E01. **[标准输入输出流，string 亦可使用]**从键盘输入一个不包括空格的仅由英文字母组成的字符串，长度不超过 200 个字符。统计 26 个英文字母的使用频率（出现的次数），不区分大小写。最后按使用频率从大到小的顺序输出字母（小写字母）及其使用频率，没有出现的字母不输出。

E02. **[文件输入输出流]**手工在程序所在目录下创建 1 个文本文件 Data.txt，该文件中存放了 9 个 int 数据，各数据之间以空格间隔。编写 1 个程序，使用文件流的相关方法顺序完成以下操作：

a) 读取文件 Data.txt 中的所有数据；

b) 将读取得到的数据以二进制的形式存放到程序所在目录下的文件 Res.dat 中；

c) 再次读取 Res.dat 中的所有数据；

d) 并将这些数据以文本文件的形式存放到程序所在目录下的 Res.txt 文件中，存放时每行 2 个数据，每个数据占 10 列，右对齐，左边补'C'字符。

E03. **[vector，string]**编写程序对 12 个月的英文单词根据字典序进行从小到大的排序，请使用 string 向量实现。

E04. **[vector，string，标准输入输出流，文件输入输出流]**按要求编写程序。

a) 定义一个结构体类型 Student，如下所示：

```
struct Student
{
        int ID;                 //学号
        string name;            //姓名
        float score[3];         //三门课程成绩
};
```

b) 编写一个函数，按照上述结构体定义，依次从键盘输入 5 个学生的信息，创建学生**向量** A；

c) 编写一个函数，将上述**向量** A 按照学号 ID 进行增序排序，并将排序后的学生信息显示在屏幕上，显示格式要求如下：

《学号》 10 列，右对齐	*《姓名》 10 列，右对齐*	*《课程 1》 3 位有效数字，右对齐*	*《课程 2》 3 位有效数字，右对齐*	*《课程 3》 3 位有效数字，右对齐*
111	**TOM**	**89.0**	**91.0**	**89.0**

122　　MIKE　　71.5　　93.5　　79.5

……

d) 编写一个函数，删除**向量**中姓名为“TOM”的学生信息，并将处理结束后的**向量**内容输出到 D 盘根目录中的“res.dat”中，格式和 c 中的要求相同；

e) 要求使用流的方法来完成程序流程中所有的数据输入和输出操作；要求使用向量来保存学生信息。

E05. **[vector]**已知 1 个向量，其中包含{35,46,57,13,24,35, 68,13,79,88,46}。请将向量中重复的元素去除，并按从小到大的顺序排列并输出。要求完全使用向量的相关功能实现。

E06. **[引用作为函数参数]**设计一个函数，完成如下功能。

a) 输入两个整数。

b) 输出最大公约数和最小公倍数。

c) 不使用指针。

d) 编写主函数，测试函数功能。

E07. **[函数重载]**设计函数 width，完成如下功能。

a) 输入某类型参数，输出该参数在屏幕上所占宽度。

b) 参数类型为字符串、字符或整数。

c) 用多个同名函数完成。

d) 编写主函数，测试上述函数的功能。

E08. **[第 8 章：类的基本操作]**设计一个类 Circle，表示圆形。

a) 以圆心坐标(x,y)和半径 r 来确定圆。

b) 可设置圆心坐标。

c) 可设置半径。

d) 可计算圆的面积。

e) 可计算圆的周长。

f) 编写主函数，测试类的功能。

E09. **[第 8 章：类的基本操作]**设计一个类 Column，表示圆柱体。设圆柱体底面在 Z=0 的平面内。

a) 以底面圆（参见上题）和高来确定圆柱体。

b) 可设置底面圆心。

c) 可设置底面半径。

d) 可设置高。

e) 可计算底面积。

f) 可计算底面的周长。

g) 可计算侧面积。

h) 可计算体积。

i) 编写主函数，测试类的功能。

E10. **[第 8 章：类的基本操作，运算符重载]**在 E08 圆形类的基础上，完成如下功能：

a) 定义加法运算，规则：两圆之和为一个新的圆，圆心是第一个操作数的圆心（如 a+b，则 a 的圆心为 a+b 的圆心），半径为两圆半径之和。加法运算不改变操作数。

