

复旦大学计算机科学技术学院

2021~2022 学年第二学期期末模拟考试试卷

课程名称：____面向对象程序设计____ 课程代码：____COMP130135.03____

开课院系：____计算机科学技术学院____ 考试形式：线上考试（闭卷）

姓名：____ 学号：____ 专业：____

（本试卷答卷时间为 120 分钟，答案必须写在答题纸上，做在试卷上无效）

提示：请同学们秉持诚实守信宗旨，谨守考试纪律，摒弃考试作弊。学生如有违反学校考试纪律的行为，学校将按《复旦大学学生纪律处分条例》规定予以严肃处理。

题 号	一	二	三	四	总 分
得 分	24	20	24	32	100

一、选择题（每题只选择一个答案；如选多个，不计分。每题 3 分，总分 24 分）

- 关于 `list` 和 `vector`，下列说法中错误的是_____。
 - `list` 顺序访问比 `vector` 稍慢些；
 - `vector` 不支持在中间某位置插入元素的操作；
 - 对于大型数据，`list` 删除和插入要比 `vector` 快得多；
 - `vector` 支持位置索引，而 `list` 不支持。
- 关于构造函数，下列说法错误的是_____。
 - 构造函数名字和类名相同；
 - 构造函数在创建对象时自动执行；
 - 构造函数无任何函数返回类型；
 - 构造函数有且只有一个。
- C++派生类仅可以访问其基类的_____。
 - 公有成员
 - 保护成员
 - 私有成员
 - 公有和保护成员
- 下列关于运算符重载的描述中，错误的是_____。
 - 运算符重载不可以改变操作数的个数；
 - 运算符重载不可以改变运算符的功能；
 - 运算符重载不可以改变结合方向；
 - 运算符重载不可以改变运算优先级。
- 下列关于类的描述中，错误的是_____。
 - 一般来说，类可以控制对象在创建、复制、赋值和销毁时的所有行为；
 - 如果一个类没有定义构造函数，编译系统就会合成默认构造函数；
 - 在构造函数中分配资源的类，几乎都必须定义复制构造函数、赋值操作符和析构函数；
 - 如果这个类没有明确地定义复制构造函数、赋值操作符以及析构函数，编译器不会自动合成这些操作。

6、定义析构函数时，应该注意_____。

- A) 无形参，也不可重载;
- B) 返回类型是 `void` 类型;
- C) 其名与类名完全相同;
- D) 函数体中必须有 `delete` 语句。

7、下列关于虚函数和动态绑定的说法，错误的是_____。

- A) 在主函数中通过基类对象的引用或指针调用虚函数时，发生动态绑定;
- B) 调用以基类对象的引用做形参的函数时，传入一个派生类对象，发生静态绑定;
- C) C++是通过虚函数的动态绑定特性来支持多态的;
- D) 派生类不能再定义新的虚函数。

8、执行下面的程序将输出_____。

```
#include <iostream>
class BASE{
    char b;
public:
    BASE(char n):b(n){}
    virtual ~BASE(){std::cout<<b;}
};
class DERIVED:public BASE{
    char d;
public:
    DERIVED(char n):BASE(n+1),d(n){}
    ~DERIVED(){std::cout<<d;}
};
int main(){
    DERIVED a('X');
    return 0;
}
```

- A) X B) Y C) XY D) YX

二、程序阅读题（每题 5 分，共 20 分。必要的头文件和 using 语句已经略去。）

1、以下程序的运行结果是：_____

<pre>class Str{ string s; static int Count; public: Str(string str=string()):s(str) { Count++; } void extend(string str){s += str;} string getStr()const{return s;} static int getCount()const { return Count; } };</pre>	<pre>int Str::Count = 0; int main() { Str str[3], a("0"); for(int i = 0; i < 3; i++) str[i].extend("1"); cout << a.getCount() << " " << a.getStr() << endl; return 0; }</pre>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

2、以下程序的运行结果是：_____

<pre> class B{ public: B(int a0=0, int b0=0):a(a0), b(b0){} int AR()const{return a*b;} bool operator < (const B& b) { return AR() < b.AR(); } bool operator == (const B& b) { return AR() == b.AR(); } bool operator == (int n) { return AR() == n; } private: int a, b; }; </pre>	<pre> template <typename T> T* bs(T* a, int n, const T& k){ int b = 0, e = n-1, m; while(b <= e){ m = (b+e)/2; if(a[m] == k) break; else if (a[m] < k) b = m+1; else e = m-1; } return b <= e ? &a[m] : NULL; } int main() { B b[] = {B(1,2), B(3,4), B(5,6)}; B* k = bs(b, 3, B(2,6)); if(k != NULL) cout << k->AR() << endl; else cout << "Can not find." << endl; return 0; } </pre>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

3、以下程序的运行结果是：_____

<pre> class Counter{ public: int cnt; Counter():cnt(0){} }; class Person{ public: Person(){ counter = new Counter(); } void addCnt(int n){counter->cnt += n;} int getCounter(){return counter->cnt;} private: Counter* counter; }; </pre>	<pre> int main() { Person p1; Person p2 = p1; p1.addCnt(1); p2.addCnt(3); cout << p1.getCounter()+p2.getCounter() << endl; return 0; } </pre>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

4、以下程序的运行结果是：_____

<pre> class Human{ public: virtual void disp(){cout << "Human display." << endl;} ~Human(){cout << "Human over." << endl;} }; class Man:public Human{ public: void disp(){cout << "Man display." << endl;} ~Man(){cout << "Man over." << endl;} }; </pre>	<pre> int main() { Human* p = new Man(); p->disp(); delete p; return 0; } </pre>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------

三、程序填空题（每空 3 分，共 24 分）

下面是选择排序的代码，其核心是每次从没有排序的元素中，挑选出最小的一个，然后将其放在正确的位置上。经过选择排序之后，数组的元素按照从小到大的顺序排列。

```
template <typename T>
void swap(_____①_____)
{
    T c(a); a = b; b = c;
}
template <typename For>
void selectSort(For begin, For end)
{
    For it, minv;
    while (_____②_____) {
        for (_____③_____; it != end; it++)
            if (*it < *minv)
                minv = it;
        swap(*begin, *minv);
        begin++;
    }
}
```

四、编程题（2 题，共 32 分）

1、实现一个可以表示任意范围的正整数类。要求：

- 1) 正整数类的内部数据由标准库容器保存；
- 2) 正整数类支持加法、乘法运算；
- 3) 正整数类支持输入、输出操作。

测试代码如下，请给出 Integer.h 和 Integer.cpp 的代码。

```
#include "Integer.h"
using std::cout; using std::endl;
using std::cin;
void testInteger() {
    Integer a, b, c, d, e;
    cin >> a >> b >> c;
    d = a + b + c;
    e = a * b * c;
    cout << "d: " << d << " e: " << e << endl;
}
int main()
{
    testInteger();
    return 0;
}
```

程序运行时，若输入：

123456789

987654321

13579

则程序输出：

d: 1111124689 e: 1655723197878474317751

（装订线内不要答题）