E3 - 2021级程序设计基础第三次练习

A - 转为2进制

题目描述

给定一个int型整数N，输出其32位二进制原码。

测试数据保证原码能够表示出这个数。

输入0时，请置符号位为0。

输入描述

多组输入，每组一行，为N。

输出描述

每行输出对应一行输入，要求见题目描述。

样例输入

```
1
0
-1
```

样例输出

```
00000000000000000000000000000001
00000000000000000000000000000000
10000000000000000000000000000001
```

AUTHOR：*Mentor_D*

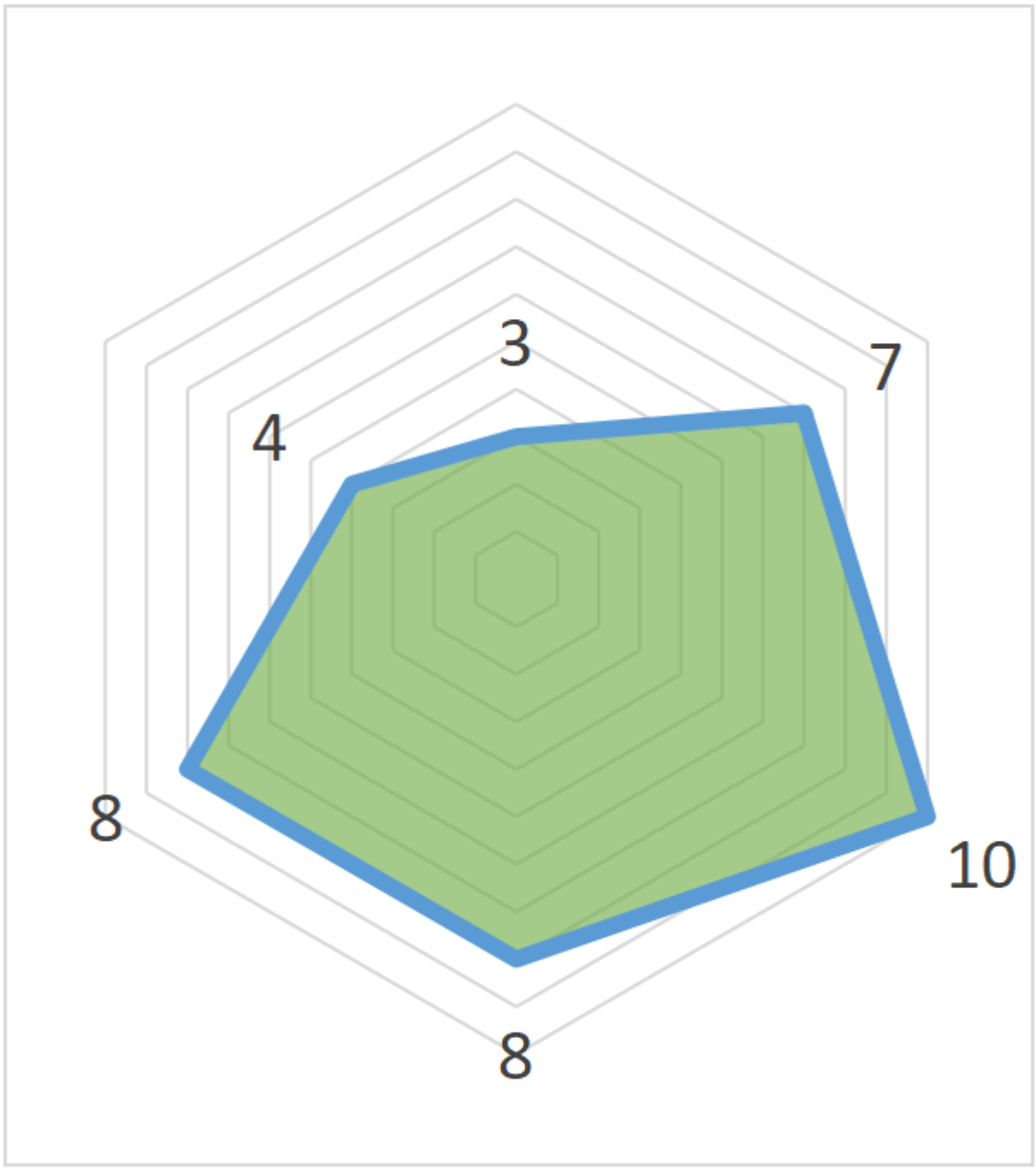
B - 六边形战士

题目背景

在美冬（*Mifuyu*）游戏出品的《圆神》中，玩家可以扮演魔法少女在世界中旅行探险，并和魔物进行战斗。游戏中使用一个六维雷达图对魔法少女的实力进行评价。

题目描述

n维雷达图绘制的过程是：首先等间隔角地绘制n条放射线作为坐标轴，然后在上面描上各个得分点，再依次将每个点连起来，然后可以根据需要将坐标轴擦掉：



如图，这是一个六维雷达图，六个维度的得分分别是3，7，10，8，8，4.