b) 定义减法运算，规则：两圆之差为一个新的圆，圆心是第一个操作数的圆心，面积为两圆面积之差的绝对值。减法运算不改变操作数。

c) 定义自增、自减运算（含前、后自增），对半径进行自增、自减运算。

d) 定义输出流运算，输出圆心坐标、半径、周长、面积。

e) 定义>、<运算，比较两圆的面积之间的大小关系。

f) 定义==、!=运算，比较两圆是否是完全相同的圆。

g) 定义&运算，确定两圆是否同心。

h) 定义|运算，确定两圆的位置关系（相交、相切、相离、包含）。

i) 编写主函数，任意生成若干圆，分别测试上述功能。

j) 提示：可自行决定是否需要增加圆的属性。但是，不提倡为了简化函数的计算而无原则地增加属性。这样做在给某些计算带来方便的同时，也可能会使另外一些计算变复杂。如：在记录半径的情况下，再记录周长和面积，可以简化某些运算符重载函数的代码。但是，半径与周长、面积之间有确定的约束关系需

要遵守。所以，增加了面积和周长属性后，改变其中任意一项的值都意味着需要重新计算另外两项的值，否则就会产生数据的不一致。

E11. **[第 8 章：类的基本操作，运算符重载，友元函数]**在 E10 的圆形类的基础上（是否需要增减属性和方法可自行决定），完成如下功能：

a) 定义两个圆对象，确定它们的位置关系为相离。

b) 令两圆沿圆心连线作相向运动。两圆运动速度之比为面积之比的反比。设速度较慢的圆的运动速度为 1，即每走一步的距离为 1。

c) 两圆运动的同时不断地缩小面积。每走一步，半径缩小 1。

d) 用程序模拟上述过程（用循环来实现，每循环一次代表走一步），找出两圆的碰撞位置（即两圆的圆心距离小于等于两圆半径之和，两圆的圆心坐标和半径）。

e) 每走一步，输出两圆的位置、半径和距离（距离是指两圆相距最近的两个点之间的距离）。

f) 提示 1：考虑一下，两圆是否一定会碰撞？

g) 提示 2：在不影响被模拟问题本身性质的前提下，是否有办法简化计算？

E12. **[第 8 章：静态成员]**现需要处理银行活期存款业务，设账户类为，CAount，请根据如下需求实现该类，并在 main 函数中测试。

- 每个账户需要有一个浮点型的成员 m_Money 用于存储账上余额；
- 每个账户需要描述存款的日期；
- 银行的年利息采用浮点型静态数据成员 m_InterestRate 描述，从而避免为每个账户存储利息；
- 为年利息成员提供静态成员 SetInterestRate 进行设置；
- 为年利息成员提供静态成员 GetInterestRate 进行获取；
- 提供存款成员函数 SaveMoney；
- 提供取款成员函数 LendMoney；
- 提供计算利息函数 CalcInterest；
- 提供结算利息函数 SaveInterest，该函数将计算出的利息结算到本金中。
- **为简化计算，请考虑以下定义或限制：**

  1) 本题目不考虑闰年，每个月都认为 30 天，一年认为 360 天。

2) 存款仅考虑发生一次！
3) 取款允许发生多次，但取款是否允许需要考虑“本金是否足够”的条件

银行相关业务和利息的计算方法如下：

(1). 假设年利率 m_InterestRate=0.036（表示 3.6%）。m_InterestRate 是静态成员变量，按照静态成员变量的感念，对所有账户 CAount 类的对象而言 m_InterestRate 只有 1 个，这样才能实现一改全改的效果！

(2). 我于 2014-1-1 到银行存了 100000 元，m_Money=100000。

(3). 2014-3-10 银行给我“结算利息”一次。2014-1-1 到 2014-3-10 之间一共间隔了 70 天，本金 m_Money=100000+100000*0.036/360*70=100700。“结算利息”以后，存款日期变为了 2014-3-10！

