员名  
 C. 对象的引用名  
 D. 对象的指针名



15. 在类外定义成员函数时，需要在函数名前加上（   ）。

 A. 类名  
 B. 对象名  
 C. 作用域运算符  
 D. 类名和作用域运算符



16. 在类定义中，为说明成员的访问权限，private， protected，public可以出现次数（   ）。

 A. 次数没有具体限制  
 B. 每种至多一次  
 C. public 至少一次  
 D. 每种至少一次



17. 关于静态数据成员的描述中错误的是（   ）。

 A. 类的静态数据成员是该类所有对象所共享的  
 B. 类的静态数据成员在在该类每个对象中都保留一个备份  
 C. 类的静态数据成员必须进行初始化  
 D. 类的静态数据成员可以不通过对象来访问



18. 若一个类的数据成员用 const 关键字修饰，则该数据成员（   ）。

 A. 静态函数中可以对该成员赋值  
 B. 构造函数只能通过初始化列表对该成员初始化  
 C. 任何函数都不能使用该成员  
 D. 外部函数可以对该成员赋值



19. 有两个类 X和Y，且 Y 有一个构造函数 Y::Y(X x); 那么以下语句：

X a;

Yb(a);

不会调用下面的哪个函数（    ）。

 A. Y 的构造函数 Y(X x)  
 B. Y 的拷贝（复制）构造函数  
 C.  X 的默认构造函数  
 D. X 的拷贝（复制）构造函数



20. 已知类 A 的定义。假设 x 为 A 的对象，则在类外访问 x 对象中数据成员a和b分别应采用（   ）。

class A {

private:

    int a;

public:

    int getValue() {

        return a;

    }

    int b;

};

 A. x::getValue(), x::b  
 B. x.a, x.b  
 C. x.a(), x.b()  
 D. x.getValue(), x.b



21. 假设程序中先前已经定义了 A、B、C 三个类，他们的构造函数分别会输出A constructed、B constructed 和C constructed，那么又定义了类 D，并在 main 函数中定义了一个D 类的对象，部分代码如下：

class D{

public:

    D():a(), b(), c() { }

private:

    A a;

    B b;

    C c;

};

intmain() {

    D d;

    return 0;

}

程序的输出结果为（    ）。

 A. A constructed B constructed C constructed  
 B. C constructed B constructed A constructed  
 C. 编译错误  
 D. 编译器决定



22. 已知类 CA 中的某个成员是类 CB 的对象，若要把这两个类定义在同一个文件中，则（   ）。

 A. CA 必须定义在 CB 之前  
 B. CB 必须定义在 CA 之前  
 C. 若 CA 定义在 CB 之后，则 CA 的声明必须出现在 CB 的定义之前  
 D. 若 CB 定义在 CA 之后，则 CB 的声明必须出现在 CA 的定义之前



23. 在设计一个类的时候，如果希望允许一个不属于该类的函数访问该类的私有成员，应该使用哪个关键字（   ）。

 A. public  
 B. private  
 C. friend  
 D. protected



24. 关于友元函数的描述中,错误的是（   ）。

 A. 友元函数是成员函数  
 B. 友元函数能访问类中私有成员  
 C. 友元函数破坏封装性  
 D. 友元函数声明在类体内



25. 如果类 A 是类 B 的友元，则（   ）。

 A. 类 A 的成员是类 B 的成员  
 B. 类 B 的成员是类 A 的成员  
 C. 类 A 的成员函数可以访问类 B 的私有成员  
 D. 类 B 是类 A 的友元



26. 已知类 A 是类 B 的友元类，类 B 是类 C 的友元类，则（   ）。

 A. 类 A 一定是类 C 的友元  
 B. 类 C 一定是类 A 的友元  
 C. 类 C 的成员函数可以访问类 B 的对象的任何成员  
 D. 类 A 的成员函数可以访问类 B 的对象的任何成员



27. 已知Point类含有无参（默认）构造函数和有参构造函数，使用Point a[3] = { Point(1, 2), Point(3,4), Point() }; 时，调用了（   ）次Point类的无参（默认）构造函数。

 A. 1  
 B. 2  
 C. 3  
 D. 4



28. 要定义一个引用变量p使之引用类MyClass的一个对象,正确的定义语句是（   ）。

 A. MyClass p = MyClass;  
 B. MyClass p = new MyClass;  
 C. MyClass &p = new MyClass;  
 D. MyClass a, &p = a;



29. 关于this指针的说法不正确的是（   ）。

 A. 不能在程序中修改this指针  
 B. this指针可以给其他指针赋值，但不能修改this指针  
 C. 静态成员函数中没有this指针  
 D. this指针可以被赋值



30. 关于C++的动态存储分配，下列说法正确的是（   ）。

 A. new和delete是C++语言中专门用于动态内存分配和释放的函数  
 B. 动态分配的内存空间也可以被初始化  
 C. 当系统内存不够时，会自动回收不再使用的内存单元，因此程序中不必用delete释放内存空间  
 D. 当动态分配内存失败时，系统会立刻崩溃，因此一定要慎用new



31. 观察程序

#include<iostream>

using namespace std;

int main() {

    int \*first, \*second;

    first = new int[5];

    second = first;

    delete[] second;

    delete[] first;

    return 0;

}

该程序是否正确，原因是什么 ？

 A. 正确，因为first和second都是独立的指针变量  
 B. 不正确，因为first和second是相关而非独立的指针变量  
 C. 正确，second会拷贝first的内容  
 D. 不正确，second和first指向相同的内容



32. C++中，不能被派生类继承的有（   ）。

 A. 析构函数  
 B. 虚函数  
 C. 静态成员函数  
 D. 友元函数



33. 下列关于派生类的叙述中，错误的是（   ）。

 A.  派生类至少要有一个基类  
 B. 派生类中包括了从基类继承的成员  
 C. 一个派生类可以作为另一个派生类的基类  
 D. 基类成员被派生类继承以后访问权限保持不变



34. 下列关于类型转换正确的是（   ）。

 A. 基类的对象可以被当作公有派生类对象使用  
 B. 派生类对象必须显式的转换为基类对象  
 C. 派生类对象可以初始化基类的引用  
 D. 基类的指针可以隐含转换为派生类的指针



35. 下列关于虚基类的描述中，错误的是（   ）。

 A. 使用虚基类可以消除由多继承产生的二义性  
 B. 构造派生类对象时，虚基类的构造函数只被调用一次  
 C. 声明“classB:virtual public A”说明类B为虚基类  
 D. 建立派生类对象时，首先调用虚基类的构造函数



36. 以下说法正确的是（   ）。

 A. 同一个运算符可以被重载多次  
 B. 可以通过运算符重载来创建C++中原来没有的运算符  
 C. 自增运算符重载时，重载函数不允许有输入参数  
 D. 运算符函数的返回类型不能声明为基本数据类型



37. 若A是一个类名，则如下语句将会产生的A对象个数为（   ）。

A a1,\*a2;

A \*a3= new A();

A& a4 = a1;

