蓝桥杯全国软件和信息技术专业人才大赛组委会

第九届蓝桥杯全国软件和信息技术专业人才大赛

竞赛规则及说明（个人赛软件类）

1. 组别

竞赛分为：C/C++大学 A 组，C/C++大学 B 组，C/C++大学 C 组，Java 大学 A 组，Java 大学 B 组，Java 大学 C 组共 6 个组别。

每位选手只能申请参加其中一个组别的竞赛。各个组别单独评奖。

一本院校（985、211）本科生只能报大学 A 组。所有院校研究生只能报大学 A 组。

其它本科院校本科生可自行选择大学 A 组或大学 B 组。

其它高职、高专院校可自行选择报任意组别。

2. 竞赛赛程

预赛（省赛）时长：4 小时。6 个组别同时进行。

决赛时长：4 小时。分上下午两个场次（每位选手只参加其中一个场次）。

详细赛程安排以组委会公布信息为准。

3. 竞赛形式

个人赛，一人一机，全程机考。

选手机器通过局域网连接到各个赛场的竞赛服务器。

选手答题过程中无法访问互联网，也不允许使用本机以外的资源（如 USB 连接）。

竞赛系统以“服务器-浏览器”方式发放试题、回收选手答案。

4. 参赛选手机器环境

选手机器配置：

X86 兼容机器，内存不小于 1G，硬盘不小于 60G

操作系统：WindowsXP 或 Windows7

C/C++ 语言开发环境：

- Dev-cpp 5.4.0 支持 ANSI C，ANSIC++，STL
- C/C++ API 帮助文档（中文，chm 格式）

Java 语言开发环境：

- JDK 1.6
- Eclipse Helios for JavaSE
- API 帮助文档（中文，chm 格式）

5. 试题形式

竞赛题目完全为客观题型。

根据选手所提交答案的测评结果为评分依据。

共有三种题型。

5.1. 结果填空题

- 题目描述一个具有确定解的问题。要求选手对问题的解填空。
- 不要求解题过程，不限制解题手段（可以使用任何开发语言或工具，甚至是手工计算），只要求填写最终的结果。

5.2. 代码填空题

- 题目描述一个具有确定解的问题。
- 题目同时给出该问题的某一解法的代码，但代码有缺失部分。

要求选手读懂代码逻辑，对其中的空缺部分补充代码，使整段代码完整。

只填写空缺部分，不要填写完整句子。不要写注释、说明或其它题目中未要求的内容。

所填代码应该具有通用性，不能只对题面中给出的特殊示例有效。

5.3. 编程大题

题目为若干具有一定难度梯度、分值不等的编程题目。

这些题目的要求明确、答案客观。

题目一般要用到标准输入和输出。要求选手通过编程，对给定的标准输入求解，并通过标准输出，按题目要求的格式输出解。题目一般会给出示例数据。

一般题目的难度主要集中于对算法的设计和逻辑的组织上。理论上，选手不可能通过猜测或其它非编程的手段获得问题的解。

选手给出的解法应具有普遍性，不能只适用于题目的示例数据（当然，至少应该适用于题目的示例数据）。

为了测试选手给出解法的性能，评分时用的测试用例可能包含大数据量的压力测试用例，选手选择算法时要尽可能考虑可行性和效率问题。

6. 试题涉及的基础知识

- Java 大学 C 组

解题所涉及的知识：基本语法、面向对象、网络编程、接口、集合、IO、多线程、内部类、异常与保护，基本数据结构。（不涉及 swing 等图形界面，不涉及 html、JSP、Tomcat、开源框架等 web 开发方面，不涉及 JDBC、SQL 等数据库编程方面）

解题允许使用的特性：JDK1.6 支持的特性

- Java 大学 B 组

解题所涉及的知识：Java 大学 C 组全部知识 + 数据结构（高校《数据结构》教材中出现的经典结构，及其通过组合、变形、改良等方法创造出的变种）+ 大学程度的基本数学知识（含：解析几何、线性代数、微积分、概率、复平面基本性质）