(4). 2014-3-30 我到银行想取款 200000，由于本金 m_Money 只有 100700，所以不允许取款！

(5). 2014-4-4 我到银行想取款 50000，由于本金有 100700，所以允许取款。2014-3-10 到 2014-4-4 之间一共间隔了 25 天，本次取得的金额是 50000+50000*0.036/360*25=50125，本金 m_Money=100700-50000=50700（利息不从我的账户中扣除，这是由银行提供给我的回报），存款日期仍然维持为 2014-3-10

E13. **[第 8 章：运算符重载，友元函数]**设有描述复数的类 CComplex 定义如下：

```
class CComplex
{
    double m_real;
    double m_image;
public:
    void setValue(double real,double image)
    {
        m_real=real;
        m_image=image;
    }
};
```

请使用友元函数实现如下重载：

- 重载<<运算符，使得可以用 cout<<输出复数，每个复数输出的格式为：

  “实部+虚部*i”;

- 重载+运算符，使得可以实现两个复数相加；
- 重载+运算符，使得可以实现复数和实数的相加；
- 重载前置++运算符，使得可以实现复数的实部和虚部分别加 1；
- 重载后置++运算符，使得可以实现复数的实部和虚部分别加 1；
- 在 main 函数中测试并试用这些运算符。

E14. **[第 9 章：构造函数，拷贝构造，析构函数]**设有类定义如下：

```
#define MAXLENGTH 1000
class CString
{
    char m_buff[MAXLENGTH];
public:
    CString();                  //构造函数1，设置为空字符串
    CString(char *src);         //构造函数2，在考虑了src的长度是否合法后初始化当前对象
    CString(char ch);           //构造函数3，0号元素赋值为ch
    int setString(char *src); //设置字符串，在考虑了src的长度是否合法后设置当前对象
    friend ostream& operator<<(ostream& Out, const CString& S);      //输出字符串
};
```

请编写程序实现如下功能：

- 为其中的每个成员函数给出实现代码；
- 重载下标运算符[ ]，使得可以通过[ ]访问数组的指定元素；
- 自行根据需要决定是否需要添加拷贝构造函数和析构函数；
- main 函数要求如下，不允许修改：

```
void main()
{
  CString s1;
  cout<<s1;
  s1.setString("abc1");
  cout<<s1;

  CString s2("abc2");
  cout<<s2;

  CString s3('a');
  cout<<s3;
```

```
    CString s4(s2);
    cout<<s4;
}
```

E15. **[第 9 章：构造函数，拷贝构造，析构函数，浅拷贝和深拷贝]**设有类定义如下：

```
class CString
{
    char * m_pbuff;   //指向字符串的指针
    int  m_length;    //表示当前字符串的最大允许长度，也就是字符数组的长度
public:
    CString();   //构造函数1，设置为空字符串，m_length=100
    CString(char *src);  //构造函数2，在考虑了src的长度后初始化当前对象，
m_length>=src的长度+1
    CString(char ch);   //构造函数3，0号元素赋值为ch, m_length=100
    int SetString(char *src);     //设置字符串，在考虑了src的长度后设置当前对
象,m_length>=src的长度+1
    friend ostream& operator<<(ostream& Out, const CString& S);       //输出字
符串
};
```

请编写程序实现如下功能：

- 为其中的每个成员函数给出实现代码；
- 重载下标运算符[]，使得可以通过[]访问数组的指定元素；
- 自行根据需要决定是否需要添加拷贝构造函数和析构函数；
- main 函数要求如下，不允许修改：

```
void main()
{
    CString s1;
    cout<<s1;
    s1.setString("abc1");
    cout<<s1;

    CString s2("abc2");
    cout<<s2;

    CString s3('a');
    cout<<s3;

    CString s4(s2);
    cout<<s4;
}
```

E16. **[第 9 章：类成员，类成员的构造（冒号语法）]**下面的一段程序中的 Cline 类中缺少了部分成员函数，该程序的运行结果如下：

```
Point 1 is:(0,0)
Point 2 is:(0,0)
Length=0
Point 1 is:(1,1)
Point 2 is:(5,5)
Length=5.65685
```