 A. 1  
 B. 2  
 C. 3  
 D. 4



38. 类C2是类C1的派生类，下列说法中错误的是（   ）。

 A. C1的成员都是C2的成员  
 B. C2的对象可以赋值给C1的变量  
 C. C1的友元函数都是C2的友元函数  
 D. 创建C2的对象时，要执行C1的构造函数



39. 若类A和类B都是类模板T的模板类，则以下说法正确的是（   ）。

 A. A的友元函数中，可以访问B的私有成员  
 B. A的对象可以赋值给B类型的变量  
 C. A和B共享T的静态成员变量  
 D. A的成员函数中，不能访问B的私有成员



40.若类B是从类A派生而来，而语句 p.Do(); 在被多次执行时，有的时候调用的是类A的Do成员函数，有的时候调用的是类B的Do成员函数，那么 p 的类型可以是（   ）。

 A. A &  
 B. A \*  
 C. B  
 D. A



41. 将一个对象放入STL中的容器里时，（   ）。

 A. 实际上被放入的是该对象的一个拷贝（副本）  
 B. 实际上被放入的是该对象的指针  
 C. 实际上被放入的是该对象的引用  
 D. 实际上被放入的就是该对象自身



42. 以下说法正确的是（   ）。

 A. 在multiset容器中，可以同时存储多个相同的元素  
 B. 在set容器中，各元素的存储位置取决于它们的插入顺序  
 C. 向deque容器添加新对象时，需要比较对象的大小  
 D. 在list容器中，各元素的存储位置取决于它们的值



43. 若有如下定义：

set<int>::iterator it1, it2, it3;

set<int> s;

则以下语句不符合语法的是（    ）。

 A. it1++;  
 B. --it2;  
 C. it3 = it1 + 1;  
 D. boolisEnd = (it1 == s.end());



44. 关于函数对象，如下说法中正确的是（   ）。

 A. 如果一个类重载了“()”运算符，该类的对象是函数对象  
 B. 如果一个类定义了函数指针成员变量，则该类的对象是函数对象  
 C. 函数对象类中只能定义成员函数，不能定义成员变量  
 D. 函数对象类只能定义一个函数对象



45. 有如下两个类定义：

class AA { };

class BB {

    AA v1, \*v2;

    BB v3;//嵌套定义

    int v4;

}

其中有一个成员变量的定义是错误的，这个变量是（    ）。

 A. v1  
 B. v2  
 C. v3  
 D. v4



46. 对于如下代码，请指出错误的代码行号是（    ）。

1   class A {

2   public:

3       A(int a = 0) {

4           x = a;

5       }

6       static void f1() {

7           x++;//只能调用静态数据成员

8       };

9   private:

10      int x;

11      static int y;

12  };

 A. 3  
 B. 4  
 C. 7  
 D. 11



47. 对于如下代码

1   class B {

2   public:

3       void f() { }

4       void g() { f(); }

5   }

6   class D : public B {

7   public:

8       void f() { }

9       void g() { B::f() }

10  }

11  int main() {

12      B b, pb = &b;

13      D d, pd = &d;

14      b.f();

15      pd->f();

16      return 0;

17  }

其中第14行代码和第15行代码分别调用的是（    ）。

 A. B::f() 与 D::f()  
 B. B::f() 与 B::f()  
 C. D::f() 与 D::f()  
 D. D::f() 与 B::f()



48. 以下程序的输出结果是（   ）。

#include<iostream>

using namespace std;

class A {

public:

    A(int a1, int a2):a1(a1), a2(a2) { }

    void print();

    void print() const;

private:

    int a1, a2;

};

void A::print() {

    cout << a1 << " ";

}

void A::print()const {

    cout << a2 << " ";

}

int main() {

    A a(1, 2);

    const A b(1, 2);

    a.print();

    b.print();

    return 0;

}

 A. 1 1  
 B. 1 2  
 C. 2 1  
 D. 2 2



49. 有如下程序：

#include<iostream>

using namespace std;

class A {

private:

    int a;

public:

    A(int a = 0):a(a) {

        cout << "1";

    }

    A(A &x) {

        a = x.a;

        cout << "2";

    }

    ~A() {

        cout << a;

    }

    int get\_a() {

        return a;

    }

};

int main() {

    A d1(5), d2(d1);

    A \*pd = new A(8);

    cout << pd->get\_a();

    delete pd;

    return 0;

}

程序运行时的输出结果是（    ）。

 A. 12185  
 B. 1215588  
 C. 128512  
 D. 1218855



50. 有如下程序：

#include<iostream>

using namespace std;

class A {

public:

    void setA(int);

private:

    int a;

};

class B {

public:

    void setB(int);

private:

    int b;

};

class C:public A,private B {

public:

    void setC(int, int, int);

private:

    int c;

};

void A::setA(intx){

    a = x;

}

void B::setB(intx){

    b = x;

}

void C::setC(int x, int y, int z) {

    setA(x);

    setB(y);

    c = z;

}

int main() {

    C obj;

    obj.setA(5);       // A

    obj.setB(6);       // B

    obj.setC(6, 7, 9); // C

    return 0;

}

main函数中（    ）是错误的。

 A.  A语句  
 B. B语句  
 C. C语句



51. 设置虚基类的目的是（   ）。

 A. 区分类是否有虚函数  
 B. 使类成为抽象类  
 C. 提高程序运行效率  
 D. 消除二义性



52. 运行时多态是基于（   ）。

 A. 友元函数  
 B. 虚函数  
 C. 函数重载  
 D. 运算符重载



53. C++泛型编程通过（    ）实现。

 A. class  
 B. template  
 C. virtual  
 D. typedef



54. 关于函数声明 float fun(int a, int b) throw，下列叙述正确的是（   ）。

 A. 表明函数抛出 float 类型异常  
 B. 表明函数抛出任何类型异常  
 C. 表明函数不抛出任何类型异常  
 D. 语法错误



55. 有如下两个程序片段：

#include<vector>

//程序片段一

std::vector<int>vec;

for(auto itr =vec.begin(); itr != vec.end(); itr++)

itr->print();

//程序片段二

std::vector<int>vec;

for(autoitr = vec.begin(); itr != vec.end(); ++itr)

itr->print();

则下列说法中，正确的有（    ）。

(a)itr的类型为auto

(b)程序二性能上要比程序一更好

(c)程序一和程序二性能上没有区别

 A. ab  
 B. b  
 C. ac  
 D. c



56. 如果 X 是整型变量，则合法的形式是（   ）。

 A. &(X+1)  
 B. \*X  
 C. \*&X  
 D. &\*X



57. 判断以下程序的正确性，并说明原因。

#include <iostream>

using namespace std;

int main() {

int \*first, \*second;

first = new int[5];

second = first;

delete[] second;

delete[] first;

return 0;

}

 A. 正确，因为 first 和 second都指向动态内存，需要手动释放  
 B. 不正确，应该先释放 first 指向的内存空间  
 C. 正确，second 会拷贝 first 的内容  
 D. 不正确，second 和 first 指向相同的动态内存，重复释放会可能导致未定义行为



58. C++中既可以用于文件输入又可以用于文件输出的流类是（    ）。

 A. fstream  
 B. ifstream  
 C. ofstream  
 D. iostream



59. 下列是关于函数模板的程序，则下列说法中正确的有（   ）。

#include <iostream>

using namespace std;

template <typename T> void f(T &i) { cout << "1"; }

template <> void f(const int &i) { cout << "2"; }

int main() {

int i = 42;

f(i);

return 0;

}

(a)程序输出1

(b)程序输出2

(c)template <>void f(const int &i) 显式实例化void f(const int &)

