**第四届“蓝桥杯”全国软件专业人才设计与创业大赛选拔赛**C/C++本科B组

<题目在后面>

考生须知：

考试开始后，选手首先下载题目到本机不被还原的盘符，并使用考场现场公布的解压密码解压试题。

考试时间为9:00-13:00，共4个小时。**13点整时间截止时，将无法提交答案。选手因考试结束不能提交答案的，责任由选手自负。**

在考试强制结束前，选手可以主动结束考试（需要身份验证），结束考试后将无法继续提交或浏览答案。

选手可浏览自己已经提交的答案。

**对同一题目，选手可多次提交答案，以最后一次提交的答案为准。**

选手切勿在提交的代码中书写“姓名”、“考号”，“院校名”等与身份有关的信息或其它与竞赛题目无关的内容，否则成绩无效。

选手必须通过浏览器方式提交自己的答案。选手在其它位置的作答或其它方式提交的答案无效。

试题包含三种类型：“结果填空”、“代码填空”与“程序设计”。

**结果填空题：**要求选手根据题目描述直接填写结果。求解方式不限。不要求源代码。

把结果填空的答案直接通过网页提交即可，不要书写多余的内容。

**代码填空题：**要求选手在弄清给定代码工作原理的基础上填写缺失的部分，使得程序逻辑正确、完整。所填写的代码不超过一条语句（即中间不能出现分号）。

把代码填空的答案（仅填空处的答案，不包括题面已存在的代码或符号）直接通过网页提交即可，不要书写多余的内容。

注意选择自己使用的编译环境。

使用ANSI C/ANSI C++ 标准，不要依赖操作系统或编译器提供的特殊函数。

**程序设计题目：**要求选手设计的程序对于给定的输入能给出正确的输出结果。考生的程序只有能运行出正确结果才有机会得分。

注意：在评卷时使用的输入数据与试卷中给出的示例数据可能是不同的。选手的程序必须是通用的，不能只对试卷中给定的数据有效。

对于编程题目，要求选手给出的解答完全符合ANSI C++标准，不能使用诸如绘图、Win32API、中断调用、硬件操作或与操作系统相关的API。

代码中允许使用STL类库，但不能使用MFC或ATL等非ANSI C++标准的类库。例如，不能使用CString类型（属于MFC类库）。

注意: main函数需要返回0

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

所有源码必须在同一文件中。调试通过后，拷贝提交。

提交时，注意选择所期望的编译器类型。

结果填空 (满分4分)

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，不限解决问题的方式，只要求提交结果。

必须通过浏览器提交答案。

结果填空 (满分5分)

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，不限解决问题的方式，只要求提交结果。

必须通过浏览器提交答案。

结果填空 (满分8分)

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，不限解决问题的方式，只要求提交结果。

必须通过浏览器提交答案。

结果填空 (满分12分)

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，不限解决问题的方式，只要求提交结果。

必须通过浏览器提交答案。

代码填空 (满分5分)

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，不限解决问题的方式。

只要求填写缺失的代码部分，千万不要画蛇添足，填写多余的已有代码或符号。

必须通过浏览器提交答案。

注意选择自己使用的编译环境。

代码填空 (满分7分)