请为 Cline 函数补充必要的成员函数与实现代码，使得程序正确运行。

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

class CPoint
{
    int m_x; //点的X坐标
    int m_y;     //点的Y坐标°
public:
    CPoint()
    {
        m_x=0;
        m_y=0;
    }
    CPoint (int x,int y)
    {
        m_x=x;
        m_y=y;
    }
    int getx()
    {
        return m_x;
    }
    int gety()
    {
        return m_y;
    }
    void showPoint()
    {
        cout<<"("<<this->m_x<<","<<this->m_y<<")"<<endl;
```

```
    }
};

class CLine
{
    CPoint m_point1;
    CPoint m_point2;
};

void main()
{
    CLine line1;
    line1.ShowLine();
    cout<<"Length="<<line1.distance()<<endl;

    CLine line2(1,1,5,5);
    line2.ShowLine();
    cout<<"Length="<<line2.distance()<<endl;
}
```

E17. **[第 9 章：类成员，类成员的构造（冒号语法）]**设有类 CTime 和 CDate 分别用于描述时间和日期，另外有 CDateTime 类描日期和时间，请为三个类给出具体的实现代码，并在 main 函数中测试。

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

class CTime
{
    int m_hour,m_mintue,m_second;
public:
    CTime(int hour=0,int minute=0,int second=0);
    //如果时间合法，则赋值，否则赋值 0 时 0 分 0 秒
    int SetTime(int hour=0,int minute=0,int second=0);
    //如果时间合法，则赋值并返回 1，否则不赋值，并返回 0
    int getHour();
    int getMinute();
    int getSecond();
    void ShowTime(bool flag);
    //flag 为 True 则以 24 小时制显示时间，否则以 AM PM 的方式显示
    //自己考虑是否需要添加其他成员函数
```

```
};

class CDate
{
    int m_year,m_month,m_day;
public:
    CDate(int year=2000,int month=1,int day=1);
    //如果日期合法，则赋值，否则赋值 2000 年 1 月 1 日
    int SetDate(int year=2000,int month=1,int day=1);
    //如果日期合法，则赋值并返回 1，否则不赋值，并返回 0
    int GetYear();
    int GetMonth();
    int GetDay();
    void ShowDate(bool flag);
    //flag 为 TRUE，则以中文的方式显示日期，否则以 MM/DD/YYYY 的方式显示日期
    //自己考虑是否需要添加其他成员函数
};

class CDateTime
{
    CTime m_time;
    CDate m_date;
public:
    //添加必要的构造函数
    //int SetDateTime(...); 自己设计参数，考虑该函数的返回值加以表示全部正确、日期不对、时间不对等情况
    //void ShowDateTime(...); 自己设计参数，考虑可以中文日期和西文日期格式，以及 24 小时和 AM PM 都可以控制
    //添加自己认为必要的其他成员函数
};
```

在主函数中验证上述类的功能，要求的主函数如下所示，不允许进行修改：

```
void main()
{
    CDateTime d1(2014, 5, 2, 27, 12, 5);
    d1.ShowDateTime(false, true);       //第1个参数表示以英文方式显示日期, 第2个参数表示以24小时制方式显示时间

    CDateTime d2;
    d2.ShowDateTime(true, false);       //第1个参数表示以中文方式显示日期, 第2个参数表示以AM PM的方式显示时间

    int iStatus;
```

```
    iStatus=d2.SetDateTime(2014, 5, 2, 21, 55, 5);
    switch(iStatus)
    {
    case 1:
        cout<<"日期和时间数据设置正确!"<<endl;
        break;
    case -1:
        cout<<"日期数据设置不正确!"<<endl;
        break;
    case -2:
        cout<<"时间数据设置不正确!"<<endl;
        break;
    }
    d2.ShowDateTime(false, false);      //第1个参数表示以英文方式显示日期，第2个参数表示以AM PM的方式显示时间

    cout<<"月="<<d2.GetMonth()<<endl;              //认真考虑一下如何实现此操作？
    cout<<"分钟="<<d2.GetMinute()<<endl;           //认真考虑一下如何实现此操作？
}
```

