第一章

一、选择题

1.B; (typedef ,typeid ,typename，都为保留字);

2.C; (标识符，应该以字母或，下划线开头);

3.C; (标识符中有的特殊符号，只能有下划线);

二、填空题

1. cin，cout

2. new，delete

3. int a(55);

三、改错题

1.没有定义变量 num;

2.不能给变量 x，声明指向常量的指针 const int *p=&x; 如果吧 x 定义为常量 const，*p 不能当作“左值”。

3.p 为常量指针，不能吧 p 作为“左值”，p=&y，错误。

四、编程题

1. 分别用字符和 ASCII 码形式输出整数值 65 和 66.

```
#include < iostream >
using namespace std;
void main()
{
    char a='A',b='B';
    int ascii_1=53,ascii_2=54;//ASCII 码中的，5 和 6
    cout<<"字符输出："<<(int)a<<"，"<<(int)b<< endl;
    cout<<"ASCII 码输出："<<(char)ascii_2<<(char)ascii_1<<"，";
    cout<<(char)ascii_1<<(char)ascii_1<< endl;
}
```

2.编写一个 int 型变量分配 100 个整形空间的程序。

```
#include < iostream >
using namespace std;
void main()
{
    int *p;
```

```
    p = new int[100];
    for(int i = 0;i < 100;i++)
    {
        *(p+i)=i;
    }
    for(i = 0;i < 100;i++)
    {
        cout<<*(p+i)<<"，";
    }
    delete p;

}
```

3.编写完整的程序，它读入 15 个 float 值，用指针把它们存放在一个存储快里，然后输出这些值和以及最小值。

```
#include < iostream >
#include < algorithm > //用于数组排列的头文件
using namespace std;
void main()
{
    float *p;
    p = new float[15];
    cout<<"输入 15 个 float 类型的值：" << endl;
    for(int i = 0;i < 15 ; i++)
    {
        cin>>*(p+i);
    }
    for(i = 0;i < 15;i++)
    {
        cout<<*(p+i)<<"，";
    }
    sort(p,p+15);
    cout<<"\n 最小的是："<<*(p)<< endl;
    delete p;

}
```

4.声明如下数组：

```
int a[] = {1 ,2 ,3, 4, 5, 6, 7, 8}；
```

先查找 4 的位置，讲数组 a 复制给数组 b，然后将数组 a 的内容反转，再查找 4 的位置，最后分别输出数组 a 和 b 的内容。

```
#include < iostream>
#include < algorithm>
#include < functional>
using namespace std;
void main()
{
    int a[]={1,2,3,4,5,6,7,8},b[8];
    cout<<"数组 a 中‘4’的位置是："<< find(a,a+8,4)<< endl;//查找 4 的位置
    copy(a,a+8,b);//将数组 a 复制给数组 b
    reverse_copy(b,b+8,a);//把数组 b，逆向复制给 a，完成 a 的逆转
    cout<<"数组 a 反转后，‘4’的位置是："<< find(a,a+8,4)<< endl;//在查找 4 的位置

    cout<<"数字 a 的内容："<< endl;
    for(int i=0;i<8;i++)
        cout<< a[i]<<" ,";
    cout<<"\n 数组 b 中的内容："<< endl;
    for(i=0;i<8;i++)
        cout<< b[i]<<" ,";

}
```

[color=#FF0000]第二章 [/color]

一、单项选择

1.D； 2.D；

二、作图题

1． 已知一个学生类具有性别和年龄两个属性，男学生张明的年龄为 12 岁，女学生李红的年龄为 11 岁。给出这个学生类的类图和它们的对象图。

(类)Student (对象)张明 (对象)李红

string sex; sex(男); sex(女);

int age; age(11); age(12);

方法… 方法… 方法…

2． 一个圆具有圆心坐标和半径两个属性，并且能够给出圆面积，请画出这个圆类的类图。

(类) Circularity (类)Point

Point p; float x;

float radii; float y;

float getX();

float getAcreage(); float getY();

3． 画出一个班级类的类图，为它设计必要的属性以表示这个类的特征。

(类) PubClass

string no;//编号
int num;//人数
…

4． 画出一种电话卡的类图，为它设计必要的属性。
(类) Card
long no;//编号
float balance;//余额

5． 为上题的电话卡设计必要的成员函数，以便提供基本服务。
(类) Card
long no;//编号
float balance;//余额
float getBalance();//显示余额

三、编程题
1.使用多种方法编写将两个字符串连接在一起的程序。

```
#include < iostream >
#include < string >
using namespace std;
void main()
{
    //使用 string 类定义字符串，完成字符串连接
    string str1("C++"),str2("程序设计");
    string str3;
    str3 = str1+str2;//连接方式 1
    cout<< str3<< endl;

    //使用 char 数组定义字符串，完成连接
    char c1[] = {"c++"},c2[] = {"program"};
    char c3[20];
    int i=0,k=0;
    for(i=0;i<20;i++)//初始化 c3
        c3[i]='\0';
    i=0;
    while(c1[i]!='\0')
    {
        c3[k]=c1[i];
        i++;
        k++;
    }
    i=0;
```

```
    while(c2[i]!='\0')
    {
        c3[k]=c2[i];
        i++;
        k++;
    }
    cout<< c3<< endl;
}
```

2.已知一个 string 的对象 str 的内容为“We are here!”，使用多种方法输出“h”。

```
#include < iostream >
#include < functional >
#include < algorithm >
#include < string >
using namespace std;
void main()
{
    string str1("We are here!");
    cout<< str1[7]<< endl;//通过数组

    string str2=str1.substr(7,1);//通过得到子字符串
    cout<< str2<< endl;

    char *p=find(str1.begin(),str1.end(),'h');//通过 find 函数
    if(p)
        cout<<*p<< endl;
}
```

[color=#F70909]
第三章 [/color]
一、填空题
1.函数原型声明；

2.inline

3.传值，传引用

4.函数 func 返回引用

5.int *fun(char,int);

二、单项选择题

1.A； 2.C； 3.D；

三、改错题

1.y = x * x - T; 错误，T 是类型，不是变量，不能参加运算；

2.y 没有类型。

```
T max(T x, T y)
{
return (x>y) ? (x) : (y) ;
}
```

3.函数 change 的参数定义成了常量，只能使用参数，而无权修改他。

```
void change (string & s)
{
s = s + "pig!";
}
```

