

## 一、单项选择题（每题 2 分，共 40 分）

- \_\_\_\_\_不是属于面向对象程序设计的特性  
A. 抽象性                      B. 数据相关性                      C. 多态性                      D. 继承性
- 将对某一类数据的处理算法应用到另一类数据的处理中,要用到 C++的\_\_\_\_\_  
A. 类                              B. 虚函数                              C. 运算符重载                      D. 模板
- C++与 C 语言最根本的不同之处在于\_\_\_\_\_  
A. 使用了类                      B. 能够实现变量自动初始化                      C. 支持软件重用                      D. 支持接口重用
- 动态内存分配的主要目的是\_\_\_\_\_  
A. 使程序按动态联编方式运行                      B. 正确合理的使用内存  
C. 提高程序的运行速度                      D. 提高程序的可维护性
- 在 C++函数的形参前加 const 关键字,是为了提高函数的\_\_\_\_\_  
A. 数据封装性                      B. 可理解性                              C. 可维护性                              D. 可重用性
- 函数重载的目的是\_\_\_\_\_  
A. 实现共享                      B. 使用方便,提高可读性                      C. 提高速度                              D. 减少空间
- 从程序片断: `char name[] = "C++"; course(name);`可判断函数 `course` 的调用采用的是\_\_\_\_\_  
A. 传值调用                      B. 带缺省参数值的函数调用                      C. 引用调用                              D. 传址调用
- 用来说明类中公有成员的关键字是\_\_\_\_\_  
A. public                              B. private                              C. protected                              D. friend
- 如果一个类的成员函数 `print()` 不修改类的数据成员值,则应将其声明为  
A. `void print() const;`                              B. `const void print();`  
C. `void const print();`                              D. `void print(const);`
- 下列关于构造函数的论述中,不正确的是\_\_\_\_\_  
A. 构造函数的函数名与类名相同                              B. 构造函数可以设置默认参数  
C. 构造函数的返回类型缺省为 `int` 型                              D. 构造函数可以重载
- 在程序代码: `A::A(int a, int *b) { this->x = a; this->y = b; }`中, `this` 的类型是\_\_\_\_\_  
A. `int`                              B. `int *`                              C. `A`                                      D. `A *`
- 内存泄漏是指\_\_\_\_\_  
A. 内存中的数据出现丢失                              B. 试图释放一个已经释放了的动态分配的堆内存  
C. 函数中局部变量所占的栈内存没有及时回收                              D. 动态分配的堆内存存在程序退出后始终被占用
- 从程序片断: `student zhangsan("张三","M",22); zhangsan.id("2005131000");`可判断 `id` 是一个\_\_\_\_\_  
A. 私有成员数据                      B. 私有成员函数                              C. 公有成员数据                              D. 公有成员函数
- 友元函数\_\_\_\_\_  
A. 可以被声明为 `const`                              B. 没有 `this` 指针  
C. 可以用类名或对象名来调用                              D. 只能用对象名来调用

- 二、下面的程序是关于 CBook 类的一个定义，试改正程序中的错误。（共 10 分）

-2-

```
}
```

### 三、根据如下所示程序，回答下列问题（共 10 分）

```
#include <iostream.h>
class CJournal
{
public:
    CJournal() { cout << "Journal default constructor" << endl; }
    virtual void subscribe() = 0;
    void read() { cout << "Read paper" << endl; }
    ~CJournal() { cout << "Journal default destructor" << endl; }
};

class CComputerDesign : public CJournal
{
public:
    CComputerDesign () {cout << "《Computer Design》 default constructor" << endl; }
    virtual void subscribe() { cout << "Subscribing 《Computer Design》" << endl; }
    void read() {cout << "Reading 《Computer Design》" << endl; }
    ~CComputerDesign() { cout << "《Computer Design》 default destructor" << endl; }
};

void main()
{
    CComputerDesign journal1;
    CJournal *p_journal;
    journal1.subscribe();
    journal1.read();
    // ①
    p_journal = &journal1;
    p_journal->subscribe();
    p_journal->read();
}
```

1. 当程序运行到①处时，写出程序运行的输出结果（3 分）
2. 当程序结束时，程序会在第 1 问的基础上增加哪些输出。（4 分）
3. 若在主函数中定义一个对象 CJournal journal; 程序编译时会否出错？为什么？（3 分）

### 四、编程题(共 40 分)

1. 定义一个商品类 CGoods，其中包含商品号(long no)、商品名(char \*p\_name)、商品价格(double price)三个数据成员，以及相应的构造函数、拷贝构造函数、析构函数、打印数据成员的成员函数。（10 分）

```
#include <iostream>
#include <string.h>
```

```
using namespace std;
class CCourse
{
private:
    long no;
    char *p_name;
    float credit;
public:
    CCourse(long no_val, char p_val, float credit_val);
    CCourse(const CCourse &r_course);
    ~CCourse() {delete p_name;}
    void print() const;
};

CCourse::CCourse(long no_val, char p_val, float credit_val)
{
    no=no_val;
    p_name=new char[strlen(p_val)+1];
    strcpy(p_name, p_val);
    credit=credit_val;
}

CCourse::CCourse(const CCourse &r_course)
{
    no=r_course.no;
    p_name=new char[strlen(r_course.p_name)+1];
    strcpy(p_name, r_course.p_name);
    credit=r_course.credit;
}

void CCourse::print() const
{
    cout<<"Course number"<<no<<endl;
    cout<<"Course name"<<p_name<<endl;
    cout<<"Course credit"<<credit<<endl;
}
```