E18. **[第 9 章：类成员，类成员的构造（冒号语法）]**有一个类的定义如下：

```
class  CBook
{
private:
    string name;
    string author;
    double price;
    string publisher;
public:
    CBook(){name = "无"; author="无"; price = 0.0; publisher = "无";}
    CBook(const CBook &);       // 拷贝构造函数
    CBook(string , string, double, string);     // 带参数的构造函数
    CBook(char*, char*, double, char*);         // 带参数的构造函数
    ～CBook();                  // 析构函数
    void SetName(char*);        // 设置书名的成员函数
    void SetName(string&);   // 设置书名的成员函数
    void print()const;          // 在屏幕上显示书的信息的成员函数
    friend ostream & operator<<(ostream& out, const CBook &);   // 输出操作符的重载函数
};
```

按要求完善 CBook 类的定义

(1) 完善 CBook 类的所有成员函数定义。

(2) 其中输出格式要求如下：

书名：

作者：

价格：

出版社：

(3) 析构函数中要求输出如下信息：

a) “书名”对象被析构了!

b) 输出时，上面 a 步骤中要求的书名用对象的 name 属性值替换

(4) 在主函数中验证上述类的功能，要求的主函数如下所示，不允许进行修改：

```
void main()
{
 string n="C++程序设计", a="王涛", pub="苏州大学出版社";

 CBook b1;
 cout<<b1<<endl;
 b1.SetName(n);
 b1.print();
 b1.SetName("VB");
 cout<<b1<<endl;

 CBook b2=b1;
 cout<<b2<<endl;

 CBook b3(n,a,35.0,pub);
 cout<<b3<<endl;

 CBook *b4=new CBook("VC++", "李国", 45.0, "清华大学出版社");
 cout<<*b4<<endl;
 delete b4;
}
```

E19. **[第 9 章：构造函数，拷贝构造，析构函数，运算符重载]**有一个类定义如下：

```
class CRmb
{
private:
    int yuan;
```

```
    int jiao;
    int fen;
public:
    CRmb(){yuan=0; jiao=0; fen=0;}
    CRmb(int, int, int);     // 带参数的构造函数
    CRmb(const CRmb &);          // 拷贝构造函数
    CRmb(double);            // 类型转换构造函数,将一个实型数据转换成人民币对象
     ~CRmb();                        // 析构函数
     friend ostream & operator<<(ostream& out, const CRmb &);     // 在屏幕上输出Rmb的值，输出格式如下：
    //   现在有：  元   角   分

    //======如下函数自己设计原型==========
    //(1)前自增运算符重载函数，要求实现对分自增1，并且考虑进位问题
    //(2)后自增运算符重载函数，要求实现对分自增1，并且考虑进位问题
    //(3)重载“+”运算符，要求实现下面两种加法运算
    //          a、CRmb对象+CRmb对象
    //          b、CRmb对象+double对象
};
```

在上述基础上，按要求完善类的定义：

(1) 完善类的所有成员函数定义

(2) 在析构函数中输出：“析构函数被调用!”

(3) 提供以下类的完整测试程序，不允许进行修改。

```
void fn(CRmb& x)
{
  cout<<"In fn: "<<x<<endl;
}

void main()
{
  CRmb m1;
  cout<<m1<<endl<<endl<<endl;

  CRmb m2(1001, 9, 9);
  m2++;
  cout<<m2<<endl;
  ++m2;
  cout<<m2<<endl<<endl<<endl;

  CRmb m3(m2);
  cout<<m3<<endl<<endl<<endl;
```

```
    CRmb m4=m1+m2;
    cout<<m4<<endl;
    m4=m1+50.0;
    cout<<m4<<endl<<endl<<endl;

    fn(50.13);
}
```