四、编程题

1.编写一个求方程 ax2 + bx + c = 0 的根 的程序，用 3 个函数分别求当 b2-4ac 大于零、等于零、和小于零时的方程的根。要求从主函数输入 a,b,c 的值并输出结果。

```
#include < iostream.h >
#include < math.h >
void equation_1 (int a, int b, int c)
{
    double x1, x2, temp;
    temp = b*b - 4 * a * c;
    x1 = (-b + sqrt(temp) ) / (2 * a * 1.0);
    x2 = (-b - sqrt(temp) ) / (2 * a * 1.0);
    cout<<"两个不相等的实根"<< endl;
    cout<<"x1 = "<< x1<<",   x2 = "<< x2<< endl;
}
void equation_2 (int a, int b, int c)
{
    double x1, x2, temp;
    temp = b*b - 4 * a * c;
    x1 = (-b + sqrt(temp) ) / (2 * a * 1.0);
```

```
        x2 = x1;
        cout<<"两个相等的实根"<< endl;
        cout<<"x1 = "<< x1<<",    x2 = "<< x2<< endl;

}
void equation_3 (int a, int b, int c)
{
        double temp, real1, real2, image1, image2;
        temp = - (b*b - 4 * a * c);
        real1 = -b / (2 * a *1.0);
        real2 = real1;
        image1 = sqrt(temp);
        image2 = - image1;
        cout<<"两个虚根"<< endl;
        cout<<"x1 = "<< real1<<" + "<< image1<<"j"<< endl;
        cout<<"x2 = "<< real2<<" + "<< image2<<"j"<< endl;

}
void main()
{
        int a, b, c;
        double temp;
        cout<<"输入 a,b,c 的值"<< endl;
        cin>>a>>b>>c;
        cout<<"方程为："<< a<<"x*x+"<< b<<"x+"<< c<<" = 0"<< endl;
        temp = b*b - 4 * a * c;
        if(temp > 0)
                equation_1 (a, b, c);
        if(temp == 0)
                equation_2 (a, b, c);
        if(temp < 0)
                equation_3 (a, b, c);

}
```

2.定义函数 up(ch)，如字符变量 ch 是小写字母就转换成大写字母并通过 up 返回，否则字符 ch 不改变。要求在短小而完全的程序中显示这个程序是怎样被调用的。

```
#include < iostream >
using namespace std;
char up (char c)
{
```

```
    if(c >= 97 && c <= 123)
        return (c - 32) ;
    else
        return c;
}
void main()
{
    int i;
    char c[15] = {'A','v','e','t','E','T','%','&','4','Y','e','i','@','9','^'};
    for(i = 0 ; i < 15 ; i++)
        cout<< up(c[i])<<", ";
    cout<< endl;
}
```

3.编写主程序条用带实数 r 和整数 n 两个参数的函数并输出 r 的 n 次幂。

```
#include < iostream.h >
#include < math.h >
double power(double a, int b)
{
    int i;
    double result = 1.0;
    for(i=0;i< b;i++)
        result = result * a;
    return result;
}
void main()
{
    double r;
    int n;
    cout<<"r = ";
    cin>>r;
    cout<<"n = ";
    cin>>n;
    cout<< r<<"的"<< n<<"次幂是："<< power(r,n)<< endl;
}
```

4.编写有字符型参数 C 和整形参数 N 的函数，让他们显示出由字符 C 组成的三角形。其方式为第 1 行有 1 个字符 C，第 2 行有 2 个字符 C ，等等。

```
#include < iostream >
using namespace std;
void print_triangle(char c, int n)
{
	int i, j;
	for(i=0; i< n; i++)
	{
		for(j=0; j<=i; j++)
		{
			cout<< c;
		}
		cout<< endl;
	}
}
void main()
{
	print_triangle('a',10);
}
```

5.编写一个 ieqiu 字符串长度的函数，strlen（），再用 strlen（）函数编写一个函数 revers（s）的倒序递归程序，使字符串 s 逆序。

```
#include < iostream >
#include < string >
using namespace std;
int strlen(char *str)
{
	int len = 0;
	while(str[len] != '\0')
	{
		len++;
	}
	return len;
}
void revers(char *b)
{
	char c;
	int j, len;
	len=strlen(b);
	j=len/2-1;
	while(j>=0)
	{
		c=*(b+j);
```

```
            *(b+j)=*(b+len-j-1);
            *(b+len-j-1)=c;
             j--;
        }
        b[len]='\0';
}
void main()
{
        char str[]={"1234567890"};
        cout<< str<<"----的长度："<< strlen(str)<< endl;
        cout<< str<< endl;//倒序前
        revers(str);//
        cout<< str<< endl;//倒序后
}
```

6.用函数模板实现 3 个数值中按最小值到最大值排序的程序。

```
#include < iostream >
using namespace std;
template
void sort(T a, T b, T c)
{
        T array[3],temp;
        int i,j;

        array[0] = a;
        array[1] = b;
        array[2] = c;
        for(i=0;i<3;i++)
        {
                for(j=0;j<2;j++)
                        if(array[j]>array[j+1])
                        {
                                temp = array[j];
                                array[j] = array[j+1];
                                array[j+1] = temp;
                        }
        }
        cout<< array[0]<< array[1]<< array[2]<< endl;

}
void main()
{
```

```
    sort(5,1,9);
}
```

7.利用函数模板设计一个求数组元素中和的函数，并检验之。

```
#include < iostream >
using namespace std;
template
T sum (T a[],int n)
{
    int i;
    T s=0;
    for(i=0;i< n;i++)
        s = s + a[i];
    return s;
}
void main ()
{
    int a[5]={1,2,3,4,5};
    int s = sum(a,5);
    cout<< s<< endl;
}
```