(d)f(i)隐式实例化函数void f(int &)

 A. ad  
 B. bc  
 C.  acd  
 D. bcd



60. 下列程序的输出为（   ）。

#include <iostream>

using namespace std;

class Shape {

public:

virtual void print() {

cout << "SHAPE" << endl;

}

virtual ~Shape() { }

};

classBox : public Shape {

public:

virtual void print(int i) {

cout << "BOX" << endl;

}

};

int main() {

Shape \*s = new Box();

s->print();

delete s;

return 0;

}

 A. SHAPE  
 B. BOX  
 C. 未定义行为



1、从完成C++源文件编辑，到生成执行文件，C++编译器必须执行的步骤依次为编译、链接。

 A. 对  
 B. 错



C++是强类型语言。

 A. 对  
 B. 错



3、如果a为int型变量，b为int型变量，则a/static\_cast<double>(b)的值为int型。

 A. 对  
 B. 错



4、数组可以通过值参数和引用参数两种方式传递给函数。

 A. 对  
 B. 错



5、当一个函数在运行过程中产生一个异常，则这个函数会终止，但是整个程序不一定终止运行。

 A. 对  
 B. 错



6、构造函数和析构函数都不能重载。

 A. 对  
 B. 错



7、构造函数可以声明为虚函数。

 A. 对  
 B. 错



8、已知类X成功地重载了--、=、+和[]这几个运算符，其中肯定属于成员函数的运算符是=和[]。

 A. 对  
 B. 错



9、在声明一个对象时如果使用同一类类型的另一个对象来初始化，这时调用的是该类的拷贝构造函数。

 A. 对  
 B. 错



10、C++语言中，既允许单继承，又允许多继承。

 A. 对  
 B. 错



11、派生类的继承方式有两种：公有继承和私有继承。

 A. 对  
 B. 错



12、派生类是从基类派生出来的，它不能再生成新的派生类。

 A. 对  
 B. 错



13、构造函数可以被继承，析构函数不能被继承。

 A. 对  
 B. 错



14、在公有继承中，基类中的公有成员和私有成员在派生类中都是可见的。

 A. 对  
 B. 错



15、在公有继承中，基类中只有公有成员对派生类对象是可见的。

 A. 对  
 B. 错



16、某类的友元类中的所有成员函数可以存取或修改该类的私有成员。

 A. 对  
 B. 错



17、多继承情况下，派生类的构造函数的执行顺序取决于定义派生类时所指定的各基类的顺序。

 A. 对  
 B. 错



18、虚函数和普通函数的区别在于函数名字与函数体之间的绑定方式不同，普通函数使用的是静态绑定，而虚函数使用的是动态绑定。

 A. 对  
 B. 错



19、在抽象类中定义的函数只能是纯虚函数。

 A. 对  
 B. 错



20、模板对类型参数化提供了很好的支持，类模板实例化时，编译器将根据给出的模板实参生成一个类。

 A. 对  
 B. 错



1. 算法是一个计算过程，是程序设计的基础和精髓。一个有效的算法应该具有如下特点，其中错误的是（   ）。

 A. 零个或多个输入及一个或多个输出  
 B. 有效性  
 C. 无穷性  
 D. 确定性



2. #include<file.h>与 #include"file.h"的区别是（   ）。

 A. 前者首先从当前工作路径搜寻并引用file.h，而后者首先从标准库的路径寻找和引用file.h  
 B. 前者首先从标准库的路径寻找和引用file.h，而后者首先从当前工作路径搜寻并引用file.h  
 C. 两者都是首先从标准库的路径寻找和引用file.h  
 D. 两者都是首先从当前工作路径搜寻并引用file.h



3. 头文件中的ifndef/define/endif的作用是（   ）。

 A. 定义常量  
 B. 标记为特殊的头文件  
 C. 防止头文件被重复引用  
 D. 注释头文件



4. 在一个程序中，如果函数fA调用了函数fB，函数fB又调用了函数fA，那么（    ）。

 A. 称为函数的循环调用  
 B. 这样调用方式是不允许的  
 C. 称为函数的直接递归调用  
 D. 称为函数的间接递归调用



5. 引入内联函数的目的是为了（    ）。

 A. 方便变量的使用  
 B. 提高函数调用的效率  
 C. 扩大局部变量的使用范围  
 D. 节省内存空间



6. 当给函数的参数设置默认值时，正确的说法是（   ）。

 A. 若给某一参数设置了默认值，那么在参数表中其前所有的参数都必须也设置默认值  
 B. 若给某一参数设置了默认值，那么在参数表中其后所有的参数都必须也设置默认值  
 C. 给函数的参数设置默认值可以提高程序运行效率  
 D. 任何情况下，函数调用时均可不列出已设置默认值的参数



7.下列说法正确的是（   ）。

int \*p1 =new int[10];

int \*p2 =new int[10]();

 A. p1和p2申请的空间里面的值都是随机值  
 B. p1和p2申请的空间里的值都已经初始化  
 C. p1申请的空间里的值是随机值，p2申请的空间里的值已经初始化  
 D. p1申请的空间里的值已经初始化，p2申请的空间里的值是随机值



8. 按照常理，下列概念可以理解为对象而不是类的是（   ）。

 A. 国际知名学府  
 B. 国家重点建设大学  
 C. 诺贝尔奖获得者  
 D. 杭州师范大学



9. 在面向对象程序设计中，实现信息隐蔽是靠（    ）。

 A. 对象的封装  
 B. 对象的继承  
 C. 对象的间接访问  
 D. 对象的分类



10. 下列有关类的说法，错误的是（    ）。

 A. 在类中，如果不作特别说明，所有的数据成员均为私有数据成员  
 B. 类是一种用户自定义的数据类型  
 C. 在类中，如果不作特别说明，所有的成员均为公有成员  
 D. 只有类中的成员函数或类的友元函数才能存取类中的私有数据



11. 下面有关类与对象的说法中，错误的是（    ）。

 A. 对象是类的实例  
 B. 类和对象的关系与数据类型和变量的关系相似  
 C. 一个对象不能同时属于多个没有关系的独立的类  
 D. 一个类只能有一个对象



12. 下列关于构造函数说法，错误的是（    ）。

 A. 构造函数必须与类同名  
 B. 构造函数必须有返回值  
 C. 构造函数可以省略不写  
 D. 在构造函数中可以对类中的成员进行初始化



13. 以下有关析构函数的叙述，错误的是（   ）。

 A. 析构函数名前必须冠有符号“~”  
 B. 在一个类只能定义一个析构函数  
 C. 析构函数不允许用返回值  
 D. 析构函数和构造函数一样可以有形参



14. 关于类的对象，说法错误的是（   ）。

 A. 释放对象时，相应类的析构函数被自动执行  
 B. 在成员函数中，只能访问成员函数所作用的对象的private成员  
 C. 对象的成员变量可以是其他类的对象  
 D. 同类的对象间，可以互相赋值



15. 设已经有A、B、C、D4个类的定义，程序中A、B、C、D析构函数调用顺序是（   ）。

C c;

int main() {

A \*pa = new A();

B b;

static D d;

delete pa;