解题允许使用的特性：同 java 大学 C 组

- Java 大学 A 组

解题所涉及的知识：Java 大学 B 组全部知识 + 设计模式，反射，XML，多核与并发，

软件测试。

解题允许使用的特性：同 Java 大学 C 组

- C/C++大学 C 组

解题所涉及的知识：结构、数组、指针、标准输入输出、文件操作、递归、基本数据结构（在代码填空中不会出现 C++知识，不会出现 ANSI C/C++ 之外的 windows API 调用）

解题允许使用的特性：选手可以使用 C 风格或 C++风格或混合风格解答编程大题。

允许使用 ANSI C(99) / ANSI C++(98) 特性。允许使用 STL 类库。

- C/C++大学 B 组

解题所涉及的知识：C/C++大学 C 组全部知识 + 数据结构（高校《数据结构》教材中出现的经典结构，及其通过组合、变形、改良等方法创造出的变种）、函数指针、位运算 + 大学程度的基本数学知识（含：解析几何、线性代数、微积分、概率、复平面基本性质）

解题允许使用的特性：同 C/C++大学 C 组

- C/C++大学 A 组

解题所涉及的知识：C/C++大学 B 组全部知识 + 函数模板、宏替换、汇编知识

解题允许使用的特性：同 C/C++大学 C 组

7. 涉及的领域知识

除了编程语言的基础知识，大赛很少用到特定领域的知识。比如：电信、医药、地质、银行等特定领域。如果偶尔用到，会详细解释概念，并给出足够的示例。

但“数学领域”是个例外。大赛假定选手具有足够的中学数学知识。

包括：

- 算数：素数，整出，余数，求模，不定方程 …
- 代数：函数，方程，多项式，…
- 解析几何：笛卡尔坐标系，点到直线的举例，极坐标，…
- 复数：模，夹角，矢量的合成和分解

8. 评分

全程机器阅卷。必要环节有少量人工介入。

- 结果填空题：

答案唯一。

只有 0 分或满分（格式错误为 0 分）。

- 程序填空题：

按选手填写的代码代入程序中能否得出正确结果为判据。

测试数据与题面中的数据可能不同。

只有 0 分或满分（格式错误为 0 分）

C/C++组选错了编译器类型可能得 0 分。

- 编程大题：

主要以选手所提交的程序的运行结果为依据。特殊情况会参考选手程序的编码风格、逻辑性、可读性等方面。

多个测试用例单独计分。通过则该用例得分。

C/C++选手选错了编译器类型可能得 0 分

C/C++选手主程序没有 return 0 可能得 0 分。

Java 选手使用了 package 语句按 0 分处理。

Java 选手主类名字不是 Main 按 0 分处理。

其它题目中明确告知的规则如不遵守，都可能导致 0 分。

9. 高职高专特色

为了照顾到高职高专的教学特点，大赛为高职高专设计的部分填空题目（并非全部），题面上给出了算法的详细描述，要求选手分析代码，填写缺少的语句。

对于部分编程大题也给出某种解法的提示。当然，选手完全可以另辟蹊径，用自己的方法解决问题。仍然是以代码的最终执行效果为评分依据。

具体题目示例参见样题。

10.通用样题

【编程大题】花朵数

一个 N 位的十进制正整数，如果它的每个位上的数字的 N 次方的和等于这个数本身，则称其为花朵数。

例如：当 N=3 时，153 就满足条件，因为 1^3 + 5^3 + 3^3 = 153，这样的数字也被称为水仙花数（其中，“^”表示乘方，5^3 表示 5 的 3 次方，也就是立方）。

当 N=4 时，1634 满足条件，因为 1^4 + 6^4 + 3^4 + 4^4 = 1634。

当 N=5 时，92727 满足条件。

实际上，对 N 的每个取值，可能有多个数字满足条件。

程序的任务是：求 N=21 时，所有满足条件的花朵数。注意：这个整数有 21 位，它的各个位数字的 21 次方之和正好等于这个数本身。

如果满足条件的数字不只有一个，请从小到大输出所有符合条件的数字，每个数字占一行。因为这个数字很大，请注意解法时间上的可行性。要求程序在 1 分钟内运行完毕。

【程序运行参考结果】

128468643043731391252

449177399146038697307

【编程大题】信用卡号验证

当你输入信用卡号码的时候，有没有担心输错了而造成损失呢？其实可以不必这么担心，因为并不是一个随便的信用卡号码都是合法的，它必须通过 Luhn 算法来验证通过。该校验的过程：