8.重载上题中的函数模板，使他能够进行两个数组的求和。

```
#include < iostream >
using namespace std;
template
T sum (T a[], int n)
{
    int i;
    T s=0;
    for(i=0;i< n;i++)
        s = s + a[i];
    return s;
}

template //重载上面的模板
T sum (T a[], int n, T b[], int m)
{
    return sum(a,n)+sum(b,m);
}
```

```
void main ()
{
	int a[5]={1,2,3,4,5};
	int b[10]={1,2,3,4,5,6,7,8,9,10};
	int s1 = sum(a, 5);
	int s2 = sum(b, 10);
	int s3= sum(a, 5, b, 10);
	cout<< s1<< endl;
	cout<< s2<< endl;
	cout<< s3<< endl;
}
```

[color=#EE1111]第四章[/color]

一、填空题

1.数据成员、成员函数；

2.类、重载、1；

3.fun:fun(fun &)、fun:fun(const fun &);

二、单项选择题

1.C。2.C。3.没又答案，应该是 A::~A(void)、或 A::~A()。4.B。 5.C。 6.C。 7.D

三、改错题

1.return m;---错误，没又定义变量 m；

2.A.init(24,56);---错误，应该先定义 A 对象：Point A;

四、完成程序题

1.

```
#include < iostream >
using namespace std;
class base
{
	private :  //私有数据成员
		int a, b;
	public :
		void init(int x, int y)//公有函数
		{
			a = x;
			b = y;
		}
		void print()
```

```
        {
            cout<<"2 * "<< a<<" - "<< b<<" = "<<(2*a-b)<< endl;
        }

};
void main()
{
    base a;
    a.init(68,55);
    a.print();
}
```

2.

```
#include
using namespace std;
class Point
{
    private :
        int m, n;
    public :
        Point(int, int);//整型变量，为参数的构造函数
        Point(Point&);//复制构造函数的原型
        print()
        {
            cout<<"m = "<< m<<", n = "<< n<< endl;
        }

};
Point::Point(int a, int b)
{
    m = a;
    n = b;
}
Point::Point(Point & t)//复制构造函数的定义
{
    m = t.m;
    n = t.n;
}
void main()
{
    Point a(10,89);
    Point b(a);
    a.print();
```

```
    b.print();
}
```

五、程序分析题

1.没有结果，因为没有 main 函数
如果加 main 函数

```
void main()
{
    base b(10, 20);
}
```

输出：
初始化...10,20
Destory...10,20

2.
输出：
55

六、编程题

1.设计一个点类 Point，再设计一个矩形类，矩形类使用 Point 类的两个坐标点作为矩形的对角顶点。并可以输出 4 个坐标值和面积。使用测试程序验证程序。

```
#include
using namespace std;
class Point     //点类
{
    private:
            int x, y;//私有成员变量，坐标
    public :
            Point()//无参数的构造方法，对 xy 初始化
            {
                x = 0;
                y = 0;
            }
            Point(int a, int b)//又参数的构造方法，对 xy 赋值
            {
                x = a;
                y = b;
```

```
        }
        void setXY(int a, int b)//设置坐标的函数
        {
            x = a;
            y = b;
        }
        int getX()//得到 x 的方法
        {
            return x;
        }
        int getY()//得到有的函数
        {
            return y;
        }

};
class Rectangle      //矩形类
{
    private:
        Point point1, point2, point3, point4;//私有成员变量，4 个点的对象
    public :
        Rectangle();//类 Point 的无参构造函数已经对每个对象做初始化啦，这里不用对每个点多初始化了
        Rectangle(Point one, Point two)//用点对象做初始化的，构造函数，1 和 4 为对角顶点
        {
            point1 = one;
            point4 = two;
            init();
        }
        Rectangle(int x1, int y1, int x2, int y2)//用两对坐标做初始化，构造函数，1 和 4 为对角顶点
        {
            point1.setXY(x1, y1);
            point4.setXY(x2, y2);
            init();
        }
        void init()//给另外两个点做初始化的函数
        {
            point2.setXY(point4.getX(), point1.getY() );
            point3.setXY(point1.getX(), point4.getY() );
        }
        void printPoint()//打印四个点的函数
        {
```

```
            cout<<"A：("<< point1.getX() <<"， "<< point1.getY() <<")"<< endl;
            cout<<"B：("<< point2.getX() <<"， "<< point2.getY() <<")"<< endl;
            cout<<"C：("<< point3.getX() <<"， "<< point3.getY() <<")"<< endl;
            cout<<"D：("<< point4.getX() <<"， "<< point4.getY() <<")"<< endl;
        }
        int getArea()//计算面积的函数
        {
            int height, width, area;
            height = point1.getY() - point3.getY();
            width = point1.getX() - point2.getX();
            area = height * width;
            if(area > 0)
                return area;
            else
                return -area;

        }

};
void main()
{
    Point p1(-15, 56), p2(89, -10);//定义两个点
    Rectangle r1(p1, p2);//用两个点做参数，声明一个矩形对象 r1
    Rectangle r2(1, 5, 5, 1);//用两队左边，声明一个矩形对象 r2

    cout<<"矩形 r1 的 4 个定点坐标："<< endl;
    r1.printPoint();
    cout<<"矩形 r1 的面积："<< r1.getArea() << endl;

    cout<<"\n 矩形 r2 的 4 个定点坐标："<< endl;
    r2.printPoint();
    cout<<"矩形 r2 的面积："<< r2.getArea() << endl;
}
```

2.使用内联函数设计一个类，用来表示直角坐标系中的任意一条直线并输出它的属性。

```
#include < iostream.h >
#include < math.h >
class Line
{
    private:
        int x1, y1, x2, y2;
    public :
```

```
        Line();
        Line(int =0, int =0, int =0, int=0 );
        void printPoint();
        double getLength();

};
inline Line::Line(int a, int b, int c, int d)
{
    x1 = a;
    y1 = b;
    x2 = c;
    y2 = d;
}
inline void Line::printPoint()
{
    cout<<"A："<< x1 <<", "<< y1 << endl;
    cout<<"B："<< x2 <<", "<< y2 << endl;
}
inline double Line::getLength()
{
    double length;
    length = sqrt((x2-x1)*(x2-x1) + (y2-y1)*(y2-y1) );
    return length;
}
void main()
{
    Line line(10,80,-10,12);
    line.printPoint();
    cout<< line.getLength() << endl;

}
```

[color=#FF0000]第五章 [/color]

一、填空题

1.常成员函数；
2.常量成员函数；
3.const

二、单项选择题

1.B; 2.A; 3.C; 4.A;