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，不限解决问题的方式。

只要求填写缺失的代码部分，千万不要画蛇添足，填写多余的已有代码或符号。

必须通过浏览器提交答案。

注意选择自己使用的编译环境。

程序设计（满分4分）

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，必须通过编程的方式解决问题。

注意：在评卷时使用的输入数据与试卷中给出的示例数据可能是不同的。选手的程序必须是通用的，不能只对试卷中给定的数据有效。

仔细阅读程序的输入、输出要求，千万不要输出没有要求的、多余的内容，例如：“请您输入xx数据：”。

建议仔细阅读示例，不要想当然！

程序处理完一个用例的数据后，立即退出（return 0），千万不要循环等待下一个用例的输入。

程序必须使用标准输入、标准输出，以便于机器评卷时重定向。

对于编程题目，要求选手给出的解答完全符合ANSI C++标准，不能使用诸如绘图、Win32API、中断调用、硬件操作或与操作系统相关的API。

代码中允许使用STL类库，但不能使用MFC或ATL等非ANSI C++标准的类库。例如，不能使用CString类型（属于MFC类库）。

注意: main函数结尾需要return 0

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

提交时，注意选择所期望的编译器类型。

程序设计（满分10分）

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，必须通过编程的方式解决问题。

注意：在评卷时使用的输入数据与试卷中给出的示例数据可能是不同的。选手的程序必须是通用的，不能只对试卷中给定的数据有效。

仔细阅读程序的输入、输出要求，千万不要输出没有要求的、多余的内容，例如：“请您输入xx数据：”。

建议仔细阅读示例，不要想当然！

程序处理完一个用例的数据后，立即退出（return 0），千万不要循环等待下一个用例的输入。

程序必须使用标准输入、标准输出，以便于机器评卷时重定向。

对于编程题目，要求选手给出的解答完全符合ANSI C++标准，不能使用诸如绘图、Win32API、中断调用、硬件操作或与操作系统相关的API。

代码中允许使用STL类库，但不能使用MFC或ATL等非ANSI C++标准的类库。例如，不能使用CString类型（属于MFC类库）。

注意: main函数结尾需要return 0

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

提交时，注意选择所期望的编译器类型。

程序设计（满分16分）

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，必须通过编程的方式解决问题。

注意：在评卷时使用的输入数据与试卷中给出的示例数据可能是不同的。选手的程序必须是通用的，不能只对试卷中给定的数据有效。

仔细阅读程序的输入、输出要求，千万不要输出没有要求的、多余的内容，例如：“请您输入xx数据：”。

建议仔细阅读示例，不要想当然！

程序处理完一个用例的数据后，立即退出（return 0），千万不要循环等待下一个用例的输入。

程序必须使用标准输入、标准输出，以便于机器评卷时重定向。

对于编程题目，要求选手给出的解答完全符合ANSI C++标准，不能使用诸如绘图、Win32API、中断调用、硬件操作或与操作系统相关的API。

代码中允许使用STL类库，但不能使用MFC或ATL等非ANSI C++标准的类库。例如，不能使用CString类型（属于MFC类库）。

注意: main函数结尾需要return 0

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

提交时，注意选择所期望的编译器类型。

程序设计（满分29分）

问题的描述在考生文件夹下对应题号的“题目描述.txt”中。相关的参考文件在同一目录中。请先阅读题目，必须通过编程的方式解决问题。

注意：在评卷时使用的输入数据与试卷中给出的示例数据可能是不同的。选手的程序必须是通用的，不能只对试卷中给定的数据有效。

仔细阅读程序的输入、输出要求，千万不要输出没有要求的、多余的内容，例如：“请您输入xx数据：”。

建议仔细阅读示例，不要想当然！

程序处理完一个用例的数据后，立即退出（return 0），千万不要循环等待下一个用例的输入。

程序必须使用标准输入、标准输出，以便于机器评卷时重定向。

对于编程题目，要求选手给出的解答完全符合ANSI C++标准，不能使用诸如绘图、Win32API、中断调用、硬件操作或与操作系统相关的API。

代码中允许使用STL类库，但不能使用MFC或ATL等非ANSI C++标准的类库。例如，不能使用CString类型（属于MFC类库）。

注意: main函数结尾需要return 0

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

提交时，注意选择所期望的编译器类型。

# 题目标题: 高斯日记

大数学家高斯有个好习惯：无论如何都要记日记。

他的日记有个与众不同的地方，他从不注明年月日，而是用一个整数代替，比如：4210

后来人们知道，那个整数就是日期，它表示那一天是高斯出生后的第几天。这或许也是个好习惯，它时时刻刻提醒着主人：日子又过去一天，还有多少时光可以用于浪费呢？

高斯出生于：1777年4月30日。

在高斯发现的一个重要定理的日记上标注着：5343，因此可算出那天是：1791年12月15日。

高斯获得博士学位的那天日记上标着：8113

请你算出高斯获得博士学位的年月日。

提交答案的格式是：yyyy-mm-dd, 例如：1980-03-21

请严格按照格式，通过浏览器提交答案。

注意：只提交这个日期，不要写其它附加内容，比如：说明性的文字。

# 标题: 马虎的算式

小明是个急性子，上小学的时候经常把老师写在黑板上的题目抄错了。

有一次，老师出的题目是：36 x 495 = ?

他却给抄成了：396 x 45 = ?