1、从卡号最后一位数字开始，逆向将奇数位(1、3、5 等等)相加。

2、从卡号最后一位数字开始，逆向将偶数位数字，先乘以 2（如果乘积为两位数，则将其减去 9），再求和。

3、将奇数位总和加上偶数位总和，结果应该可以被 10 整除。

例如，卡号是：5432123456788881

则奇数、偶数位（用红色标出）分布：5432123456788881

奇数位和=35

偶数位乘以 2（有些要减去 9）的结果：1 6 2 6 1 5 7 7，求和=35。

最后 35+35=70 可以被 10 整除，认定校验通过。

请编写一个程序，从标准输入获得卡号，然后判断是否校验通过。通过显示：“成功”，否则显示“失败”。

比如，用户输入：356827027232780
程序输出：成功
【程序测试参考用例】
356406010024817　成功
358973017867744　成功
356827027232781　失败
306406010024817　失败
358973017867754　失败

【C/C++组代码填空】
下列代码把一个串 p 复制到新的位置 q。请填写缺少的语句；

```
char* p = "abcde";
char* q = (char*)malloc(strlen(p)+1);
for(int i=0;_____________; i++) q[i] = p[i];
q[i] = 0;
```

【参考答案】

```
p[i] 或 *(p+i) 或 p[i] != '\0' 或 ...
```

（答案不唯一，以选手提供代码带入专用验证程序测试为依据，验证程序比题面中提供的程序片段更严谨，更完善）

【Java 组代码填空】
有 n 个孩子站成一圈，从第一个 孩子开始顺时针方向报数，报到 3 的人出列，下一个人继续从 1 报数，直到最后剩下一个孩子为止。问剩下第几个孩子。下面的程序以 10 个孩子为例，模拟了这个 过程，请完善之（提示：报数的过程被与之逻辑等价的更容易操作的过程所代替）。

```
Vector a = new Vector();
for(int i=1; i<=10; i++)
{
a.add("第" + i + "个孩子");
}
for(;;)
{
if(a.size()==1) break;
for(int k=0; k<2; k++)
_________________;
a.remove(0);
}
System.out.println(a);
```

【参考答案】

```
a.add(a.remove(0))
```

（答案不唯一，以选手提供代码带入专用验证程序测试为依据，验证程序比题面中提供的程序片段更严谨，更完善）

【结果填空题】 有趣的平方数

625 这个数字很特别，625 的平方等于 390625，刚好其末 3 位是 625 本身。除了 625，还有其它的 3 位数有这个特征吗？还有一个！该数是：_____

【参考答案】

376

11. 高职高专特色样题

【代码填空--java】

下面的代码定义了一个方法 hasSameChar，用于判定一个给定的串中是否含有重复的字符，比如“about”中，就没有重复的字符，而“telecom”，“aabaa”中都含有重复的字符，其中“e”重复了 2 次，而“a”重复了 4 次，这些都算作有重复。

请根据方法的说明，分析给出的源程序，并填写划线部分缺失的代码。

注意，只填写缺少的，不要重复周围已经给出的内容，也不要填写任何说明性文字等。

```
public class A
{
    /*
      判断串 s 中是否含有重复出现的字符
      如果有重复则返回 true
      其它情况返回 false

      判断的思路是：从左到右扫描每个字符
      对当前的字符，从右向左在 s 串中搜索它的出现位置，可以用 lastIndexOf 方法
      如果找到的位置与当前的位置不同，则必然存在该字符的重复现象，即可返回 true
      其它情况返回 false

      在特殊情况下，比如传入的是空指针，或者 s 为空串，或者只含有 1 个字符，都不可能含有重复字符，
      因此，这些情况直接返回 false
    */
    public static boolean hasSameChar(String s){
        if(s==null || s.length()<2) return false;
        for(int i=0; i<s.length(); i++){
            char c = s.charAt(i);
            int k = s.lastIndexOf(c);
            if(_____) return true;
        }
        return false;
    }

    public static void main(String[] args){
        System.out.println(hasSameChar("a"));  //false
        System.out.println(hasSameChar("abcdefg"));  //false
        System.out.println(hasSameChar("abacdefag"));  //true
        System.out.println(hasSameChar("abcdebfg"));  //true
```

```
    }
}
```

