《C++面向对象程序设计》模拟试题（7）

参考答案

一、单选题(共 20 分，每题 2 分)

B A C C A　B D B C D

二、判断正误，对于你认为错误的论述，说明原因或举出反例

1. 使用没有初始化的指针，是一种语法错误。
   错。正确指出对错得 1 分。指出不是语法错误，得 2 分。
2. 运算符的优先级可以通过重载来改变。
   错。正确指出对错得 1 分。指出运算符的优先级保持不变，得 2 分。
3. 在形如 x=expression; 的赋值语句中，x 的值总是等于右端表达式 expression 的值。
   错。正确指出对错得 1 分。指出不总是等于或举反例说明，得 2 分。
4. 当定义一个类时，其数据成员的存储空间即被分配。
   错。正确指出对错得 1 分。指出当创建对象时存储空间才被分配或举反例说明，得 2 分。
5. 类的私有成员只能由该类的成员函数直接访问。
   错。正确指出对错得 1 分。指出友元函数或举反例说明，得 2 分。
6. 类的静态成员函数不能声明为常函数，构造函数和析构函数也不能。
   正确。
7. 可以将基类对象看作派生类对象。
   错。正确指出对错得 1 分。指出子类对象可以看作父类对象或举反例说明，得 2 分。
8. 继承用以改进数据隐藏和封装。
   错。正确指出对错得 1 分。指出继承破坏了数据隐藏和封装或举反例说明，得 2 分。
9. 通过单参构造函数可以把用户自定义类型的对象转换成内置类型的数据。
   错。正确指出对错得 1 分。指出说法相反或举反例说明，得 2 分。
10. 抽象类中的所有成员函数必须声明为纯虚函数。
    错。正确指出对错得 1 分。指出不必都声明纯虚函数，得 2 分。

三、回答下列各题（共 20 分，每题 4 分）

1. 是重载函数。当调用对象是常对象时，调用常成员函数，否则，调用非常成员函数。
2. 浅复制时，如果改变被复制对象的指针数据成员的内容，复制后对象的内容也跟着改变。
3. 数据成员是常量、数据成员是引用类型、基类未提供无参构造函数。
4. 动物(Animal)和虎(Tiger)、鸟(Bird)之间是泛化关系；
   鸟(Bird)和企鹅(Penguin)、鸽子(Pigeon)之间是泛化关系；
   鸽子(Pigeon)和鸽群(Pigeon Group)之间是聚合关系；
   鸽子(Pigeon)和飞翔(Fly)之间是实现关系。
5. 分配给奶农，能更好地适应“不同类型的奶农有不同的挤奶行为”。实现时，可在奶农的子类中给出不同的挤奶行为实现。

分配给奶牛，能更好地适应“不同类型的奶牛有不同的被挤奶行为”。实现时，可在奶牛的子类中给出不同的被挤奶行为实现。

四、程序设计（8 分）

```
class Grid:public Rect {
public:
    Grid (int n):Rect(NULL),count(n) {
        precs=new Shape*[count];
        for (int i=0; i<count; i++) {
            precs[i]=new Rect(NULL);
        }
    }
    void Draw ( ) {
        for (int i=0; i<count; i++)
            precs[i]->Draw();
    }
```

```
~Grid()
    {
        for (int i=0; i<count; i++) {
            delete precs[i];
        }
        delete[ ] precs;
    }
private:
    int count;
    Shape ** precs;
};
```

五、分别写出下面两个程序的运行结果

1． 2　3

2． Base( )　Data( )　Child( )　Base(const Base&)　Data(const Data&)
~Child( )　~Data( )　~Child( )　~Data( )

六、（分油）

```
class Bottle {
public:
    Bottle( int maxCapacity,int use=0)
        :capacity(maxCapacity),used(use)
    {}
    void Pour(Bottle& other)  {
        int maxPoured =
            other.capacity - other.used;
        if ( used > maxPoured ) {
            used -= maxPoured;
            other.used +=maxPoured;
        } else  {
            other.used += used;
            used = 0;
        }
    }
    int GetUsed() const
```

```
int main() {
    Bottle b10(10,10),b7(7),b3(3);
    ///10 斤倒 7 斤，7 斤倒 3 斤，
    ///3 斤倒 10 斤，7 斤倒 3 斤，3 斤倒 10 斤
    b10.Pour(b7);
    b7.Pour(b3);
    b3.Pour(b10);
    b7.Pour(b3);
    b3.Pour(b10);
    ///7 斤倒 3 斤，10 斤倒 7 斤，
    ///7 斤倒 3 斤，3 斤倒 10 斤
    b7.Pour(b3);
    b10.Pour(b7);
    b7.Pour(b3);
    b3.Pour(b10);
    ///这时，10 斤瓶和 7 斤瓶
    cout<<"B10 = "<<b10.GetUsed()<<endl;
```

```
        {     return used;     }
private:
    const int capacity;
    int     used;
};
```

```
    cout<<"B7    = "<<b7.GetUsed()<<endl;
    cout<<"B3    = "<<b3.GetUsed()<<endl;
    return 0;
}
```

七、（连分数）

```
class CFraction   {
public:
    CFraction(const unsigned int data[ ], int theCount)
            :count(theCount)
    {
            items = new unsigned int[theCount];
            for(int i=0;i<theCount;++i)    items[i] = data[i];
    }
    ~CFraction( ) {     delete[] items; }
    //取前 n 项对应分数的分子和分母, n>0 且 n≤count
    void    GetFraction( int& num, int & den, int n) const
      {     GetPart(num,den,1,n);       }
private:
    //取第 from 项到 to 项对应分数的分子和分母
    void GetPart( int& num, int & den, int from,int to) const;
private:
    const   unsigned int     count;   //项数
    unsigned        int *     items;   //各项的整数值
};
//取第 from 项到 to 项对应分数的分子和分母
void    CFraction::GetPart( int& num, int & den, int from,int to) const
{
    if(from >= to) {
        num = items[to-1];
        den = 1;
    } else {
        GetPart(num,den,from+1,to);
        int newNum = items[from-1]*num+den;
        den = num;
        num = newNum;
    }
}
```

八、(Reader，Writer)

<table>
<tr><td>class Reader {
public:
　　virtaul ~Reader( ) { }
　　virtual int read()=0;
};</td><td>class KeyboardReader: public Reader　{
public:
　　virtual int read() { return ReadKeyboard(); }
};</td></tr>
<tr><td>class Writer {
public:
　　virtaul ~Writer( ) { }
　　virtual int write(int ch)=0;
};</td><td>class PrintWriter: public Writer
{
public:
　　virtual int write(int ch)
　　{ return　WritePrinter(ch); }
};</td></tr>
<tr><td colspan="2">KeyboardReader DefaultReader;
PrintWriter DefaultWriter;
void Copy(Reader& reader=DefaultReader, Writer& writer=DefaultWriter)
{
　　int c;
　　while ((c=reader.read()) != EOF)
　　writer.write(c);
}</td></tr>
</table>

（全卷完）