2. 为 CGoods 类增加一个商品总数(int count)数据成员, 并增加一个成员函数 getCount() 获取 count 的值, 编写一个友元函数 getName() 获取商品名称 p\_name。做如上修改后, 重新实现 CGoods 类(与第 1 问相同的不用再重复)。(10 分)

3. 为 CGoods 类定义小于运算符 (‘<’) 和不小于运算符 (‘>=’) 两个运算符重载函数。CGoods 类对象大小的比较是根据其商品价格 (price) 的值的实现。 (与第 2 问相同的不用再重复) (10 分)

4. 以 CGoods 类为基类，派生出服装类 CClothes 和食品类 CFood 两个派生类，并在这两个类中分别增加一个表示品牌的指针数据成员 (char \*p\_brand) 和表示用途的成员函数 (void usedFor()) ——可分别输出一条表示服装和食品用途的信息)。写出 CClothes 类和 CFood 类的完整定义 (包括构造、析构和 usedFor() 成员函数的实现)。 (10 分)

五、附加题 (共 30 分。注意：确保基本题成绩能达到 75 分以上，再尝试做附加题！)

1. 在基本题第四题的基础上，在 CGoods 类增加总商品数 (long total\_goods) 和商品总价格 (double total\_price) 两个数据成员，以及相应的获取这两个数据成员值的成员函数 getTotalGoods() 和 getTotalPrice()。(注意说明数据成员和成员函数的存储类型，以便能够用类名来调用 getTotalGoods() 和 getTotalPrice() 这两个函数)。为了能够采用动态联编的方式调用派生类的 usedFor() 成员函数，应该在 CGoods 类及其派生类 CClothes 和 CFood 类中作何改动？ (15 分)

2. 编写一个实现两个数交换的函数模板 swap，然后使用该函数模板再编写一个对具有 n 个数组元素（通用类型）的数组采用冒泡排序算法进行排序的函数模板。（15 分）

## 一、单项选择题

1-5. BDABC 6-10. BDAAC

11-15. CADBB 16-20. CADDA

(每小题答对 2 分，不答 0 分，答错 0 分)

## 二、

#include &lt;math.h&gt; 改为#include &lt;string.h&gt;

p\_book = new char[strlen(p\_val)]改为 p\_book = new char[strlen(p\_val)+1]

CBook abook;改为 CBook abook(p);

p\_book\_obj = abook 改为 p\_book\_obj = &amp;abook;

p\_book\_obj.print() 改为 p\_book\_obj-&gt;print();

(该对得 2 分，未改或改错得 0 分，正确之处该成错误倒扣 1 分)

## 三、

1. Journal default constructor

&lt;&lt;Computer Design&gt;&gt; default constructor

Subscribing &lt;&lt;Computer Design&gt;&gt;

Reading &lt;&lt;Computer Design&gt;&gt;

(全部答对得 3 分，答错或漏答一条输出扣 1 分)

2. Subscribing &lt;&lt;Computer Design&gt;&gt;

Reading paper

&lt;&lt;Computer Design&gt;&gt; default destructor

Journal default destructor

(全部答对得 4 分，答错或漏答一条输出扣 1 分)

3. 会出错，因为 CJournal 中包含有纯虚函数，故 CJournal 是抽象类，不能定义抽象类对象。

(答对编译会出错得 1 分，答对原因得 2 分。)

## 四、

1. #include &lt;iostream&gt;

#include &lt;string.h&gt;

using namespace std;

class CCourse

{

private:

long no;

char \*p\_name;

float credit;

public:

CCourse(long no\_val, char \*p\_val, float credit\_val);

CCourse(const CCourse &amp;r\_course);

~CCourse() { delete p\_name; }

void print() const;

};

CCourse::CCourse(long no\_val, char \*p\_val, float credit\_val)

{

no = no\_val;

p\_name = new char[strlen(p\_val)+1];

strcpy(p\_name, p\_val);

```

        credit = credit_val;
    }
    CCourse::CCourse(const CCourse &r_course)
    {
        no = r_course.no;
        p_name = new char[strlen(r_course.p_name)+1];
        strcpy(p_name, r_course.p_name);
        credit = r_course.credit;
    }
    void CCourse::print() const
    {
        cout << "Course number: " << no << endl;
        cout << "Course name: " << p_name << endl;
        cout << "Course credit: " << credit << endl;
    }

```

(数据成员定义正确得 2 分，部分正确得 1 分，不正确得 0 分)