【编程大题】

用天平称重时，我们希望用尽可能少的砝码组合称出尽可能多的重量。

如果只有 5 个砝码，重量分别是 1，3，9，27，81。则它们可以组合称出 1 到 121 之间任意整数重量（砝码允许放在左右两个盘中）。

本题目要求编程实现：对用户给定的重量，给出砝码组合方案。

例如：

用户输入：

```
5
```

程序输出：

```
9-3-1
```

用户输入：

```
19
```

程序输出：

```
27-9+1
```

要求程序输出的组合总是大数在前小数在后。

可以假设用户的输入的数字符合范围 1~121。

【解题思路提示】

我们把已知的砝码序列记为：x1, x2, x3, x4, x5, x6 (这里多加一个标准砝码，为解题叙述方便)

对于任意给定的重量 x，如果刚好等于 xi 则问题解决。

否则一定会位于两个标准砝码重量的中间，不妨设为：xi < x < xj

令 a = x – xi, b = xj – x

则，x 要么可以表示为： xi + a, 要么可以表示为： xj – b

这样问题就归结为怎样表示出 a 或 b

另一思路：对于每个 xi，可以乘以一个系数 ki，再求和。

ki 的数值无外乎：-1 0 1

这样，因为标准砝码的数量的很少的，我们就可以多层循环暴力组合 ki 来求解。

还有更“土气”但有效的思路：既然输入范围只有 120 左右，如果对每一种情况都做人工求解，只要列一个大表，等查询的时候，直接输出答案就好了啊！但…这似乎是个耗时的工程…

【代码填空--c】

下面的代码先调用 init 函数填充一个二维数组，然后调用 show 函数，把该数组显示出来。

二维数组指针传入函数的时候被当做一维数组来处理。这里利用了二维数组在内存中存储特征：先存第 1 行，再存第 2 行，...

因此，我们根据一维数组的下标，可以计算出其二维数组的下标。

```
#include <stdio.h>
#define ROW 5
#define COL 5

void init(int* p)
{
    int i;
    for(i=0; i<ROW*COL; i++){
        p[i] = i+1;
    }
}

void show(int* p)
{
    int i;
    for(i=0; i<ROW*COL; i++){
        printf("%3d ", p[i]);
        if(__________________){   //填空位置
            printf("\n");
        }
    }
}

int main()
{
    int a[ROW][COL];
    init((int*)a);
    show((int*)a);
    return 0;
}
```

为了显示为二维数组，需要在每一行的结束输出换行符。关键是计算出应该输出的时机。

注意：

第一行之前不能换行（那样会产生一个空行），最后一行结束需要换行符。

另外注意不要把程序写“死”了，当改变 ROW COl 值得时候，程序的反应应该是正常的。

对于题目中的值，应该输出：

```
  1   2   3   4   5
  6   7   8   9  10
 11  12  13  14  15
 16  17  18  19  20
```

21 22 23 24 25

请仔细阅读源程序，填写划线位置缺少的代码。

参考答案：i%COL==COL-1

【代码填空--c】

假设一个句子是由许多单词用一个或多个空格分开的，比如：

"cow dog cat cow horse dog duck cat dog dog"

请你统计出每个词出现的次数。

输出类似：

cow = 2

dog = 3

cat = 2

horse = 1

duck = 1

下面的程序实现了这个功能。其基本思想是：

准备一个结构数组，每个结构中存储：单词和它出现的次数。

从头扫描串，分离出一个个的单词，在结构数组中查找，如果有，就把该结构的计数增加，如果没有，就创建新的结构，并记录其次数为 1.