但结果却很戏剧性，他的答案竟然是对的！！

因为 36 \* 495 = 396 \* 45 = 17820

类似这样的巧合情况可能还有很多，比如：27 \* 594 = 297 \* 54

假设 a b c d e 代表1~9不同的5个数字（注意是各不相同的数字，且不含0）

能满足形如： ab \* cde = adb \* ce 这样的算式一共有多少种呢？

请你利用计算机的优势寻找所有的可能，并回答不同算式的种类数。

满足乘法交换律的算式计为不同的种类，所以答案肯定是个偶数。

答案直接通过浏览器提交。

注意：只提交一个表示最终统计种类数的数字，不要提交解答过程或其它多余的内容。

# 题目标题: 第39级台阶

小明刚刚看完电影《第39级台阶》，离开电影院的时候，他数了数礼堂前的台阶数，恰好是39级!

站在台阶前，他突然又想着一个问题：

如果我每一步只能迈上1个或2个台阶。先迈左脚，然后左右交替，最后一步是迈右脚，也就是说一共要走偶数步。那么，上完39级台阶，有多少种不同的上法呢？

请你利用计算机的优势，帮助小明寻找答案。

要求提交的是一个整数。

注意：不要提交解答过程，或其它的辅助说明文字。

标题: 黄金连分数

黄金分割数0.61803... 是个无理数，这个常数十分重要，在许多工程问题中会出现。有时需要把这个数字求得很精确。

对于某些精密工程，常数的精度很重要。也许你听说过哈勃太空望远镜，它首次升空后就发现了一处人工加工错误，对那样一个庞然大物，其实只是镜面加工时有比头发丝还细许多倍的一处错误而已，却使它成了“近视眼”!!

言归正传，我们如何求得黄金分割数的尽可能精确的值呢？有许多方法。

比较简单的一种是用连分数：

1

黄金数 = ---------------------

1

1 + -----------------

1

1 + -------------

1

1 + ---------

1 + ...

这个连分数计算的“层数”越多，它的值越接近黄金分割数。

请你利用这一特性，求出黄金分割数的足够精确值，要求四舍五入到小数点后100位。

小数点后3位的值为：0.618

小数点后4位的值为：0.6180

小数点后5位的值为：0.61803

小数点后7位的值为：0.6180340

（注意尾部的0，不能忽略）

你的任务是：写出精确到小数点后100位精度的黄金分割值。

注意：尾数的四舍五入！ 尾数是0也要保留！

显然答案是一个小数，其小数点后有100位数字，请通过浏览器直接提交该数字。

注意：不要提交解答过程，或其它辅助说明类的内容。

# 题目标题：前缀判断

如下的代码判断 needle\_start指向的串是否为haystack\_start指向的串的前缀，如不是，则返回NULL。

比如："abcd1234" 就包含了 "abc" 为前缀

char\* prefix(char\* haystack\_start, char\* needle\_start)

{

char\* haystack = haystack\_start;

char\* needle = needle\_start;

while(\*haystack && \*needle){

if(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_) return NULL; //填空位置

}

if(\*needle) return NULL;

return haystack\_start;

}

请分析代码逻辑，并推测划线处的代码，通过网页提交。

注意：仅把缺少的代码作为答案，千万不要填写多余的代码、符号或说明文字！！

# 标题：三部排序

一般的排序有许多经典算法，如快速排序、希尔排序等。

但实际应用时，经常会或多或少有一些特殊的要求。我们没必要套用那些经典算法，可以根据实际情况建立更好的解法。

比如，对一个整型数组中的数字进行分类排序：

使得负数都靠左端，正数都靠右端，0在中部。注意问题的特点是：负数区域和正数区域内并不要求有序。可以利用这个特点通过1次线性扫描就结束战斗!!

以下的程序实现了该目标。

其中x指向待排序的整型数组，len是数组的长度。

void sort3p(int\* x, int len)

{

int p = 0;

int left = 0;

int right = len-1;

while(p<=right){

if(x[p]<0){

int t = x[left];

x[left] = x[p];

x[p] = t;

left++;

p++;

}

else if(x[p]>0){

int t = x[right];

x[right] = x[p];

x[p] = t;

right--;