每个成员函数定义正确得 2 分，每个成员函数有小错误扣 1 分，完全不正确不得分)

2. 在 class CCourse 定义中增加一条：

```
private:
```

```
    static int total_course;
```

(答对得 1 分，未加 static 得 0 分)

在类外部增加一条：

```
int CCourse::total_course = 0;
```

(答对得 1 分，答错或漏答得 0 分)

在 CCourse 类的构造函数中增加一条：

```
total_course++; (答对得 1 分)
```

在 CCourse 类的拷贝构造函数中增加一条：

```
total_course++; (答对得 1 分)
```

在 CCourse 类的析构函数中增加一条：

```
total_course--; (答对得 1 分)
```

在 class CCourse 定义中增加一条：

```
public:
```

```
static getTotalCourse() { return total_course; }
```

(答对得 2 分，未加 static 得 1 分)

在 class CCourse 定义中增加一条：

```
friend char *getCourseName(const CCourse &r_course);
```

(答对得 1 分，未加 friend 得 0 分)

在类外部定义：

```
char *getCourseName(const CCourse &r_course)
{
```

```
    return r_course.p_name; (答对得 2 分)
```

3. 在 class CCourse 定义中增加一条：

```
public:
```

```
bool operator <(const CCourse &r_course);
```

(答对得 2 分)

在类外部定义：

```
bool CCourse::operator <(const CCourse &r_course)
```



```
{
    if (credit < r_course.credit)
        return true;
    else
        return false;    (答对得 3 分)
```

在 class CCourse 定义中增加一条:

```
public:
    bool operator >=(const CCourse &r_course);    (答对得 2 分)
```

在类外部定义:

```
bool CCourse::operator >=(const CCourse &r_course)
{
    if (credit >= r_course.credit)
        return true;
    else
        return false;
}
```

(答对得 3 分)

4.

```
class CHLP : public CCourse
{
private:
    char *p_openby;
public:
    CHLP(long no_val, char *p_val, float credit_val, char *p_open) : CCourse(no_val,
p_val, credit_val)
    {
        p_openby = new char[strlen(p_open)+1];
        strcpy(p_openby, p_open);
    }
    ~CHLP() { delete p_openby; }
    void studyFor() { cout << "Study for structured programming" << endl; }
};
```

(答对得 5 分, 其中构造函数 3 分, 析构函数 1 分, studyFor() 函数 1 分)

```
class COOP : public CCourse
{
private:
    char *p_openby;
public:
    COOP(long no_val, char *p_val, float credit_val, char *p_open) : CCourse(no_val,
p_val, credit_val)
    {
        p_openby = new char[strlen(p_open)+1];
        strcpy(p_openby, p_open);
    }
    ~COOP() { delete p_openby; }
```

```
void studyFor() { cout << "Study for object oriented programming" << endl; }  
};
```

(答对得 5 分，其中构造函数 3 分，析构函数 1 分，studyFor() 函数 1 分)

五、

1. 在 class CCourse 定义中增加一条：

```
public:  
    virtual void studyFor() { cout << "study for degree\n"; }
```

(答对得 2 分)

增加：

```
#include <stdlib.h>
```

主函数可定义为：

```
void main()  
{  
    char choice, instructor[10];  
    float credit;  
    long id;  
    CCourse *p_course;  
  
    cout << "Select course:\n";  
    cout << "1. for High Level Language Programming\n";  
    cout << "2. for Object Oriented Programming\n";  
    cin >> choice;  
    cout << "Enter course number: ";  
    cin >> id;  
    cout << "Enter credit: ";  
    cin >> credit;  
    cout << "Enter instructor name: ";  
    cin >> instructor;  
  
    switch (choice)  
    {  
        case '1':  
            p_course = new CHLP(id, "高级语言程序设计", credit, instructor);  
            break;  
        case '2':  
            p_course = new COOP(id, "面向对象程序设计", credit, instructor);  
            break;  
        default:  
            exit(0);  
    }  
    p_course->studyFor();  
    delete p_course;  
}
```

(答对得 13 分)

2.

```
#include <iostream>
using namespace std;

template <class T>
void swap(T &a, T &b)
{
    T temp;

    temp = a;
    a = b;
    b = temp;
}

template <class T>
void bubbleSort(T a[], int n)
{
    int i, j;

    for (i=1; i < n; i++)
        for (j=0; j < n-i; j++)
            if (a[j] > a[j+1])
                swap(a[j], a[j+1]);
}

template <class T1>
void print(T1 a[], int n)
{
    for (int i=0; i < n; i++)
        cout << a[i] << " ";
    cout << endl;
}

void main()
{
    int a[] = {2, 3, 1, 6, 43, 22};
    double b[] = {2.3, 3.2, 1.6, -6.0, 4.3, 2.2};
    print(a, 6);
    bubbleSort(a, 6);
    print(a, 6);
    print(b, 6);
    bubbleSort(b, 6);
    print(b, 6);
    (答对得 15 分)
```