}

 A. A B C D  
 B. A B D C  
 C. A C D B  
 D. A C B D



16. 以下说法中正确的是（   ）。

 A. 构造函数中可以调用本类的静态成员函数  
 B. 构造函数和析构函数都可以是虚函数  
 C. 通过赋值号“=”用一个对象对另一个对象进行赋值，就会引发复制构造函数的调用  
 D. 用new运算符动态创建对象时，用于初始化该对象的一定是无参数的构造函数



17. 下列情况中，不会调用对象析构函数的是（   ）。

 A. 将对象指针强行赋值为NULL时  
 B. 函数中定义的动态对象变量当函数执行结束时  
 C. 用delete删除对象指针时  
 D. 当对象变量脱离其作用域时



18. 关于类的静态成员，说法错误的是（   ）。

 A. 在类的静态成员函数中，可以访问该类的静态成员变量  
 B. 在类的非静态成员函数中，不可以访问该类的静态成员变量  
 C. 不需要创建类的对象，就可以访问类的静态成员  
 D. 类的静态成员变量由该类的全部对象共有



19. 假设Complex是一个复数类，下面（   ）将不会引起拷贝构造函数的调用。

 A. Complexc1(5); Complex c2(c1);  
 B. Complexc1(5); Complex c2 = c1;  
 C. Complexc1(5); Complex c2[2] ={ c1, 2};  
 D. Complexc1(5), c2; c2 = c1;



20. 关于常量成员和静态成员，下面说法正确的是（   ）。

 A. 从同一个类模板得到的两个模板类可以共享同一个静态成员  
 B. 公有的静态成员在没有对象生成的时候也能被直接访问  
 C. 常量成员函数只有常量对象可以调用  
 D. 在静态成员函数中不能访问非静态成员，但可以访问常量成员



21. 下列关于运算符重载的说法中，错误的是（    ）。

 A. new和delete运算符可以重载  
 B. 重载运算符不能改变其原有的操作数个数  
 C. 三元运算符“?:”不能重载  
 D. 所有运算符既可以作为类的成员函数重载,又可以作为非成员函数重载



22. 若需要为xv类重载乘法运算符，运算结果为xv类型，在将其声明为类的成员函数时，下列原型声明正确的是（   ）。

 A. xv operator\*(xv,xv);  
 B. xv\*(xv);  
 C. operator\*(xv);  
 D. xv operator\*(xv);



23. 已知表达式++a中的“++”是作为成员函数重载的运算符，则与++a等效的运算符函数调用形式为（   ）。

 A. a.operator++(1)  
 B. operator++(a)  
 C. operator++ (a,1)  
 D. a.operator++()



24. 将运算符重载为类成员函数时，其参数表中没有参数，说明该运算符是（    ）。

 A. 不合法的运算符  
 B. 一元运算符  
 C. 无操作数的运算符  
 D. 二元运算符



25. 将运算符“+”重载为非成员函数，下列原型声明中,错误的是（   ）。

 A. MyClock operator+(MyClock, long);  
 B. MyCIock operator+(MyClock, MyCIock);  
 C. MyClock operator+(long, long);  
 D. MyCIock operator+(long,MyClock);



26. 下列对派生类的描述中，错误的是（    ）。

 A. 一个派生类可作为另一个派生类的基类  
 B. 派生类中继承的基类成员的访问权限到派生类保持不变  
 C. 派生类至少有一个基类  
 D. 派生类成员除了它自己的成员外，还包含它的基类成员



27. 当派生类从基类公有继承时，以下说法错误的是（    ）。

 A. 可以把指向派生类对象的指针赋给基类对象的指针  
 B. 可以用基类对象为派生类对象赋值  
 C. 可以把派生类对象的地址赋给基类对象的指针  
 D. 可以用派生类对象为基类对象赋值



28. 由于以下（    ）原因，在编程实践中很少使用私有继承。

 A. 经过私有继承之后，基类的成员再也无法在以后的派生类中直接发挥作用  
 B. 私有继承不能很好的保护基类成员的安全性  
 C. 私有继承对基类的编写方法有更为严格的要求  
 D. 私有继承的性能低于公有继承



29. 以下说法正确的是（   ）。

 A. 通过指针访问对象的虚函数时，所执行的函数体由指针的类型决定  
 B. 在基类的函数成员中，可以利用多态机制访问派生类的成员函数  
 C. 派生类的对象中不包括基类的private成员  
 D. 类B是类A的派生类，若函数f是A的友元，则f也是B的友元



30. 类的成员函数中，下列函数中不能为虚函数的是（   ）。

 A. 构造函数  
 B. 析构函数  
 C. 运算符重载函数  
 D. const函数



31. C1是一个抽象类，下列语句中错误的是（    ）。

 A. C1 \*pC;  
 B. void fun(C1 arg);  
 C. void fun(C1 \*arg);  
 D. void fun(C1 &arg);



32. 下列关于说法中，正确的是（   ）。

 A. 类的虚函数中，不能访问this指针  
 B. 类的静态成员函数中，不能访问this指针  
 C. 类的友员函数中，不能访问this指针  
 D. 类的构造函数中，不能访问this指针



33. 有如下类定义：

class Animal {

public:

virtual void Name() = 0;

};

class Koala : public Animal {

public:

void Name() { /\* 函数体略 \*/ }

};

关于上述类定义，下列描述中错误的是（    ）。

 A. 类Koala是类Animal的派生类  
 B. 类Koala中的Name函数是一个虚函数  
 C. 类Animal中的Name函数是一个纯虚函数  
 D. 语句“Animal a;”能够建立类Animal的一个对象a



34. 关于基类中的虚函数与一般成员函数，正确的是（    ）。

 A. 虚函数只能定义一个，而一般成员函数可以多个  
 B. 虚函数与一般成员函数在基类中没什么区别  
 C. 虚函数不能定义函数体，而一般成员函数可以  
 D. 虚函数不能被调用，而一般成员函数可以



35. 基类中的虚函数如果在派生类中未重新定义，则会出现（    ）的情况。

 A. 无法使用派生类的对象调用这个函数  
 B. 不允许定义派生类的对象  
 C. 无法使用基类的对象调用这个函数  
 D. 使用派生类的对象调用的仍然是基类的这个函数



36. 关于抽象类，下列说法正确的是（    ）。

 A. 纯虚函数与虚函数的声明语法相同  
 B. 可用new操作符来生成抽象类的对象  
 C. 带有纯虚函数的类称为抽象类



37. 下面有关封装、继承、多态的描述，说法错误的是（   ）。

 A. 封装是指把客观事物封装成抽象的类，类的数据和函数只让可信的类或对象操作，不可信的则信息隐藏  
 B. 继承可以使用现有类的所有功能，并在无需重新编写原来的类的情况下对这些功能进行扩展  
 C. 隐藏是指派生类中的函数把基类中相同名字的函数屏蔽掉了  
 D. 覆盖是指不同的函数使用相同的函数名，但是函数的参数个数或类型不同



38. 关于函数模板，下列说法正确的是（    ）。

 A. 一个函数模板可以自动处理所有类型的数据  
 B. 只有能够进行函数模板中运算的类型，才可以作为类型实参  
 C. 自定义的类，不需要重载模板中得运算符，也可以作为类型实参



39. 下面（    ）派生是不可行的。

 A. 从类模版派生出类模版  
 B. 从模版类派生出类模版  
 C. 从类模版派生出普通类  
 D. 从普通类派生出类模版



40. 下面的说法中正确的是（   ）。

 A. 将一个对象放入STL中的容器里时，实际上被放入的是该对象的一个拷贝（副本）  
 B. 构造函数不可以重载  
 C. const成员函数内部不能修改成员变量的值，但是可以调用非const成员函数  
 D. 只有定义时用了virtual关键字的成员函数才是虚函数