三、改错题

1.static int getn(){return number;}错误
静态成员函数，只允许访问静态成员变量，number 不是静态成员变量
2.void main()
{
    test *two[2] = {new test(4, 5), test(6 ,8)};
    for(i=0; i<2; i++)
        delete two[i];
}

四、完成程序题

```
#include < iostream >
using namespace std;
class test
{
    int x;
    public :
            test(int a)
            {
                x = a;
            }
            int GetX()
            {
                return x;
            }
};
void main()
{
    int i;//填空一，声明变量 i
    test *p, a[2][3] = {{1, 2, 3}, {4, 5, 6}};
    for( p=&a[0][0], i=0; i<=6; i++, p++)//填空 2，初始化 p，i
    {
        if((p-a[0])%3 == 0)
            cout<< endl;
        cout<< p->GetX() <<" ";
    }
}
```

五、编程题

1.声明复数的类，complex，使用友元函数 add 实现复数加法。

```
#include < iostream >
using namespace std;
class Complex
{
	private:
		double real, image;
	public :
		Complex(){}
		Complex(double a,double b)
		{
			real = a;
			image = b;
		}
		void setRI(double a, double b)
		{
			real = a;
			image = b;
		}
		double getReal()
		{
			return real;
		}
		double getImage()
		{
			return image;
		}
		void print()
		{
			if(image>0)
				cout<<"复数："<< real <<" + "<< image <<"i"<< endl;
			if(image<0)
				cout<<"复数："<< real <<" - "<< image <<"i"<< endl;
		}
		friend Complex add(Complex ,Complex);//声明友元函数
};


Complex add(Complex c1, Complex c2)//定义友元函数
{
	Complex c3;
	c3.real = c1.real + c2.real;//访问 Complex 类中的私有成员
	c3.image = c1.image + c2.image;
```

```
    return c3;
}
void main()
{
    Complex c1(19, 0.864), c2, c3;
    c2.setRI(90,125.012);
    c3 = add(c1, c2);
    cout<<"复数一：";c1.print();
    cout<<"复数二：";c2.print();
    cout<<"相加后：";c3.print();
}
```

2.例子 5.8，114 页例子不错；

3.编写一个程序，该程序建立一个动态数组，为动态数组的元素赋值，显示动态数组的值并删除动态数组。

```
#include < iostream >
using namespace std;
void main()
{
    int i, n, temp=0;
    cout<<"输入数组大小:";
    cin>>n;
    double *array = new double[n]; //用指针，动态申请数组大小
    cout<<"给每个数组元素赋值："<< endl;
    for(i=0; i < n; i++)
    {
        cout<<"array["<< i <<"] = ";
        cin>>temp;
        *(array+i) = temp;//给数组元素赋值
    }
    cout<<"动态数组个元素的值如下："<< endl;
    for(i=0; i < n; i++)
    {
        cout<<"array["<< i <<"] = "<< array[i] << endl;//打印数组元素
    }
    delete [] array;//释放内存
}
```

4.定义一个 Dog 类，它用静态数据成员 Dogs 记录 Dog 的个体数目，静态成员函数 GetDogs 用来存取 Dogs。设计并测试这个类。

```
#include < iostream >
using namespace std;
class Dog
{
    private:
            static int dogs;//静态数据成员，记录 Dog 的个体数目
    public :
            Dog(){}
            void setDogs(int a)
            {
                dogs = a;
            }
            static int getDogs()
            {
                return dogs;
            }
};
int Dog :: dogs = 25;//初始化静态数据成员
void main()
{
    cout<<"未定义 Dog 类对象之前：x = "<< Dog::getDogs() << endl;; //x 在产生对象之前即存在，输出 25
    Dog a, b;
    cout<<"a 中 x："<< a.getDogs() << endl;
    cout<<"b 中 x："<< b.getDogs() << endl;
    a.setDogs(360);
    cout<<"给对象 a 中的 x 设置值后："<< endl;
    cout<<"a 中 x："<< a.getDogs() << endl;
    cout<<"b 中 x："<< b.getDogs() << endl;
}
```

[color=#FF0000]
第六章[/color]

一、填空题
1.单一继承； 2.private protected public
二、单项选择
1.D;2.A;3.C;4.D;
三、改错题
1.类 derived 和 base 中均没变量 b，derived 的构造函数中的 m(b)错误；
2.Derived 类中重载 show()方法

```
    void Show()
    {
        Base1::Show();Base2::Show();
```

```
    }
```
四、编程题

1.设计一个基类，从基类派生圆柱，设计成员函数输出它们的面积和体积；

```
#include < iostream >
using namespace std;
class Basic//基类
{
    protected:
            double    r;
    public      :
            Basic(){ r = 0; }
            Basic(double a):r(a){}
};
class Circular : public Basic//从基类派生圆类
{
    protected:
            double area;
    public      :
            Circular(double a)
            {
                r = a;
                area = area = 3.1415926 * r * r;
            }
            double getArea()//返回圆面积
            {
                return area;
            }
};
class Column : public Circular//从圆类派生圆柱类
{
    protected:
            double h;
            double cubage;
    public    :
            Column(double a, double b) : Circular(a)
            {
                h = b;
                cubage = getArea() * h;
            }
            double getCubage()//返回圆柱体积函数
            {
                return cubage;
            }
```

```
};
void main()
{
	Circular circular(45);
	Column column(12, 10);
	cout<<"圆的面积："<< circular.getArea() << endl;
	cout<<"圆柱的体积："<< column.getCubage() << endl;
}
```

3.定义一个线段类作为矩形的基类，基类有起点和终点坐标，有输出左边和长度以及线段和 x 轴的夹角的成员函数。矩线段对象的两个坐标作为自己一条边的位置，它具有另外一条边，能输出矩形的 4 个顶点坐标。给出类的定义并用程序验证它们的功能。

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

class Point//点类
{
	protected:
		double x, y;
	public :
		Point(){}
		Point(double a, double b)
		{
			x = a; y = b;
		}
		double getX()
		{return x;}
		double getY()
		{return y;}
};
class Line
{
	protected:
		Point p1, p2;//Point 对象做成员
		double length, angle;
	public:
		Line(double a, double b, double c, double d):p1(a, b), p2(c, d)//用两对坐标初始化
线段
		{
			init();
		}
```

