

上海海事大学试卷

2023 — 2024 学年第一学期期末考试

《 面向对象程序设计 》(A 卷) 考试形式(闭) 卷

班级	学号				姓名		总分	
题 目	一	二	三	四				
得 分								
阅卷人								

考生诚信考试承诺书

我承诺：自觉遵守上海海事大学考场规则，服从监考人员的监督管理，不做违背考试纪律的任何事情，做到诚信考试。如有违反任何考试规定，自愿接受上海海事大学有关条款处理（直至开除）。

考生签名：_____

日期：_____年____月____日

-----从此处开始，可以直接命题-----

一、选择题（共 10 题，每题 2 分，共 20 分）请将正确答案写在答题纸上。

1. 类的实例化是指（ ）。
A. 定义类 B. 创建类的对象 C. 指明具体类 D. 调用类的成员
2. 类的构造函数被自动调用执行的情况是在创建该类的（ ）。
A. 成员函数时 B. 数据成员时 C. 对象时 D. 友元函数时
3. 在下列函数原型中，可以作为类 AA 构造函数的是（ ）。
A. void AA(int); B. int AA(); C. AA(int) const; D. AA(int);
4. 不属于成员函数的是（ ）。
A. 静态成员函数 B. 友元函数 C. 构造函数 D. 析构函数
5. 下列函数中，不能声明为虚函数的是（ ）。
A. 私有成员函数 B. 公有成员函数 C. 构造函数 D. 析构函数
6. 当一个类的某个函数被声明为 virtual 时，该函数在该类的所有派生类中（ ）。
A. 都是虚函数 B. 只有被重新声明时才是虚函数
C. 只有被重新声明为 virtual 时才是虚函数 D. 都不是虚函数
7. 下列关于 C++ 运算符函数的返回类型的描述中，错误的是（ ）。

- A. 可以是类类型 B. 可以是 int 类型
C. 可以是 void 类型 D. 可以是 float 类型
8. 下列运算符不能用友元函数重载的是 ()。
A. + B. = C. * D. <<
9. 下列运算符中, () 运算符在 C++ 中不能被重载。
A. ?: B. [] C. new D. &&
10. 类 B 是类 A 的公有派生类, 类 A 和类 B 中都定义了虚函数 func(), p 是个指向类 A 对象的指针, 则 p->A::func() 将 ()。
A. 调用类 A 中的 func() 函数
B. 调用类 B 中的 func() 函数
C. 根据 p 所指的对象的类型而确定调用类 A 或类 B 中的 func() 函数
D. 既调用类 A 中的 func() 函数, 也调用类 B 中的 func() 函数

二、填空题 (共 10 题, 每题 2 分, 共 20 分) 请将正确答案写在答题纸上。

- 对于任意一个类, 析构函数的个数最多为_____个。
- 假定 Dc 是一个类, 则执行 “Dc a[10], b(2)” 语句时, 系统自动调用该类构造函数的次数为_____。
- _____运算符通常用于实现释放该类对象中指针成员所指向的动态存储空间的任务。
- 若多个基类及其派生类中都定义了同名函数成员, 则要访问相应函数时, 就需要在函数名前加上_____和_____运算符。
- 若要保证一个公共基类在派生类中只产生一个基类子对象, 则必须都以_____的方式继承它。
- 保护继承时, 基类中的所有非私有成员在派生类中是_____成员。
- 当创建一个派生类对象时, 先调用基类的构造函数, 然后调用_____的构造函数, 最后调用派生类的构造函数。
- 对于基类数据成员的初始化必须在派生类构造函数中的_____处执行。
- 静态联编在_____阶段完成, 动态联编在_____阶段完成。
- 如果派生类中没有给出纯虚函数的具体实现, 这个派生类仍然是一个_____。