41. 下列关于容器的end()成员函数的功能，正确的是（   ）。

 A. 判断是否为容器末尾  
 B. 返回最后一个元素的引用  
 C. 返回指向最后一个元素的迭代器  
 D. 返回指向最后一个元素下一个位置的迭代器



42. 下列关于迭代器，说法正确的是（    ）。

 A. 迭代器不直接操作容器中的数据，而是通过算法间接操作  
 B. 算法和迭代器之间不是相互独立而是互相依赖的  
 C. 迭代器是算法和容器的桥梁



43. STL的算法模板binary\_search可用于下列哪类容器（    ）。

 A. map  
 B. vector  
 C. set  
 D. list



44.以下对于map的使用中，可以正确输出15的是（   ）。

 A.



map<int, int> maps;

maps.insert(10, 15);

cout << maps.find(10)->second << endl;

 B.



map<int, int> maps;

maps.insert(pair<int, int> (10, 15));

cout << maps.find(10)->second << endl;

 C.



map<int, int> maps;

maps.insert(pair(10, 15));

cout << maps.find(10)->second << endl;

45.若要使下面程序片段能够编译通过，对于类A，以下（   ）运算符不是必须有定义的。

class A { /\* 类体略 \*/ };

A obj;

inta[] = {1, 2, 3, 4, 5};

copy(a, a + 5, obj);

 A. ++obj  
 B. --obj  
 C. \*obj  
 D. 以上三个都可以没定义



46.在已经包含头文件的情况下，下列文件操作的程序片段在语法上正确的是（   ）。

 A.



int x = 0;  
ifstream myfile;  
myfile.open("a");  
myfile << x;  
if(myfile.eof())

cout << x << ": end of file" << endl;

 B.



int x = 0;  
ifstream myfile('a');  
myfile >> x;  
if(myfile.eof())

cout << x << ": end of file" << endl;

 C.



int x = 0;  
ifstream myfile;  
myfile.Open("a");  
myfile >> x;  
if(myfile.eof())

cout << x << ": end of file" << endl;

 D.



int x = 0;  
ifstream myfile("a");  
myfile >> x;  
if(myfile.eof())

cout << x << ": end of file" << endl;

47.下列程序的输出结果是（    ）。

#include <iostream>

using namespace std;

class MyClass {

public:

MyClass(int i = 0) {

cout<< i;

}

MyClass(const MyClass &x) {

cout<< 2;

}

MyClass &operator=(const MyClass &x){

cout << 3;

return \*this;

}

~MyClass() {

cout<< 4;

}

};

intmain() {

MyClass obj1(1), obj2(2);

MyClass obj3 = obj1;

return0;

}

 A. 11214444  
 B. 11314444  
 C. 122444  
 D. 123444



48.下列程序的输出结果是（    ）。

include <iostream>

using namespace std;

class Base {

public:

void print() { cout << 'B'; }

};

class Derived : public Base {

public:

void print() { cout << 'D'; }

};

int main() {

Derived \*pd = new Derived();

Base \*pb = pd;

pb->print();

pd->print();

delete pd;

return 0;

}

 A. BB  
 B. DB  
 C. BD  
 D. DD



49. 下列程序的输出结果是（    ）。

#include <iostream>

class Base {

public:

virtual ~Base();

};

Base::~Base() {

cout<< "Base ";

}  
class Derived : public Base {

public:

    virtual ~Derived();

};  
Derived::~Derived() {

cout<< "Derived ";

}

voidfun(Base \*b) {

delete b;

}  
int main() {

Base \*b = new Derived();

fun(b);

return 0;

}

 A. Base  
 B. Derived  
 C. Derived Base



50. 下列程序的输出结果是（    ）。

class A {

public:

void FuncA() {

cout << "FuncA called\n";

}

virtual void FuncB() {

cout << "FuncB called\n";

}

};

classB : public A {

public:

void FuncA() {

A::FuncA();

cout << "FuncAB called\n";

}

virtual void FuncB() {

cout << "FuncBB called\n";

}

};

int main() {

B b;

A \*pa;

pa = &b;

A \*pa2 = new A;

pa->FuncA();

pa->FuncB();

pa2->FuncA();

pa2->FuncB();

delete pa2;

return 0;

}

 A.



FuncA called

FuncB called

FuncA called

FuncB called

 B.



FuncA called

FuncBB called

FuncA called

FuncB called

 C.



FuncA called

FuncBB called

FuncAB called

FuncBB called

 D.



FuncAB called

FuncBB called

FuncA called

FuncB called

51. 下列关于构造组合类对象时的初始化次序说法错误的是（    ）。

 A. 处理完初始化列表之后，再执行构造函数的函数体  
 B. 成员对象构造函数调用顺序：按成员对象的声明顺序，先声明者先构造  
 C. 首先对构造函数初始化列表中列出的成员进行初始化，初始化次序是成员在类体中定义的次序  
 D. 初始化列表中未出现的成员对象，不进行初始化



52. 运算符重载是对已有的运算符赋予多重含义，下列说法正确的是（   ）。

 A. 可以对基本类型（如 int 类型）的数据，重新定义“ +”运算符的含义  
 B. 可以改变已有运算符的优先级和其操作数个数  
 C. 只能重载 C++中已经有的运算符，不能定义新运算符  
 D.  C++中已经有的所有运算符都可以重载



53. 下列关于纯虚函数和抽象类的描述中，不正确的是（   ）。

 A. 纯虚函数是一个没有具体实现的虚函数  
 B. 抽象类是包括纯虚函数的类  
 C. 抽象类只能作为基类，其纯虚函数的实现在派生类中必须给出  
 D. 抽象类可以实例化



54. 假设 p1、p2 是 STL 中的 list 容器上的迭代器，那么以下语句（   ）是不符合语法的。

 A. p1++;  
 B. p1--;  
 C. p1 += 1;  
 D. p1 == p2;



55. 在公有派生类的成员函数不能直接访问基类中继承来的某个成员，则该成员一定是基类中的（   ）。

 A. 私有成员  
 B. 公有成员  
 C. 保护成员  
 D. 保护成员或私有成员



56. 假设class Y : public X，即类 Y 是类 X 的派生类，则说明一个 Y 类的对象时和删除 Y 类对象时 ，调用构造函数和析构函数的次序分别为（   ）。

 A.  X,Y; Y,X  
 B. X,Y; X,Y  
 C. Y,X; X,Y  
 D. Y,X; Y,X



57. 下列程序输出结果是（   ）。

#include <iostream>

using namespace std;

int f(int &a, int &b) {

a = 3;

b = 4;

return a + b;

}

int main() {

int a = 1;

int b = 2;

int c = f(a, a);

cout << a << b << c << endl;

return 0;

}

 A. 127  
 B. 447  
 C. 428  
 D. 427



58. 下列程序的输出结果是（   ）。

#include <iostream>

#include <vector>

using namespace std;

int main() {

vector<int> v1(1, 2);

vector<int> v2{1, 2};

cout << v1.size() << v2.size() << endl;

return 0;