分离单词的算法是：一直扫描待定串，如果遇到空格，就把已经累积的字符做成单词送出，如果不是空格，就把该字符累积起来，然后扫描下一个字符。

请仔细阅读程序，填写划线部分缺少的代码。

```
#include <stdio.h>
#include <string.h>
#define MAX_NUM 30

typedef struct{
    char word[20];
    int num;
}WORD_CELL;

// 在 words 中查找，如果找到，把其 num 增加，如果没有，建立新记录
int PutWord(WORD_CELL* words, int n, char* word)
{
    WORD_CELL* p= words;
    int i;
    for(i=0; i<n; i++){
        if(strcmp(p->word,word)==0){
            p->num++;
            return n;
```

```
        }
        p++;
    }

    strcpy(_____________________________); //填空位置
    p->num = 1;
    return n+1;
}

void show(WORD_CELL* words, int n)
{
    int i;
    for(i=0; i<n; i++){
        printf("%s = %d\n", words[i].word, words[i].num);
    }
}

void f(char* p)
{
    char buf[100];  //存储临时产生的单词
    int nBuf = 0;  // 记录 buf 中的有效字符数

    WORD_CELL words[MAX_NUM];
    int nWords = 0; //记录 words 中的有效记录数目

    for(;*p;p++){
        if(*p==' '){
            if(nBuf>0){
                buf[nBuf] = '\0';
                nWords = PutWord(words, nWords, buf);
                nBuf = 0;
            }
        }
        else {
            buf[nBuf++] = *p;
        }
    }

    if(nBuf>0){
        buf[nBuf] = '\0';
        nWords = PutWord(words, nWords, buf);
    }
    show(words, nWords);
}
```

```
int main()
{
    char word[] = "cow   dog cat cow horse dog  duck  cat dog    dog";
    f(word);
    return 0;
}
```

参考答案：p->word, word

【代码填空--java】

如下代码以螺旋方式在一个矩阵中顺序填写数字，然后显示该矩阵。
Mat 类实现了顺时针填充。
MyMat 类继承了 Mat 类，实现了逆时针填充。

填充的大约步骤是：
先安装约定的行列数目，创建一个临时的二维数组，
然后选定一个起始位置和初始的填充方向，
在该位置填充一个数字，然后移动到下一个位置。
如果下一个位置是越界的，或者已经有了填充的数字，则变换填充的方向（转弯）