三、看程序, 写结果 (共 4 题, 每题 5 分, 共 20 分) 请将正确答案写在答题纸上。

1. 程序如下:

```
#include<iostream>
using namespace std;
class A{
public:
```

```

int n;
};
class B:public A{};
class C:public A{};
class D:public B,public C
{public:int getn(){return B::n;}
};
int main()
{
D d;
d.B::n=10;
d.C::n=20;
cout<<d.B::n<<endl;
cout<<d.getn()<<endl;
return 0;
}

```

运行结果：

2. 程序如下：

```

#include<iostream>
using namespace std;
class A
{
public:
A(int i, int j)
{a=i;b=j;}
void Move(int x,int y){a+=x;b+=y;}
void Show(){cout<<"("<<a<<","<<b<<")"<<endl;}
private:
int a,b;
};
class B:private A
{

```

```

public:
B(int i,int j,int k,int l):A(i,j){x=k;y=l;}
void Show(){cout<<x<<" "<<y<<endl;}
void fun(){Move(3,5);}
void f1(){A::Show();}
private:
int x,y;
};
int main( )
{
A e(1,2);
e.Show();
B d(3,4,5,6);
d.fun();
d.Show();
d.f1();
return 0;
}
运行结果:

```

3. 程序如下:

```

#include <iostream>
using namespace std;
class Bclass{
public:
Bclass(int i, int j){x=i;y=j;}
virtual int fun(){return 0;}
protected:
int x,y;
};
class Iclass:public Bclass{
public:
Iclass(int i, int j, int k):Bclass(i, j){z=k;}

```

```

int fun(){return (x+y+z)/3;}
private:
int z;
};
int main()
{
Iclass obj(2, 4, 10);
Bclass p1 = obj;
cout<<p1.fun()<<endl;
Bclass &p2= obj;
cout<<p2.fun()<<endl;
cout<<p2.Bclass::fun()<<endl;
Bclass *p3 = &obj;
cout<<p3->fun()<<endl;
return 0;
}

```

运行结果：

4. 程序如下：

```

#include <iostream>
using namespace std;
class A{
public:
    virtual ~A( ){cout<<"A"<<endl;}
};
class B:public A{
char *buf;
public:
B(int i){buf=new char[i];}
virtual ~B()
{
delete []buf;
cout<<"B"<<endl;
}
}

```

```

}
};
void fun(A *a)
{
delete a;
}
int main( )
{
A *a=new B(10);
fun(a);
return 0;
}

```

运行结果：

四、编程题（共 4 题，每题 10 分，共 40 分）请将正确答案写在答题纸上。

1. 定义一个 Dog 类，用一个静态数据成员 countofdogs 来记录 Dog 类对象的个数，静态成员函数 getCount()用来获取 Dog 类对象的个数。主函数中自拟数据统计 Dog 类对象的个数。
2. 定义平面直角坐标系上的一个点的类 CPoint 作为基类，派生出描述一条直线的类 CLine，再派生出一个矩形类 CRect。要求成员函数能求出两点间的距离，矩形的周长和面积。主函数中自拟数据计算线段的长度以及矩形的周长和面积。
3. 定义一个描述圆柱体的类 CCylinder，定义圆柱体的底面半径 radius 与高 high 为私有数据成员。用公有成员函数 Volume 计算圆柱体的体积，公有成员函数 Show 显示圆柱体的半径，高和体积。在主函数中用 new 运算符动态建立圆柱体的对象。初值为 (10, 10)。然后调用 Show 显示圆柱体的半径，高，体积。最后用 delete 运算符收回圆柱体动态分配的空间。
4. 设计一个函数模板，实现对任意数据类型的数组进行升序排序。主函数中自拟数据并输出排序后的结果。

答题纸

班级 _____ 学号 _____ 姓名 _____

一、选择题（共 10 题，每题 2 分，共 20 分）

题号	1	2	3	4	5	6	7	8	9	10
答案										

二、填空题（共 10 题，每题 2 分，共 20 分）

题号	答案
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

三、看程序，写结果（共 4 题，每题 5 分，共 20 分）

1.

2.

3.

4.

四、编程题（共 4 题，每题 10 分，共 40 分）