}

 A. 11  
 B. 22  
 C. 12  
 D. 21



59. 下列程序输出结果是（   ）。

#include <iostream>

using namespace std;

class GeneralException {

public:

virtual void print() { cout << "G" << endl; }

};

class SpecialException : public GeneralException {

public:

void print() override { cout << "S" << endl; }

};

void f() { throw SpecialException(); }

intmain() {

try {

f();

}

catch(GeneralException e) {

e.print();

}

return 0;

}

 A. S  
 B. G  
 C. SG  
 D. GS



60. 下列说法正确的是（   ）。

#include <iostream>

using namespace std;

class A {

public:

    virtual void f() { cout <<"A" << endl; }

};

class B: public A {

private:

    void f() { cout << "B"<< endl; }

};

void g(A &a) { a.f(); }

intmain() {

    B b;

    g(b);

}

 A. 编译正确，输出 A  
 B. 编译正确，输出 B  
 C. 编译错误



1、计算机最小的存储单位是bit。

 A. 对  
 B. 错



2、命名常量（const）在定义时一定要初始化。

 A. 对  
 B. 错



3、如果a为int型变量，b为int型变量，则a/static\_cast<double>(b)的值为int型。

 A. 对  
 B. 错



4、数组可以通过值参数和引用参数两种方式传递给函数。

 A. 对  
 B. 错



5、当一个函数在运行过程中产生一个异常，则这个函数会终止，但是整个程序不一定终止运行。

 A. 对  
 B. 错



6、构造函数和析构函数都不能重载。

 A. 对  
 B. 错



7、构造函数可以声明为虚函数。

 A. 对  
 B. 错



8、已知类X成功地重载了--、=、+和[]这几个运算符，其中肯定属于成员函数的运算符是=和[]。

 A. 对  
 B. 错



9、在声明一个对象时如果使用同一类类型的另一个对象来初始化，这时调用的是该类的拷贝构造函数。

 A. 对  
 B. 错



10、C++语言中，既允许单继承，又允许多继承。

 A. 对  
 B. 错



11、派生类的继承方式有两种：公有继承和私有继承。

 A. 对  
 B. 错



12、派生类是从基类派生出来的，它不能再生成新的派生类。

 A. 对  
 B. 错



13、构造函数可以被继承，析构函数不能被继承。

 A. 对  
 B. 错



14、在公有继承中，基类中的公有成员和私有成员在派生类中都是可见的。

 A. 对  
 B. 错



15、在公有继承中，基类中只有公有成员对派生类对象是可见的。

 A. 对  
 B. 错



16、某类的友元类中的所有成员函数可以存取或修改该类的私有成员。

 A. 对  
 B. 错



17、多继承情况下，派生类的构造函数的执行顺序取决于定义派生类时所指定的各基类的顺序。

 A. 对  
 B. 错



18、虚函数和普通函数的区别在于函数名字与函数体之间的绑定方式不同，普通函数使用的是静态绑定，而虚函数使用的是动态绑定。

 A. 对  
 B. 错



19、在抽象类中定义的函数只能是纯虚函数。

 A. 对  
 B. 错



20、模板对类型参数化提供了很好的支持，类模板实例化时，编译器将根据给出的模板实参生成一个类。

 A. 对  
 B. 错



1. 编写、执行 C++程序一般需要经过的几个步骤依次是（    ）。

 A. 编辑、执行  
 B. 编辑、连接、执行  
 C. 编辑、编译、连接、执行  
 D. 编辑、连接、编译、执行



2. C++对C语言作了很多改进，即从面向过程变成为面向对象的主要原因是（   ）。

 A. 增加了一些新的运算符  
 B. 允许函数重载，并允许设置缺省参数  
 C. 规定函数说明符必须用原型  
 D. 引进了类和对象的概念，支持多态



3. C++程序属于（    ）。

 A. 汇编程序  
 B. 编译程序  
 C. 解释程序  
 D. 半编译半解释程序



4. 关于编程，以下习惯不值得推崇的是（   ）。

 A. 自己写的代码自己并不会看不懂，给自己用的代码不加注释也是可以的  
 B. 编程完成后应该对所写代码进行充分且有效的测试  
 C. 先动脑，再动笔，最后动键盘。这样的程序一般DEBUG代价更小  
 D. 尽量使用整块的时间完成一个类或一个方法的编程工作，避免打断编程时的逻辑



5. 对于一个完整的可执行C++程序，以下不可或缺的是（   ）。

 A. 至少一行#include  
 B. main 函数  
 C. 至少一个.h 文件  
 D. using namespace std;



6. 以下不属于C++关键字的是（   ）。

 A. break  
 B. number  
 C. auto  
 D. namespace



7. 下列符号中，正确的C++标识符是（   ）。

 A. enum  
 B. 2b  
 C. foo-9  
 D. \_32



8. 下列表达式的值为0的是（   ）。

 A. 3/5.0  
 B. 3%5  
 C. 3/5  
 D. 3&5



9. 下列各种运算符中，优先级最低的是（   ）。

 A. =  
 B. ||  
 C. ?:  
 D. ==



10. 若int w=1, x=2, y=3, z=4; 则下列条件表达式的值是（   ）。

w<x?w:y<z?y:z

 A. 1  
 B. 2  
 C. 3  
 D. 4



11. 设有定义语句：char c1=92, c2=92; 则以下表达式中值为真的是（   ）。

 A. !(c1^c2)  
 B. !(c1&c2)  
 C. !(~c2)  
 D. !(c1|c2)



12. 以下哪个属于算术运算符（   ）。

 A. ^  
 B. %  
 C. =  
 D. ||



13. 定义枚举类型enum ShapeType {circle = 10, square = 20, rectangle = 30, ellipse}; 那么ellipse 的值为（   ）。

 A. 40  
 B. 0  
 C. 31  
 D. 不确定



14.下列关于变量的说法中错误的是（   ）。

 A. 变量在定义的同时也可以赋予初值  
 B. 变量在使用之前需要首先声明其类型和名称  
 C. 未经初始化的变量的值是0  
 D. 变量可以有不同的类型



15. 在函数中使用 static 声明的局部变量的生存期是（   ）。

 A. 整个程序  
 B. 从函数第一次被调用到函数调用结束  
 C. 从函数第一次被调用到程序结束  
 D. 从程序开始到函数调用结束



16. 初始化一个二维数组staticint a[3][4]={{1},{0,6},{0,0,11}};

a[2][2]的值为（    ）。

 A. 1  
 B. 0  
 C. 6  
 D. 11



17. 下列给字符数组进行初始化中，正确的是（   ）。

 A. char s1[]="12345abcd";  
 B. char s2[3]="xyz";  
 C. char s3[][3];  
 D. char s4[2][3]={"xyz","mnp"};



18.下列关于指针运算的叙述中，错误的是（   ）。

 A. 两个指针，可以进行相等或不等的比较运算  
 B. 可以用空指针赋值给指针变量  
 C. 指向数组的指针，可以和整数进行加减运算  
 D. 指向同一数组的两个指针，可以进行有意义的相加运算



19. 对于“int \*pa[5];”的描述中，正确的是（   ）。

 A. pa是一个指向数组的指针，所指向的数组是5个int型元素  
 B. pa是指向数组首地址的指针  
 C. pa是一个具有 5 个元素的指针数组，每个元素是一个int型指针  
 D. 语法错误



20. 在一个源程序中，main函数的位置（   ）。

 A. 必须在最开始  
 B. 必须在系统调用的库函数的后面  
 C. 理论上可以任意  
 D. 必须在最后



21. 决定 C++语言中函数的返回值类型的是（   ）。

 A. return 语句中的表达式类型  
 B. 调用该函数时主调函数的返回类型  
 C. 调用该函数时接受返回值的变量类型  
 D. 在定义函数时指定的数据类型



22. 关于函数定义，下列错误的是（   ）。

 A. 函数的形参表可以为空  
 B. 函数定义需要有返回类型  
 C. 函数中的return语句不会结束当前函数的执行  
 D. 返回类型为void的函数可以不写return语句



23. 有函数定义：void fun(int &n) { }; 并有下列变量定义：int a=10; int b=20; const int &p=a; 则下面语句能正确执行的是（   ）。

