座号 ____

秋季学期试卷(A 卷)

课程名: 面向对象程序设计 课程号: 08305121 学分: 5

得分 ____

一、判断题（每小题 2 分，共 20 分）

1. C++的结构体(**struct**)及类(**class**)中成员的访问属性均默认为**public**。 ( )
2. 任何类都至少有一个构造函数；都有一个析构函数。 ( )
3. 执行赋值运算时由系统自动调用拷贝构造函数创建左值对象。 ( )
4. 对象的空间通常由其基本空间和资源空间构成。 ( )
5. 类的静态数据成员占用具体对象的存储空间。 ( )
6. 静态成员函数不必通过该类的对象调用，可直接用类名及作用域区分符（::）进行调用。 ( )
7. 创建具有 10 个元素的对象数组时，将调用 10 次相应的构造函数或拷贝构造函数。 ( )
8. 非静态成员函数中，隐含传递的形参 **this** 是一个指针常量，即它是一个锁定指向的指针。因此赋值语句 **this**=NULL;企图改变其指向是错误的。 ( )
9. 派生类不继承基类中的访问属性为 **private** 的数据成员。 ( )
10. 重载运算符时，可以改变运算符的优先级和结合方向。 ( )

得分 ____

二、填空题（每空 2 分，共 20 分）

请根据运行结果，完成程序。

```
#include <iostream>
#include <cstring>
using namespace ①_____;

template <typename T> void SWAP1(T &x, T &y)
{
    ②_____ temp = x;
    x = y; y = temp;
}
template <typename T> void SWAP2(T *x, T *y)
{
    ③_____ temp = *x;
    *x = *y; *y = temp;
}
template <typename T> void SWAP3(T *&x, T *&y)
{
    ④_____ temp = x;
    x = y; y = temp;
}

int main()
{
    int m = 3, n = 5;
    double a = 3.3, b = 5.5;
    char *str1, *str2;
    str1 = new char[80];
    str2 = new char[80];
    strcpy(str1, "Tom");
    strcpy(str2, "Jerry");

    SWAP1(⑤_____, ⑥_____);
    cout << "m = " << m << ", n = " << n << endl;

    SWAP2(⑦_____, ⑧_____);
    cout << "a = " << a << ", b = " << b << endl;

    SWAP3(⑨_____, ⑩_____);
    cout << "str1: " << str1 << ", str2: " << str2 << endl;
    delete [] str1;
    delete [] str2;
    return 0;
}
```

运行结果

```
m = 5, n = 3
a = 5.5, b = 3.3
str1: Jerry, str2: Tom
```

得分 | | **三、阅读程序写出运行结果（每空 1 分，共 30 分）**

3.1（10 分）

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

class Complex
{
public:
   Complex(double real=0, double imag=0): re(real), im(imag) {}
   friend Complex operator+(const Complex &x, const Complex &y)
   {
      return Complex(x.re+y.re, x.im+y.im);
   }
   Complex & operator+=(const Complex &x)
   {
      re += x.re;
      im += x.im;
      return *this;
   }
   Complex & operator++()
   {
      re++;                              // 实部增 1，虚部不变
      return *this;
   }
   Complex operator++(int)
   {
      Complex temp(*this);
      re++;                              // 实部增 1，虚部不变
      return temp;
   }
   friend ostream & operator<<(ostream &out, const Complex &c)
   {
      out << '(' << c.re << ", " << c.im << ')';
      return out;
   }
   friend istream & operator>>(istream &in, Complex &c)
   {
      char str[80];
      in.getline(str, 80, '(');
      in.getline(str, 80, ',');
      c.re = atof(str);
      in.getline(str, 80, ')');
      c.im = atof(str);
      return in;
   }
private:
   double re, im;
};

int main()
{
   Complex a, b(3, 4), c, d;
   c = 5 + b;
   d = b + 5;
   cout << "a      : " << a << endl;
   cout << "b      : " << b << endl;
   cout << "c      : " << c << endl;
   cout << "d      : " << d << endl;
   cout << "++a    : " << ++a << endl;
   cout << "a      : " << a << endl;
   cout << "b++    : " << b++ << endl;
   cout << "b      : " << b << endl;
   cout << "a += b : " <<(a+=b) <<endl;
   cout << "a += a : " <<(a+=a) <<endl;
   return 0;
}
```

运行结果(3.1)

a : _____

b : _____

c : _____

d : _____

++a : _____

a : _____

b++ : _____

b : _____

a += b : _____

a += a : _____

3.2（10 分）

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

class Language
{
public:
   virtual void SayHello() const
   {
      cout << "How are you!" << endl;
   }
   virtual void SayByebye() const
   {
      cout << "Goodbye!" << endl;
   }
};

class Chinese : public Language
{
public:
   void SayHello() const
   {
      cout << "你好！" << endl;
   }
```

```
   void SayByebye() const
   {
      cout << "再见！" << endl;
   }
};

class Japanese : public Language
{
public:
   void SayHello() const
   {
      cout << "こんにちは!" << endl;
   }
   void SayByebye() const
   {
      cout << "さようなら!" << endl;
   }
};

class English : public Language
{
public:
   void SayHello() const
   {
      cout << "Hello!" << endl;
   }
   void SayByebye() const
   {
      cout << "Bye-bye!" << endl;
   }
};

int test0302A(const Language &p)
{
   p.SayHello();
   p.SayByebye();
   return 0;
}

int test0302B(const Language *p)
{
   p->SayHello();
   p->SayByebye();
   return 0;
}
```

```
int test0302C(const Language p)
{
   p.SayHello();
   p.SayByebye();
   return 0;
}

int main()
{
   Chinese c;
   Japanese j;
   English e;

   test0302A(c);
   test0302A(j);
   test0302A(e);
   test0302B(&c);
   test0302C(c);
   return 0;
}
```