Pos 类的对象用来存储当前填充的位置和填充的方向。
其中 x，y 是当前位置的坐标。dx,dy 是列向、行向的增量，用来决定下一个的位置。
比如往右移动：dx=1,dy=0, 表示列加 1，行不动。
再比如，往上移动：dx=0,dy=-1, 表示列不动，行号减 1

```
/*
待填入位置的参数：
    x 待填入列号
    y 待填入行号
    dx 当前走向--列向
    dy 当前走向--行向
*/
class Pos
{
    public int x;
    public int y;
    public int dx;
    public int dy;
    public Pos(int x, int y, int dx, int dy){
        this.x = x;
        this.y = y;
        this.dx = dx;
```

```
        this.dy = dy;
    }
    public Pos(Pos t)
    {
        x = t.x;
        y = t.y;
        dx = t.dx;
        dy = t.dy;
    }
}

class Mat
{
    private int row;  // 行数
    private int col;  // 列数

    public Mat(){
        row = 5;
        col = 5;
    }
    public Mat(int row, int col)
    {
        this.row = row;
        this.col = col;
    }

    public void showArray(int[][] ar)
    {
        System.out.println();
        for(int i=0; i<ar.length; i++){
            for(int j=0; j<ar[i].length; j++){
                System.out.print(String.format("%3d ",ar[i][j]));
            }
            System.out.println();
        }
    }

    public void print()
    {
        int n = row * col; // 总的元素个数
        int[][] ar = new int[row][col];  //创建临时的二维数组，向其中填写数字
        Pos cur = initPos(); //待填入位置初始参数
        int m = 1; // 待填入的初始数字
```

```
    for(;;){
        ar[cur.y][cur.x] = m;  //填入数字
        if(m>=n) break;
        cur = getNextPos(ar,cur);  //求下一个位置参数
        m++;  //求下一个被填入的数字
    }

    showArray(ar);
}

protected Pos initPos()
{
    return new Pos(0,0,1,0);
}

protected Pos getNextPos(int[][] ar, Pos cur)
{
    Pos t = new Pos(cur);
    t.x += t.dx;  // 按当前方向试走一下
    t.y += t.dy;

    if(t.x >=0 && t.x < col && t.y >= 0 && t.y < row  // 位置没有越界
    && ar[t.y][t.x] == 0) { //该位置也没有被填充过
        return t;
    }
    else{  // 转向后，再试验走到下一个位置
        t = changeDirection(cur);
        return getNextPos(ar, t);
    }
}

protected Pos changeDirection(Pos p)
{
    Pos t = new Pos(p);
    if(p.dx==0 && p.dy==1){ // 向右转向下
        t.dx = 1;
        t.dy = 0;
    }
    else if(p.dx==1 && p.dy==0){ // 向下转向左
        t.dx = 0;
        t.dy = -1;
    }
    else if(p.dx==0 && p.dy==-1){ // 向左转向上
        t.dx = -1;
```

```
            t.dy = 0;
        }
        else if(p.dx==-1 && p.dy==0){ // 向上转向右
            t.dx = 0;
            t.dy = 1;
        }
        return t;
    }
}

class MyMat extends Mat
{
    protected Pos changeDirection(Pos p)
    {
        // super.changeDirection(p);
        // 填写代码的位置
    }

    protected Pos initPos()
    {
        return new Pos(0,0,0,1);
    }
}

public class A
{
    public static void main(String[] args)
    {
        Mat a = new Mat(5,8);
        a.print();

        a = new MyMat();
        a.print();
    }
}
```

请仔细阅读源代码，并完成 MyMat 类的 changeDirection 这个方法的内容。
这个方法控制着填充过程，碰壁后需要转去的新方向。

对于上面的测试数据，程序应该输出：

```
  1   2   3   4   5   6   7   8
 22  23  24  25  26  27  28   9
```

```
21  36  37  38  39  40  29  10
20  35  34  33  32  31  30  11
19  18  17  16  15  14  13  12

 1  16  15  14  13
 2  17  24  23  12
 3  18  25  22  11
 4  19  20  21  10
 5   6   7   8   9
```

参考答案：

```
Pos t = new Pos(p);
if(p.dx==0 && p.dy==1){
    t.dx = -1;
    t.dy = 0;
}
else if(p.dx==1 && p.dy==0){
    t.dx = 0;
    t.dy = 1;
}
else if(p.dx==0 && p.dy==-1){
    t.dx = 1;
    t.dy = 0;
}
else if(p.dx==-1 && p.dy==0){
    t.dx = 0;
    t.dy = -1;
}
return t;
```

12.其它注意事项

（1）选手必须符合参赛资格，不得弄虚作假。资格审查中一旦发现问题，则取消其报名资格；竞赛过程中发现问题，则取消竞赛资格；竞赛后发现问题，则取消竞赛成绩，收回获奖证书及奖品等，并在大赛官网上公示。

（2）参赛选手应遵守竞赛规则，遵守赛场纪律，服从大赛组委会的指挥和安排，爱护竞赛赛场地的设备。

（3）竞赛采用机器阅卷+少量人工辅助。选手需要特别注意提交答案的形式。必须仔细阅读题目的输入、输出要求以及示例，不要随意添加不需要的内容。

（4）使用 Java 语言时，注意主类名必须是：Main，不能使用 package 语句。

使用 C/C++语言时，注意主函数需要 return 0；

（5）C 组与 C++组选手提交答案时，一定要注意选择 C 或 C++（即编译器类型）。因为使用机器阅卷，很可能会因为选手选择了错误的编译器，而使自己代码无法编译通过。