```
        Line(Point a, Point b)//用两个点的对象初始化线段
        {
            p1 = a; p2 = b;
            init();
        }
        void init()//计算线段长度，以及和 x 轴的夹角的度数
        {
            double x1 = p1.getX(), y1 = p1.getY();
            double x2 = p2.getX(), y2 = p2.getY();
            length = sqrt((x1-x2)*(x1-x2) + (y1-y2)*(y1-y2));
            angle = atan( (y2-y1) / (x2-x1) );
            angle = angle *180/3.141592653;
        }
        void printXY()
        {
            cout<<"("<< p1.getX() <<","<< p1.getY() <<");  ("<< p2.getX() <<","<<
p2.getY() <<")"<< endl;
        }
        void printLength()
        {
            cout<<"线段长度："<< length << endl;
        }
        void printAngle()
        {
            cout<<"与 x 轴的夹角："<< angle <<"°"<< endl;
        }
};
class Rectangle : public Line
{
    protected:
        Line *line;
    public:
        Rectangle(double a, double b, double c, double d, double e, double f, double g,
double h):Line(a,b,c,d)
        {
            line = new Line(e,f,g,h);
        }
        Rectangle(Point a, Point b, Point c, Point d) : Line(a, b)//4 个点对象，初始化
        {
            line = new Line(c, d);
        }
        void printPoint()
        {
            cout<<"矩形 4 个顶点:\n";
```

```
                printXY();
                line->printXY();
        }
};
void main()
{
    Point p1(0, 0), p2(4, 3), p3(12, 89), p4(10, -50);

    Line l1(0,0,4,3);
    l1.printXY();
    l1.printLength();
    l1.printAngle();

    Line l2(p1, p2);
    l2.printXY();
    l2.printLength();
    l2.printAngle();

    Rectangle r1(12,45,89,10,10,23,56,1);
    r1.printPoint();

    Rectangle r2(p1, p2, p3, p4);
    r2.printPoint();
}
```

4.基类是使用极坐标的点类，从它派生一个圆类，圆类用点类的左边作圆心，圆周通过极坐标原点，圆类有输出圆心直、圆半径和面积的成员函数。完成类的设计并验证之。

```
#include < iostream >
#include < cmath >
using namespace std;
class Point//点类
{
    protected:
            int x, y;
    public  :
            Point(){}
};

class Circular : public Point//圆类，继承点类
{
    protected:
            double r, area;
```

```
    public    :
              Circular(int a, int b)
              {
                   x = a;
                   y = b;
                   r = sqrt( x * x + y * y );
                   area = 3.1415926 * r * r;
              }
              void printPoint()
              {
                   cout<<"圆形直角坐标：("<< x <<", "<< y <<")"<< endl;
              }
              void printRadius()
              {
                   cout<<"圆的半径:"<< r << endl;
              }
              void printArea()
              {
                   cout<<"圆的面积:"<< area << endl;
              }
};
void main()
{
    Circular c(10,25);
    c.printPoint();
    c.printRadius();
    c.printArea();
}
```

5.设计一个线段基类，当创建五参数对象时，才要求用户输入长度。同样，其派生的直角三角形类也是在产生对象时要求输入两个直角边的长度。直角三角形在派生矩形类，矩形类的参数也由键盘输入。设计这些类并测试他们的功能。

```
#include < iostream >
#include < cmath >
using namespace std;
class Line//线段基类
{
    protected:
              double sizeA;
    public    :
              Line()
```

```
        {
            cout<<"输入线段的长度："<< endl;
            cin>>sizeA;
        }
        Line(double a)
        {
            sizeA = a;
        }
        double getLength()
        {
            return sizeA;
        }
};

class Triangle : public Line//三角形类
{
    protected:
        double sizeB, sizeC;
    public    :
        Triangle()
        {
            cout<<"输入线段长度："<< endl;
            cin>>sizeB;
            sizeC = sqrt( sizeB * sizeB + sizeA * sizeA );
        }
        void printSize()
        {
            cout<<"直角三角形，三条边分别为：";
            cout<<"A: "<< sizeA << ", b: "<< sizeB << ", C: "<< sizeC << endl;
        }
};

class Rectangle : public Triangle//矩形类
{
    protected:
        double sizeD;
    public    :
        Rectangle()
        {
            sizeC = sizeA;
            sizeD = sizeB;
        }
        void printSize()
        {
```

```
                    cout<<"矩形，四条边分别为：";
                    cout<<"A: "<< sizeA << ", b: "<< sizeB << ", C: "<< sizeC << ", D: "<<
sizeD << endl;
                }

};

void main()
{
/*    Line *l = new Line();
      cout<<"线段长度为:"<< l->getLength() << endl;
*/
      /*Triangle *t = new Triangle();
      t->printSize();*/

      Rectangle *r = new Rectangle();
      r->printSize();
}
```

[color=#FF0000]第七章 [/color]

一、单项选择

1.A; 2.A; 3.B; 4.D; 5.D

二、填空题

1.rbegin(), insert(iterator it, const T&)；

2.size(), 2;

3.typedef vector< 数据类型 >::reverse_iterator reverse_iterator

三、改错题

1.第六行的返回类型

```
      T getx()
      {
            return x;
      }
```

2.类 Point 的构造方法中的参数类型是 int，所以在 Line 构造方法中的 a，b 应该是 int 型；

四、编程题

1.使用类模板演示复制兼容性规则。

```
#include < iostream>
```

```
using namespace std;
template
class Point
{
	protected:
		T x, y;
	public	:
		Point(T a, T b)
		{
			x = a;
			y = b;
		}
		void show()
		{
			cout<<"x = "<< x <<", y = "<< y << endl;
		}
};

template
class Rectangle : public Point
{
	private:
		T h, w;
	public :
		Rectangle(T a, T b, T c, T d) : Point(a, b)
		{
			h = c;
			w = d;
		}
		void show()
		{
			cout<<"x = "<< x <<", y = "<< y <<"; h = "<< h <<", w = "<< w << endl;
		}
};
void main()
{
	Point	a(3, 4);
	Rectangle	b(5.1, 6.2, 7.3, 8.4);
	a.show();
	b.show();