 A. a=p;  
 B. p=b;  
 C. fun(p);  
 D. p++;



24. 一个函数功能不太复杂，但要求被频繁调用，则应把它定义为（   ）。

 A. 内联函数  
 B. 重载函数  
 C. 递归函数  
 D. 嵌套函数



25. 下列（   ）两个函数为重载函数。

 A. void a(); void b();  
 B. void a(); bool b();  
 C. void a(int x, int y); void a(int x);  
 D. void a(int x, int y); void a(int i, int j);



26.对于userRange(0.5)语句，将会调用以下（   ）函数。

 A. int userRange(int x, int y = 1) { ... }  
 B. int userRange(int x, double y) { ... }  
 C. int userRange(double x, int y) { ... }  
 D. int userRange(double x, double y = 1.0) { ... }



27. 结构和类中定义的成员默认的访问属性分别为（   ）。

 A. public, public  
 B. private, public  
 C. public, private  
 D. private, private



28、以下说法正确的是（   ）。

 A. 在静态成员函数中可以调用同类的其他任何成员函数   
 B. 在静态成员函数中不能使用this指针  
 C. 静态成员变量每个对象有各自的一份  
 D. const成员函数不能作用于非const对象



29. 通常的拷贝（复制）构造函数的参数是（   ）。

 A. 对象名  
 B. 对象的成员名  
 C. 本类对象的引用名  
 D. 本类对象的指针名



30. 已知变量v是类Vehicle的一个实例，并有函数原型声明：void run(Vehicle p); 则如下各组语句中执行时不会调用拷贝（复制）构造函数的是（   ）。

 A. Vehicle car(v);  
 B. Vehicle car = v;  
 C. run(v);  
 D. Vehicle car; car = v;



31. 下列关于友元的说法，错误的是（   ）。

 A. 如果要将类 A 中的成员函数 void A::f()声明为类 B 的友元，应在类A 中将 f 定义为 friend void B::f()  
 B. 友元函数可以访问该类的私有成员  
 C. 类间友元关系不可以被继承  
 D. 从一定意义上讲，友元是对数据隐蔽和封装性的破坏



32.当派生类从基类公有继承时，以下（    ）说法是错误的。

 A. 可以把派生类对象的地址赋给基类对象的指针  
 B. 可以用基类对象为派生类对象赋值  
 C. 可以用派生类对象为基类对象赋值  
 D. 可以把指向派生类对象的指针赋给基类对象的指针



33.下列对派生类的描述中，错误的是（    ）。

 A. 派生类成员除了它自己的成员外，还包含它的基类成员  
 B. 派生类至少有一个基类  
 C. 一个派生类可作为另一个派生类的基类  
 D. 派生类中继承的基类成员的访问权限到派生类保持不变



34. 虚函数声明时需要关键字（   ）。

 A. static  
 B. friend  
 C. virtual  
 D. abstract



35.基类中允许定义（    ）个虚函数？

 A. 任意多个  
 B. 一个  
 C. 两个  
 D. 和成员变量的个数有关



36.关于基类中的虚函数与一般成员函数，正确的是（    ）。

 A. 虚函数不能定义函数体，而一般成员函数可以  
 B. 虚函数与一般成员函数在基类中没什么区别  
 C. 虚函数只能定义一个，而一般成员函数可以多个  
 D. 虚函数不能被调用，而一般成员函数可以



37.虚函数不得声明为（    ）函数。

 A. 其他三项均正确  
 B. 静态  
 C. 友元  
 D. 构造



38.只有使用虚函数机制，才能（    ）。

 A. 使用派生类指针调用基类的成员函数  
 B. 使派生类指针指向基类对象  
 C. 使用基类指针调用派生类的成员函数  
 D. 使基类指针指向派生类对象



39.基类中的虚函数如果在派生类中未重新定义，则会出现（    ）的情况。

 A. 不允许定义派生类的对象  
 B. 使用派生类的对象调用的仍然是基类的这个函数  
 C. 无法使用派生类的对象调用这个函数  
 D. 无法使用基类的对象调用这个函数



40.如果一个类至少有一个（    ），那么就称该类为抽象类。

 A. 构造函数  
 B. 友元函数  
 C. 成员函数  
 D. 纯虚函数



41. 以下程序执行完时，a 和 b 的值变为（   ）。

void f(){

int a = 7, b = 5;

if(a != 2 \* b + 1)

a = 10;

b = 6;

else if(a == 10 && b == 6)

b = 5;

}

 A. 10 5  
 B. 7 5  
 C. 10 6  
 D. 编译错误//else without a previous if



42. 以下程序的输出结果是（   ）。

void f(){

int a = 5, b = 5;

switch(a) {

case 5: cout << "A";

case 7: cout << "C"; break;

case 4: cout << "B";

case 5: cout << "D";

case b: cout << "E";

}

}

 A. ACDE  
 B. AD  
 C. 编译错误  
 D. A



43. 已知：1+2+...+100=5050。以下程序的输出结果是（   ）。

void f() {

int a = 0, b = 0;

while(a < 100) {

++a;

b += a;

}

cout << b << endl;

}

 A. 4950  
 B. 5050  
 C. 5151  
 D. 编译错误



44. 以下程序的输出结果是（   ）。

void f() {

char str[11];

for(int i = 0; i < 10; ++i)

str[i] = 'a' + i;

str[i] = '\0';

cout << str << endl;

}

 A. 可运行，但存在危险操作（数组越界）  
 B. abcdefghij  
 C. abcdefghij（后跟一定长度乱码）  
 D. 编译错误



45. 以下程序的输出结果是（   ）。

void f() {

enum Gameresult { Win, Lose, Tie, Cancel };

Gameresult result;

for(int count = Win; count <= Cancel; ++count)

if(count == Lose)

cout << count << endl;

return;

}

 A. Cancel  
 B.  Lose  
 C. 1  
 D. 编译错误



46. 以下程序的执行结果是（   ）。

#include <iostream>

using namespace std;

void f1(int a) { a++; }

void f2(int &a) { a++; f1(a); }

int main() {

int a = 3;

f2(a);

cout << a << endl;

return 0;

}

 A. 3  
 B. 4  
 C. 5  
 D. 6



47. 以下程序的执行结果是（   ）。

#include <iostream>

using namespace std;

void f2(int &a);

void f1(int &a) {

if(a <= 0)

++a;

f2(a);

}

void f2(int &a) {

if(a > 0)

--a;

f1(a);