E20. **[第 9 章：构造函数，拷贝构造，析构函数，赋值运算符重载]**有一个类定义如下：

```
class CContry
{
private:
    char *name;          // 国家名称
    char *caption;     // 首都名称
    double   area;       // 国家面积，单位万平方公里
    double   person_num;   // 人口数量，单位 万
public:
    CContry()                       // 无参构造函数
    {
        name = new char[100];
        strcpy(name, "中国");
        caption = new char[100];
        strcpy(name, "北京");
        area = 960;
        person_num = 130000.00;
    }
    CContry(const CContry&);   // 拷贝构造函数
    CContry(char*, char* , double, double);     // 带参数的构造函数
    CContry& operator=(const CContry&);
    ～CContry();                                // 析构函数
    void set(char*,char*, double, double);        // 设置属性的成员函数
    void print()const;                // 在屏幕上输出CContry对象的信息
};
```

按要求完善类的定义：

(1) 完成类的所有成员函数定义

(2) 输出函数的输出格式如下：

a) 国家名称：

b) 首都名称：

c) 面积：

d) 人口数量：

(3) 在析构函数中输出如下信息：

a) “国家”对象被析构了！

b) 在输出时，a中的”国家”用对象的name属性值替换

(4) 自己提供类的完整测试程序，要求界面友好，输出结果应该有相应的提示信息。

(5) 要求在测试程序中new一个国家对象，并设置该对象的属性如下：

a) 国家名称： 日本

b) 首都名称： 东京

c) 面积： 37.835

d) 人口数量：12665.9683

(6) 要求在测试程序中delete日本对象

E21. **[第 10 章：继承，派生类的构造]**有一个 person 类定义如下：

```
class CPerson
{
private:
    string   m_name;          // 姓名
    int      m_age;             // 年龄
    char     m_sex;            // 性别   ‘M’表示男性   ‘F’表示女性
public:
      CPerson(string& name, int   age, char sex)
      {
        m_name = name;
        m_age = age;
        m_sex = sex;
      }
      CPerson()
      {
        m_name = “无名”;
        m_age = 18;
        m_sex = ‘M’;
      }
      void   print()
      {
         cout<<”\n 姓名：”<<m_name<<”\n 年龄：”<<m_age<<”\n”;
         if (m_sex == ‘M’)
             cout<<”性别：  男”<<endl;
         else
             cout<<”性别：  女”<<endl;
      }
```

};

请以 CPerson 类为基类定义一个派生类 CStudent,要求该类具有以下属性成员和成员函数：

(1) 学生所属大学名称，string 类型表示

(2) 学生所在年级，int 类型表示

(3) CStudent(); // 以{"无名"、18 岁、男性、"苏州大学"、1 年级}为默认值的无参构造函数；

(4) CStudent(string& name, int , char, string& collageName, int grade); // 带参数的构造函数

(5) void print(); // 显示学生类对象的所有信息

编写对 CStudent 类的测试程序，要求如下：

(1) 定义一个学生类 student1，属性值为默认属性值，并输出其信息

(2) 定义一个学生类 student2，其属性为：

a) 姓名："Liming"

b) 年龄：21

c) 性别：男

d) 大学：苏州大学

e) 年级：1

(3) 提供 CStudent 类的完整测试程序，要求界面友好，输出结果应该有相应的提示信息。

E22. **[第 10 章：继承，派生类的构造]**以前面第 18 题中 CBook 类为基础定义一个派生类 CComputerBook，该派生类有如下属性成员和成员函数：

(1) 属性成员：

a) 领域属性（string 类型）：如人工智能领域、嵌入式领域、机器学习领域等

b) 类别属性（string 类型）：如教材、实验指导书、练习册等

(2) 成员函数：

a) 无参构造函数。其中 name="无"; author="无"; price=0.0; publisher="无"; 领域属性为：无；类别为：无

b) 带参数的构造函数

c) 析构函数，析构函数中要求输出："CComputerBook 类对象被析构了!"

d) 流输出符重载函数，显示时要求每个属性信息占一行

(3) 提供类的完整测试程序，要求界面友好，输出结果应该有相应的提示信息。

(4) 在测试程序中 new 一个 CComputerBook 对象，并设置其属性如下：

a) 书名为：“C 程序设计教程”

b) 作者：“王涛”

c) 价格：35.0

d) 出版社：“清华大学出版社”

e) 领域：“程序设计”

f) 类别：“教材”

(5) 在测试层序中 delete 第(4)步创建的对象，体会析构函数的运行过程。