	Point	& ra = b;//子类对象  初始化父类的引用
	ra.show();
```

```
    Point   * p = &b;//子类对象的地址，赋给指向父类的指针
    p->show();

    Rectangle   * pb = &b;//子类指针 pb
    pb->show();

    a = b;                          //派生类对象的属性值，更新父类对象的属性值
    a.show();
}
//134 页，例 6.3  赋值兼容规则的例子
```

2.设计一个点的类模板，分别使用继承、包含的方法设计线段类模板，要求演示构造函数和复制构造函数的设计方法，并用主程序验证之。

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

template   class Point
{
    public    :
              T x, y;
              Point(T a=0, T b=0)
              {
                  x = a;
                  y = b;
              }
              void show()
              {
                  cout<<"x = "<< x <<", y = "<< y << endl;
              }
};


template   class Line_1 : public Point //  继承 Point 类模板, 的线段类模板
{
    protected:
              T x1, y1;
    public :
              Line_1(T a, T b, T c, T d) : Point(a, b)
              {
                  x1 = c;
                  y1 = d;
              }
```

```
        Line_1(const Line_1 & );//复制构造函数
        void show()
        {
            cout<<"("<< x <<", "<< y <<"); ("<< x1 <<", "<< y1 <<")"<< endl;
        }
};
template   Line_1   :: Line_1(const Line_1 & t) : Point(t.x, t.y)
{
    x1 = t.x1;
    y1 = t.y1;
}

template   class Line_2 //包含 point 类模板，的线段类
{
  protected:
        Point   p1, p2;
  public    :
        Line_2(T a, T b, T c, T d)
        {
            p1.x = a;
            p1.y = b;
            p2.x = c;
            p2.y = d;
        }
        Line_2(const Line_2   &);//复制构造函数
        void show()
        {
            cout<<"("<< p1.x <<", "<< p1.y <<"); ("<< p2.x <<", "<< p2.y <<")"<< endl;
        }
};

template   Line_2   :: Line_2(const Line_2 & t)
{
  p1 = t.p1;
  p2 = t.p2;
}
void main()
{
  Line_1   L1(1,2,3,4);
  cout<<"L1 : ";L1.show();

  Line_1   L2(L1); //用现有的对象，初始化新对象
  cout<<"L2 : ";L2.show();
```

```
    Line_2    J1(5,6,7,8);
    cout<<"J1 : ";J1.show();

    Line_2    J2(J1);
    cout<<"J2 : ";J2.show();

}
```

3.已知有一个整型数组 a，其内容为 1 3 5 7 9 2 4 6 8 10.先对数组进行升序排列，再使用它产生向量 b，然后再在向量的尾部追加 11，并按降序排列输出向量的内容和 capacity()的内容。

```
#include < iostream >
#include < vector >
#include < algorithm >
using namespace std;
void main()
{
    int a[] = {1,3,5,7,9,2,4,6,8,10};
    sort(a, a+10);//先对数组进行升序排序
    copy(a, a+10, ostream_iterator(cout," "));
    cout<< endl;
    vector    pa(a, a+10); //再声明向量

    pa.push_back(11);//向量尾部追加 11
    reverse_copy(pa.begin(), pa.end(), ostream_iterator(cout," "));//按降序输出向量的内容

    cout<<"\ncapacity : "<< pa.capacity() << endl;//输出 capacity()的内容
}
```

[color=#F70909]第八章[/color]

一、单项选择题

1.A; 2.B;

二、分析程序题

1.

2.

三、查错题

print 函数的参数应该是引用

```
void print(base & p)
{
    p.show();
}
```

四、完成程序题

```
#include < iostream >
using namespace std;
class base
{
    int i;
    int j;
    public:
        base(int I, int J) : i(I),j(J)
        {
            display();
        }
        int getI() const
        {
            return i;
        }
        int getJ() const
        {
            return j;
        }
        void display() const
        {
            cout<<"i = "<< i <<'\t'<<"j = "<< j << endl;
        }

};

class derived : public base
{
    int k;
    public:
        derived(int I, int J, int K) : base(I, J), k(K)
        {
            display();
        }
```

```
        void display() const
        {
            cout<<"i = "<< getI() <<'\t'<<"j = "<< getJ() <<'\t'<<"k = "<< k << endl;
            cout<<"i + k = "<< (getI()+k) <<'\t'<<"j + k = "<< (getJ()+k) << endl;
        }
};

void main()
{
    base b3(8, 9);
    derived d1(10, 20, 5);
}
```

[color=#FF0000]
第九章[/color]
一、单项选择

1.B; 2.A; 3.C; 4.B 5.D 6.B

二、填空题

1.输出数据按输出域右边对齐输出

2.cin.ignore(3)

3.ofstream fout("Text.txt");

三、分析程序题

1.

2.

四、完成程序题

1.

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

void main()
{
```

```
cout.precision(6);
cout<< scientific << showpos;
cout<< -25.89F <<" ";
cout<< 25.89f << endl;
}
```

2.

```
class Fun
{
    friend ostream & operator<<(ostream & os, Fun f)
    {
        os.setf(ios::left);
        return os;;
    }
};

void main()
{
    Fun fun;
    cout<< setfill('*') << setw(10) <<12345<<" ";
    cout<< fun << setw(10) << 54321 << endl;
}
```

五、编程题

1.利用流格式控制，进行成绩和名字的输出，要求名字左对齐，分数右对齐。

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

class Student
{
    private :
            string name;
            float score;
    public :
            Student(){}
            Student(string n, float s)
            {
                name = n;
                score = s;
            }
```

```
        string getName()
        {
            return name;
        }
        float getScore()
        {
            return score;
        }
};
void main()
{
    Student s1("liming", 98);
    Student s2("sdfh", 90);
    Student s3("vn.fy", 80);
    Student s4("cnbtrt", 70);
    Student s5("ryuety", 48);
    cout.width(15);    cout<< left <<"姓名"<< right <<"分数"<< endl;
    cout.width(15);    cout<< left << s1.getName() << right << s1.getScore() << endl;
    cout.width(15);    cout<< left << s2.getName() << right << s2.getScore() << endl;
    cout.width(15);    cout<< left << s3.getName() << right << s3.getScore() << endl;
    cout.width(15);    cout<< left << s4.getName() << right << s4.getScore() << endl;
    cout.width(15);    cout<< left << s5.getName() << right << s5.getScore() << endl;
}
```