}

int main() {

int a = 0;

f1(a);

cout << a << endl;  
return 0;

}

 A. 运行时错误：交叉调用无法停止，最终导致函数栈溢出  
 B. 0  
 C. 1  
 D.  编译错误



48. 以下程序中，search 函数一共被调用了（   ）次。

#include <iostream>

using namespace std;

int search(int a[], int start, int end, int target) {

if(start > end)

return -1;

if(a[start] != target)

return search(a, start + 1, end, target);

return start;

}

int main() {

int a[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

int index = search(a, 0, 9, 5);

}

 A. 4  
 B. 5  
 C. 6  
 D. 11



49.以下程序输入 0345，结果为（   ）。

#include <iostream>

using namespace std;

int main() {

int sum = 0;

char next = '0';

do {

sum += next - '0';

cin >> next;

} while(!next);

cout << sum << endl;

return 0;

}

 A. 345  
 B. 12  
 C. 3  
 D. 0



50. 以下程序的编译运行情况是（   ）。

#include <iostream>

using namespace std;

int sum(int, int);

int main() {

int a = 100, b = 200, c = 0;

c = sum(a, b);

cout << c << endl;

return 0;

}

int sum(int x, int y) {

return x + y;

}

 A. 编译通过，300  
 B. 编译通过，0  
 C. 编译出错  
 D. 编译通过，无输出



51. 下列程序的运行结果是（   ）。

#include <iostream>

using namespace std;

int main() {

const float PI = 0;

PI = 1;

cout << PI << endl;

return 0;

}

 A. 0  
 B. 1  
 C. 随机值  
 D. 编译错误



52. 以下程序输出结果为（   ）。

#include <iostream>

using namespace std;

int main() {

int n = 3;

cout << (n = 4) << " ";

cout << (n | 2) << " ";

cout << (n & 4) << " ";

return 0;

}

 A. 4 6 1  
 B. 4 6 4  
 C. 1 1 4  
 D. 4 1 4



53. 对于以下程序，输出为（   ）。

#include <iostream>

using namespace std;

namespace ns {

int a = 1;

namespace nns {

int a = 2;

}

}

int a =3;

int main() {

int a = 4;

a += 1;

{

int a = 4;

a += 1;

cout << a << ++ns::nns::a << ns::a << endl;

}

return 0;

}

 A. 531  
 B. 631  
 C. 532  
 D. 编译错误，存在未找到的变量



54.下列程序的输出结果是（    ）。

include <iostream>

using namespace std;

class Base {

public:

   void print() { cout << 'B'; }

};

class Derived : public Base {

public:

   void print() { cout << 'D'; }

};

int main() {

Derived \*pd = new Derived( );

Base \*pb = pd;

pb->print( );

pd->print( );

delete pd;

return 0;

}

 A. DB  
 B. BB  
 C. BD  
 D. DD



55、关于常量成员和静态成员，下面说法正确的是（   ）。

 A. 公有的静态成员在没有对象生成的时候也能被直接访问  
 B. 从同一个类模板得到的两个模板类可以共享同一个静态成员  
 C. 常量成员函数只有常量对象可以调用  
 D. 在静态成员函数中不能访问非静态成员，但可以访问常量成员



56、下列（   ）可作为函数模板和类模板的共同特点？

 A. 函数作参数  
 B. 函数类型做参数  
 C. 数据作参数  
 D. 数据类型作参数



57、下列（   ）容器不属于序列（顺序）容器？

 A. vector  
 B. deque  
 C. set  
 D. list



58、设v是一个vector容器，则v.size()的功能是（   ）。

 A. 返回向量v能存储的元素个数   
 B. 返回向量v的最小元素  
 C. 返回向量v的实际大小（元素个数）  
 D. 返回向量v的最大元素



59、下列关于容器的end()成员函数的功能，下列（    ）是正确的？

 A. 返回指向最后一个元素下一个位置的迭代器  
 B. 返回最后一个元素的引用  
 C. 返回指向最后一个元素的迭代器  
 D. 判断是否为容器末尾



60、查找算法find(first, last, val)的参数中，first和last表示的范围用下列（   ）区间表示比较合适？

 A. (first, last]  
 B. [first, last)  
 C. (first, last)  
 D. [first, last]



对一个整数n，如果其各个位数的数字相加得到的数m能整除n，则称n为自整除数。例如21，21%(2+1)==0，所以21是自整除数。现求出从10到n（n < 100）之间的所有自整除数。

1. #include <iostream>
2. using namespace std;
3. int main(){
4. int n;
5. cin >> n;
6. for(int i = 10; i <= n; i++)
7. if(i % (i / 10 + i % 10) == 0)
8. cout << i << endl;
9. return 0;
10. }

下面是一个图书的单价表：  
《计算概论》28.9元/本、《数据结构与算法》32.7元/本、《数字逻辑》45.6元/本、《C++程序设计教程》78元/本、《人工智能》35元/本、《计算机体系结构》86.2元/本、《编译原理》27.8元/本、《操作系统》43元/本、《计算机网络》56元/本、《Java程序设计》65元/本。  
给定每种图书购买的数量，计算应付的总费用。

1. #include<iostream>
2. #include<iomanip>
3. using namespace std;
4. int main(){
5. // 多组数据
6. int numGroup = 0;
7. cin >> numGroup;
9. // 书的数量
10. int numBook = 10;
12. // price为每本书的价格
13. double price[] = {28.9, 32.7, 45.6, 78, 35, 86.2, 27.8, 43, 56, 65};
15. // 处理多组输入
16. for (int iGroup = 0; iGroup < numGroup; iGroup++){
18. // sum为总价
19. double sum = 0;
20. // 处理每一本书
21. for (int iBook = 0; iBook < numBook; iBook++){
22. int num;
23. cin >> num;
24. sum += num \* price[iBook];
25. }
26. // 按要求输出
27. cout << fixed << setprecision(2) << sum << endl;
28. }
30. return 0;
31. }

某个科室的病房分为重症和普通，只有当病人的疾病严重程度超过了入住重症病房的最低严重值，才可以安排入住重症病房。

现在要求设计一个程序，给病人安排好病房。疾病的严重程度用0到10来表示，0表示小毛病，10表示非常严重。

#include<iostream>  
#include<stdio.h>  
using namespace std;  
int main()  
{  
int m;  
float a;  
cin >> m >> a;  
int num[50];  
float deg[50];  
for(int i=0; i<m; i++)  
scanf("%d %f", &(num[i]), &(deg[i]));  
if(m==1)  
{  
if(deg[0]>a)  
printf("%03d %.1f\n",num[0],deg[0]);  
else  
cout << "None." << endl;  
}  
else  
{  
int count=0;  
for(int i=0; i<m; i++)  
{  
for(int j=0; j<m-i-1; j++)  
{  
if(deg[j]>deg[j+1])  
{  
float temp = deg[j];  
deg[j] = deg[j+1];  
deg[j+1] = temp;  
  
  
int temp\_n = num[j];  
num[j] = num[j+1];  
num[j+1] = temp\_n;  
}  
}  
count++;  
if(deg[m-i-1]<=a)  
break;  
}  
if(count==1)  
cout << "None." << endl;  
else  
for(int i=0; i<count-1; i++)  
printf("%03d %.1f\n",num[m-i-1],deg[m-i-1]);  
}  
return 0;  
}