学号____________ 姓名____________ 院（部）____________ 专业____________ 考试时间：2008年__月__日

密 封 线

《c++》编程题试卷

第三章

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 <= 122)
                    return (c - 32) ;
          else
                    return c;
}
void main()
{
          int i;
          char c[15] = {'A','v','e','t','E','T','%','&','4','Y','e','i','@','9','ˆ'};
          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;
}
```

第四章

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)                    {
                                        x = a;
                                        y = b;
                              }
                    void setXY(int a, int b)                         {
                                        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;
  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)
                              {
                                        point1 = one;
                                        point4 = two;
                                        init();
                              }
          Rectangle(int x1, int y1, int x2, int y2)
                              {
                              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;

}
```

第五章

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();
}
```

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;
}
```

第六章

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();
}
```

第七章

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();
}
```

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
}
```

第九章

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。

```
#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。

```
#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<<" ";
        }
}
```