2.编写一个产生文本文件的程序。

```
#include < iostream>
#include < fstream >
using namespace std;
void main()
{
    char *p = {"C++程序设计"};
    ofstream myFile("Worl9_5_2.txt");
    myFile<< p;
}
```

3.编写一个程序，要求输入三角形的 3 条边，然后判断是否合理，如果不合理，给出信息并要求重新输入；如果合理，计算其面积并将结果存入文件中。

```
//我调试这个程序的时候，发现必须关掉卡巴斯基才可以，不知道为什么
#include < iostream >
#include < fstream >
#include < cmath >
#include < vector >
#include < iomanip >
```

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

class Triangle
{
	double sizeA, sizeB, sizeC, area;
	public:
		Triangle(){}
		void setArea()
		{
			double p = (sizeA + sizeB + sizeC) *0.5;
			area = sqrt( p * (p - sizeA) * (p - sizeB) * (p - sizeC) );
		}
		void setSizeA(double a)
		{
			sizeA = a;
		}
		void setSizeB(double b)
		{
			sizeB = b;
		}
		void setSizeC(double c)
		{
			sizeC = c;
		}
		void set(vector    &);
};
//****************************************
//* 成员函数：set
//* 参 数    ：向量对象的引用
//* 返回值   ：无
//* 功能     ：为向量赋值并将向量存入文件
//****************************************
void Triangle :: set(vector    & v )
{
	Triangle t;
	double a, b, c;
	while(1)
	{
		cout<<"三角形，边 A：";
		cin>>a;

		if(a == -1)//结束符为-1
		{
```

```
                ofstream writeFile;
                char fileName[20];
                cout<<"输入要保存到的文件名：";
                cin>>fileName;
                cout<<"保存到文件："<< fileName << endl;
                writeFile.open(fileName);
                if(writeFile.fail())
                {
                    cout<<"没有正确建立文件！"<< endl;
                    return;
                }
                for(int i=0; i< v.size(); i++)
                    writeFile<< v[i].sizeA <<" "<< v[i].sizeB <<" "<< v[i].sizeC <<" "<<
v[i].area << endl;
                writeFile.close();
                cout<<"一共写入"<< v.size() <<"个三角形信息"<< endl;
                return;
            }
            cout<<"三角形，边 B：";
            cin>> b;
            cout<<"三角形，边 C：";
            cin>> c;
            if( a>0 && b>0 && c>0 && a+b>c && a+c>b && b+c>a )
            {

                t.setSizeA(a);
                t.setSizeB(b);
                t.setSizeC(c);
                t.setArea();
                v.push_back(t);
            }
            else
                cout<<"不能组成三角形，重新输入"<< endl;

        }
}
void main()
{
    vector    tri;
    Triangle triangle;
    triangle.set(tri);
}
```

4.改写上题的程序，使程序反复计算，直到输入结束符号为止。要求在停止计算后，询问要

保存的文件名，然后讲结果一次写入制定文件中。

```
//需要关掉卡巴斯基
#include < iostream >
#include < fstream >
#include < cmath >
#include < vector>
#include < iomanip >
#include < string >
using namespace std;

class Triangle
{
      double sizeA, sizeB, sizeC, area;
      public:
            Triangle(){}
            void setArea()
            {
                  double p = (sizeA + sizeB + sizeC) *0.5;
                  area = sqrt( p * (p - sizeA) * (p - sizeB) * (p - sizeC) );
            }
            void setSizeA(double a)
            {
                  sizeA = a;
            }
            void setSizeB(double b)
            {
                  sizeB = b;
            }
            void setSizeC(double c)
            {
                  sizeC = c;
            }
            void set(vector    &);
};
//****************************************
//* 成员函数：set
//* 参 数    ：向量对象的引用
//* 返回值   ：无
//* 功能     ：为向量赋值并将向量存入文件
//****************************************
void Triangle :: set(vector    & v )
{
      Triangle t;
```

```
        double a, b, c;
        while(1)
        {
            cout<<"三角形，边 A：";
            cin>>a;

            if(a == -1)//结束符为-1
            {
                ofstream writeFile;
                char fileName[20];
                cout<<"输入要保存到的文件名：";
                cin>> fileName;
                cout<<"保存到文件："<< fileName << endl;
                writeFile.open(fileName);
                if(writeFile.fail())
                {
                    cout<<"没有正确建立文件！"<< endl;
                    return;
                }
                for(int i=0; i< v.size(); i++)
                    writeFile<< v[i].sizeA <<" "<< v[i].sizeB <<" "<< v[i].sizeC <<" "<<
v[i].area << endl;
                writeFile.close();
                cout<<"一共写入"<< v.size()<<"个三角形信息"<< endl;
                return;
            }
            cout<<"三角形，边 B：";
            cin>>b;
            cout<<"三角形，边 C：";
            cin>>c;
            if( a>0 && b>0 && c>0 && a+b>c && a+c>b && b+c>a )
            {

                t.setSizeA(a);
                t.setSizeB(b);
                t.setSizeC(c);
                t.setArea();
                v.push_back(t);
            }
            else
                cout<<"不能组成三角形，重新输入"<< endl;

        }
}
```

```
void main()
{
    vector    tri;
    Triangle triangle;
    triangle.set(tri);
}
```

5.从文件 TEST 中读出字符并写入 TEST1 里，要求均附加错误检查。

```
#include
#include
using namespace std;

void main()
{
    ifstream txt1("TEST.txt");
    ofstream txt2("TEST1.txt");
    char c;
    if(!txt1)
    {
        cout<<"文件打不开!"<< endl;
        return;
    }
    if(!txt2)
    {
        cout<<"没有正确建立文件!"<< endl;
        return;
    }
    while(1)
    {
        txt1>>c;
        if(txt1.eof())
        {
            txt1.close;
            return;
        }
        cout<< c;//打印字符
        txt2<< c;//写文件 TEST1.txt 中
    }
}
```

6.从键盘输入一个字符串，将其中的大写字母全部转换成小写字母，然后存入到文件名为“text”的磁盘文件中保存。输入的字符串以“$”结束。