运行结果(3.2)

3.3（10 分）

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

class Test
{
public:
   Test(int a=0) : x(a)
   {
      if(flag==0)
      {
         max = min = x;
         flag++;
      }
      else
      {
         if(x>max) max = x;
         if(x<min) min = x;
      }
   }
   Test & operator+=(const Test &t)
   {
      x += t.x;
      return *this;
   }
```

```
   Test & operator-=(const Test &t)
   {
      x -= t.x;
      return *this;
   }
   static void Show()
   {
      cout << "max = " << max << ", min = " << min << endl;
   }
   friend ostream & operator<<(ostream &out, const Test &t)
   {
      out << t.x;
      return out;
   }
private:
   static int max, min, flag;
   int x;
};

int Test::max, Test::min, Test::flag=0;  // 静态数据成员定义及初始化

int main()
{
   Test a, b(100), c(-50);
   cout << "a = " << a << ", b = " << b << ", c = " << c << endl;
   a.Show();
   b += b;
   c -= 100;
   Test d(a), e(b), f(c);
   cout << "d = " << d << ", e = " << e << ", f = " << f << endl;
   a.Show();
   return 0;
}
```

运行结果

a =___, b =_____, c =_____

max =_____, min =_____

d =___, e =_____, f =_____

max =_____, min =_____

得分	

四、（30 分）完成成员函数的定义。设计了课程类（class Course;）和学生类（class student;）简单模拟学生成绩系统。请根据程序的运行结果，在类体外完成所需定义的 5 个成员函数（每个函数 6 分，按函数首部设计、函数功能实现及返回评分）。

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

class Course
{
public:
   Course(const string &Id="", const string &Name="", int Credit=0)
      : id(Id), name(Name), credit(Credit) {}
   friend ostream & operator<<(ostream &out, const Course &c)
   {
      out << c.id << '\t' << setw(20)<< left << c.name
          << '\t' << c.credit;
      return out;
   }
private:
   string id, name;           // 课程编号、课程名称
   int credit;                // 课程的学分
};

class Student
{
public:
   Student(const string &Id="", const string &Name="",
          int Num=0, const Course *array=NULL);// ① 构造函数
   Student(const Student &s);                  // ② 拷贝构造函数
   virtual ~Student();                         // ③ 析构函数
   void Set(const string &Id, const string &Name);
                                  // ④ 设置（修改）学生的学号、姓名
   void Set(int Num, const Course *array, const double *Score);
                                  // ⑤ 设置（修改）学生所选课程、各课程成绩
   Student & operator=(const Student &s)       // 赋值运算符函数
   {
      if(&s==this) return *this;
      if(course!=NULL) delete [] course;
      if(score!=NULL) delete [] score;
      id = s.id;
      name = s.name;
      if(s.num==0 || s.course==NULL || s.score==NULL)
      {
         num = 0;
         course = NULL;
         score = NULL;
      }
```

```
      else
      {
         num = s.num;
         course = new Course[num];
         score = new double[num];
         for(int i=0; i<num; i++)
         {
            course[i] = s.course[i];
            score[i] = s.score[i];
         }
      }
      return *this;
   }
   friend ostream & operator<<(ostream &out, const Student &s)
   // 重载输出运算符（友元）函数
   {
      out << "学号: " << s.id << ", 姓名: " << s.name << endl;
      for(int i=0; i<s.num; i++)
         out << s.course[i] << '\t' << s.score[i] << '\t' << endl;
      return out;
   }
private:
   string id, name;          // 学生的学号、姓名
   int num;                  // 本学期所选课程数量
   Course *course;           // 具体的课程信息所占资源空间的地址
   double *score;            // 各课程的成绩
};
int main()
{
   Course array[] = {Course("08305121", "面向对象程序设计", 5),
                     Course("08305071", "数字逻辑 A", 5),
                     Course("08305072", "数字逻辑实验", 1),
                     Course("01034119", "大学物理(3)", 4),
                     Course("03004403", "大学英语 D 级(4)", 4)};
   double score[] = {90, 85, 80, 75, 80};
   int n = sizeof(array)/sizeof(*array);
   Student Zhang("16123721", "张三", n, array), Li;
   cout << Zhang << '\n' << Li << "===========" << endl;
   Zhang.Set(n, array, score);
   Li.Set("16127890", "李四");
   Li.Set(3, array, score);
   cout << Zhang << '\n' << Li << endl;
   return 0;
}
```

运行结果(4)

```
学号: 16123721, 姓名: 张三
08305121      面向对象程序设计      5        0
08305071      数字逻辑 A            5        0
08305072      数字逻辑实验          1        0
01034119      大学物理(3)           4        0
03004403      大学英语 D 级(4)      4        0

学号: , 姓名:
===========
学号: 16123721, 姓名: 张三
08305121      面向对象程序设计      5        90
08305071      数字逻辑 A            5        85
08305072      数字逻辑实验          1        80
01034119      大学物理(3)           4        75
03004403      大学英语 D 级(4)      4        80

学号: 16127890, 姓名: 李四
08305121      面向对象程序设计      5        90
08305071      数字逻辑 A            5        85
08305072      数字逻辑实验          1        80
```

请根据给定的运行结果，在类体外完成 5 个在类中声明的成员函数。