}

else{

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; //填空位置

}

}

}

如果给定数组：

25,18,-2,0,16,-5,33,21,0,19,-16,25,-3,0

则排序后为：

-3,-2,-16,-5,0,0,0,21,19,33,25,16,18,25

请分析代码逻辑，并推测划线处的代码，通过网页提交

注意：仅把缺少的代码作为答案，千万不要填写多余的代码、符号或说明文字！！

# 标题：错误票据

某涉密单位下发了某种票据，并要在年终全部收回。

每张票据有唯一的ID号。全年所有票据的ID号是连续的，但ID的开始数码是随机选定的。

因为工作人员疏忽，在录入ID号的时候发生了一处错误，造成了某个ID断号，另外一个ID重号。

你的任务是通过编程，找出断号的ID和重号的ID。

假设断号不可能发生在最大和最小号。

要求程序首先输入一个整数N(N<100)表示后面数据行数。

接着读入N行数据。

每行数据长度不等，是用空格分开的若干个（不大于100个）正整数（不大于100000）

每个整数代表一个ID号。

要求程序输出1行，含两个整数m n，用空格分隔。

其中，m表示断号ID，n表示重号ID

例如：

用户输入：

2

5 6 8 11 9

10 12 9

则程序输出：

7 9

再例如：

用户输入：

6

164 178 108 109 180 155 141 159 104 182 179 118 137 184 115 124 125 129 168 196

172 189 127 107 112 192 103 131 133 169 158

128 102 110 148 139 157 140 195 197

185 152 135 106 123 173 122 136 174 191 145 116 151 143 175 120 161 134 162 190

149 138 142 146 199 126 165 156 153 193 144 166 170 121 171 132 101 194 187 188

113 130 176 154 177 120 117 150 114 183 186 181 100 163 160 167 147 198 111 119

则程序输出：

105 120

资源约定：

峰值内存消耗 < 64M

CPU消耗 < 1000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。

# 

# 题目标题：翻硬币

小明正在玩一个“翻硬币”的游戏。

桌上放着排成一排的若干硬币。我们用 \* 表示正面，用 o 表示反面（是小写字母，不是零）。

比如，可能情形是：\*\*oo\*\*\*oooo

如果同时翻转左边的两个硬币，则变为：oooo\*\*\*oooo

现在小明的问题是：如果已知了初始状态和要达到的目标状态，每次只能同时翻转相邻的两个硬币,那么对特定的局面，最少要翻动多少次呢？

我们约定：把翻动相邻的两个硬币叫做一步操作，那么要求：

程序输入：

两行等长的字符串，分别表示初始状态和要达到的目标状态。每行的长度<1000

程序输出：

一个整数，表示最小操作步数

例如：

用户输入：

\*\*\*\*\*\*\*\*\*\*

o\*\*\*\*o\*\*\*\*

程序应该输出：

5

再例如：

用户输入：

\*o\*\*o\*\*\*o\*\*\*

\*o\*\*\*o\*\*o\*\*\*

程序应该输出：

1

资源约定：

峰值内存消耗 < 64M

CPU消耗 < 1000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。

# 

# 标题：带分数

100 可以表示为带分数的形式：100 = 3 + 69258 / 714

还可以表示为：100 = 82 + 3546 / 197

注意特征：带分数中，数字1~9分别出现且只出现一次（不包含0）。

类似这样的带分数，100 有 11 种表示法。

题目要求：

从标准输入读入一个正整数N (N<1000\*1000)

程序输出该数字用数码1~9不重复不遗漏地组成带分数表示的全部种数。

注意：不要求输出每个表示，只统计有多少表示法！

例如：

用户输入：

100

程序输出：

11

再例如：

用户输入：

105

程序输出：

6

资源约定：

峰值内存消耗 < 64M

CPU消耗 < 3000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。

# 

# 标题：连号区间数

小明这些天一直在思考这样一个奇怪而有趣的问题：

在1~N的某个全排列中有多少个连号区间呢？这里所说的连号区间的定义是：

如果区间[L, R] 里的所有元素（即此排列的第L个到第R个元素）递增排序后能得到一个长度为R-L+1的“连续”数列，则称这个区间连号区间。

当N很小的时候，小明可以很快地算出答案，但是当N变大的时候，问题就不是那么简单了，现在小明需要你的帮助。

输入格式：

第一行是一个正整数N (1 <= N <= 50000), 表示全排列的规模。

第二行是N个不同的数字Pi(1 <= Pi <= N)， 表示这N个数字的某一全排列。

输出格式：

输出一个整数，表示不同连号区间的数目。

示例：

用户输入：

4

3 2 4 1

程序应输出：

7

用户输入：

5

3 4 2 5 1

程序应输出：

9

解释：

第一个用例中，有7个连号区间分别是：[1,1], [1,2], [1,3], [1,4], [2,2], [3,3], [4,4]

第二个用例中，有9个连号区间分别是：[1,1], [1,2], [1,3], [1,4], [1,5], [2,2], [3,3], [4,4], [5,5]

资源约定：

峰值内存消耗 < 64M

CPU消耗 < 5000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。