```
//需要关掉卡巴斯基
#include < iostream >
#include < fstream >
using namespace std;
void main()
{
	char a[100];
	ofstream writeFile("text.txt");
	int i;
	while(1)
	{
		cin>>a;
		if(a[0] == '$')
			return;
		i = 0;
		while(a[i] != '\0')
		{
			if( a[i]>=65 && a[i]<=90 )
				a[i]=a[i] + 32;
			i++;
		}
		writeFile<< a<<" ";
	}
}
```

[color=#FF0000]第十章[/color]

一、单项选择题

1.D; 2.A; 3.B; 4.D

二、填空题

1.过程抽象和数据抽象

2.对象

3.问题域、系统边界、系统责任

4.我觉得应该是，类的成员有（数据成员）和（成员函数）两打类。

5.

四、编程题

1.取消设计实例中的 Cow 属性，练习使用模板实现包含的设计方法。

```
#include   < iostream >
#include < cmath >
using namespace std;
//********************
//*  声明 Point 类
//********************
template <class T>
class Point
{
      T x, y;
      public:
            Point(T a= 0, T b= 0) : x(a), y(b){}
            Point(Point & a)
            {
                  x = a.x; y = a.y;
            }
            void Display();
            T Distance(Point &);
            T getX(){return x;}
            T getY(){return y;}
};



//***************
//*成员函数：Point :: Display()
//*功能      ：打印点坐标
template
void Point :: Display()
{
      cout<< x <<"，"<< y << endl;
}

//**************************
//*成员函数：Point :: Distance
//*参数      ：Point 对象的引用
//*返回值   ：两点间距离
//*返回类型：T
template
T Point :: Distance(Point & a)
{
      return sqrt( (x - a.x)*(x - a.x) + (y - a.y) * (y - a.y) );
```

```
}


//**********************
//*     声明 Line
//**********************
template
class Line
{
      Point a, b;
      public:
            Line(Point & a1, Point    & a2) : a(a1), b(a2){}
            Line(Line & s) : a(s.a), b(s.b){}
            void Display();
            T Distance();
            T getArea();
};

//*************************
//*成员函数：Line::Dispaly
//*参数       ：无
//*功能       ：打印线段每个点的坐标
template
void Line :: Display()
{
      a.Display();
      b.Display();
}

//*************************
//*成员函数：Line::Distance
//*参数       ：无
//*返回值    ：线段长度
template
T Line :: Distance()
{
      T x = a.getX() - b.getX();
      T y = a.getY() - b.getY();
      return sqrt( x * x + y * y );
}


void main()
{
```

```
    Point a;
    Point b(7.8, 9.8);
    Point c(34.5, 67.8);
    a = c;
    a.Display();
    b.Display();
    cout<<"两点之距离："<< a.Distance(b) << endl;

    Line s(a, b);
    Line s1(s);
    cout<<"线段属性如下："<< endl;
    s1.Display();
    cout<<"线段长度："<< s1.Distance()<< endl;
}
```

2.取消设计实例中的 Cow 属性，练习使用模板实现继承的设计方法。

```
#include < iostream >
#include < cmath >
using namespace std;
//**********************
//*声明 Point 类
//**********************
template
class Point
{
    protected:
        T x, y;
    public:
        Point(T a = 0, T b = 0): x(a), y(b){}
        Point(Point &a)
        {
            x = a.x; y = a.y;
        }
        virtual void Display()
        {
            cout<<"X = "<< x <<", Y = "<< y << endl;
        }
        T Distance(Point &);
        T   getX(){return x;}
        T    getY(){return y;}
};
//*****************************
//*成员函数: Point::Distance
//*参数       : Point 对象的引用
```

```
//*返回值   : 两点间距离
template
T Point :: Distance(Point &a)
{
    return sqrt( (x - a.x) * (x - a.x) + (y - a.y) * (y - a.y) );
}


//************************
//*声明 Line 类
//************************
template
class Line : public Point
{
    double x2, y2;
    public:
        Line(T, T, T, T );
        Line(Line & );
        void Display();
        T Distance();
        T getX2(){return x2;}
        T getY2(){return y2;}
        friend void Disp(Line & t){cout<<t;)
Line :: Line(T a1, T a2, T a3, T a4):Point(a1, a2), x2(a3), y2(a4){}
template
Line :: Line(Line &s) : Point(s.x, s.y), x2(s.x2), y2(s.y2){}

//*****************************
//*成员函数：Line::Distance()
//*返回值   ：线段长度
template
T Line::Distance()
{
    T x = x2 - x;
    T y = y2 - y;
    return sqrt(x*x + y*y);
}

//*****************************
//*成员函数：Line::Display()
//*功能      ：打印线段两个端点坐标
template
void Line::Display()
{
```

```
    cout<<"x = "<< x <<"，y = "<< y <<"，x2 = "<< x2 <<"，y2 = "<< y2 << endl;
}

//*****************************
//*友元函数：operator<<
//*返回值  ：ostream &
//*功能    ：重载“<< ”
template
ostream & operator<<(ostream & stream, Line obj)
{
    stream<<"使用重载 << ，输出线段属性如下："<< endl;
    stream<< obj.getX() <<"，"<< obj.getY() <<"，"
          << obj.getX2() <<"，"<< obj.getY2() << endl;
    return stream;
}



void main()
{
    Pointa;
    Pointb(7.8, 9.8);
    Pointc(34.5, 67.8);
    a = c;
    a.Display();
    b.Display();
    cout<<"两点距离："<< a.Distance(b) << endl;

    Line s(7.8, 9.8, 34.5, 67.8);
    Disp(s);
    Line s1(s);
    cout<<"使用 Display 函数输出线段属性如下："<< endl;
    s1.Display();
    cout<<"线段长度："<< s1.Distance() << endl;
    cout<<"基类对象的属性"<< endl;
    a.Display();

    a = s;//派生类的对象可以赋给基类
    cout<<"派生类的对象赋给基类对象"<< endl;
    a.Display();
    cout<<"派生类的对象赋给基类的指针"<< endl;
    Point *p = &s1;
    p->Display();
    cout<<"基类对象引用派生类的对象"<< endl;
```

```
    Point &d = s1;
    d.Display();
}
```

待续，每天都会更新。