陆木缘觉得能用一个**雷达图线围成的面积和雷达图外圈的面积之比的100倍**来代表一位魔法少女的实力，于是她想知道她的队伍中的魔法少女的实力都是多少。即：

$$\text{实力值} = \frac{\text{雷达图线围成的面积}}{\text{雷达图外圈的总面积}} \cdot 100$$

“雷达图外圈”定义为所有维度都为10分（满分）的图线围成的区域。

肖梅雁看了以后，觉得雷达图也不一定非得是六维，再多几个评价维度或者少几个维度也可以评定魔法少女的实力。

输入描述

第一行是一个整数n，表示陆木缘的队伍中有n位魔法少女。

接下来的$2n$行，两行一组，共n组，是关于每个魔法少女的信息。

每组中的第一行是这个魔法少女的评价维度数k ($k \in [3, 20]$)，第二行k个数是每个维度的得分。得分为$0-10$分的整数。

输出描述

输出总共有n行，第i行是一个保留四位小数的实数，表示第i个魔法少女的实力值。

样例输入

```
4
6
8 3 10 7 8 4
3
10 10 10
5
8 0 5 3 10
3
0 0 10
```

样例输出

```
40.6667
100.0000
25.0000
0.0000
```

数据范围

$n \leq 1000, 3 \leq k \leq 20$

保证有60%的数据满足：所有评价维度都是6维。

HINT

要是实在实在不会做了，不妨把题图中的坐标轴重新画上看看？

*Author:*梁秋月

C - 三线共点===bug?

题目描述

Sheep在为自己制作测试数据时发现了自己能AC的代码但在运行自己做的测试数据时直接将编译器卡崩掉了，经过反复的寻找，发现导致最终运行不出来结果的数据在中间过程中的求交时发现了能在**计算机的浮点数精度上也能三线共点**的情况，于是Sheep就换用了另一种不会产生浮点数误差的方法求交，请你也来试试这种方法吧

题目描述

在xOy上有三条无限长直线，方程的基本形式如下：

$A_1x + B_1y + C_1 = 0$
$A_2x + B_2y + C_2 = 0$
$A_3x + B_3y + C_3 = 0$

你的任务是判断这三条直线是否会经过同一个点

输入描述

第一行，一个正整数$n(1 \leq n \leq 10^6)$表示数据组数

接下来一共$3n$行，对于第i组数据$(1 \leq i \leq n)$

第$3i-1$行，3个整数，表示A_1 ,B_1 ,C_1

第$3i$行，3个整数，表示A_2 ,B_2 ,C_2

第$3i+1$行，3个整数，表示A_3 ,B_3 ,C_3

输出描述

对每组数据

如果出现了3线共点的情况，输出一行一个字符串 `Could be a bug!`

对于一般情况，输出一行一个字符串 `A nice test data ~`

样例输入

```
2
1 -1 1
2 -1 -1
3 -1 -3
-194 611 520
-483 -425 -74
561 -756 -201
```

样例输出

```
Could be a bug!
A nice test data ~
```

数据范围

对每个测试点 均有$(1 \leq n \leq 10^6)$

对所有测试数据均有 $-600 \leq A_m, B_m, C_m \leq 600, 1 \leq m \leq 3$且 $A_m, B_m (1 \leq m \leq 3)$**不同时为0**

HINT

要注意可能产生的浮点数误差范围，避免机器产生过大的浮点数误差；

或者**选用不会浮点数误差的方法。**

测试点中可能会出现直线重合的情况，而**三线重合时我们也认为发生了三线共点的状况。**

AUTHOR: Oh So many sheep

D - Monica的密钥

题目描述

总所周知，$Monica$和$Lily$的聊天内容是经过加密的，但是每次加密的密钥可能不一样。

具体的，$Monica$每天会和聊天对象约定两个数a, b，第一条消息的密钥

$$K_1 = a^b \mod 998244353$$

之后每条消息的密钥

$$K_n = a^{K_{n-1}} \mod 998244353$$

现在$Monica$向你发出了一条信息，你解密后打算回复她，请计算K_2。

如果你只会算K_1，那么你就不能给$Monica$发消息啦。

输入描述

第一行两个整数a, b。

输出描述

输出一行一个整数K_2。

样例输入1

```
2 3
```

样例输出1

```
256
```

样例输入2

```
2 19260817
```

样例输出2

```
229045914
```

样例解释

第一个样例中

$$c = 2^3 = 8$$

$$d = 2^c = 2^8 = 256$$

数据范围

对于30%的数据，$a, b \leq 6$

对于100%的数据，a, b在int范围内

HINT

使用循环直接计算结果很可能会TLE。

对于指数，我们可以写出它的二进制表达，例如$(13)_{10} = (1101)_2$。

我们知道，底数相同的幂相乘，指数相加。

所以对于$a^{1101} = a^{1000} \times a^{100} \times a^{1}$（这里指数都是二进制表达），相当于把指数分解了。

我们可以很容易地使用循环结构求出$a^1, a^{10}, a^{100}...$的值，因此对于任意指数b，将其二进制表达分解后各个1所在位对应的幂乘起来即a^b的值。

类似求十进制数的各位数字，求二进制数的各位数字可以用如下代码

```
while(a){
    bit=a&1;//bit为1，则当前a二进制下最低位为1，反之亦然
    a/=2;
}
```

求各个幂的操作可以在上述循环中一并完成。

请注意取模操作。

AUTHOR: Monica

E - 进行一个班的分

题目描述

开学时导员将 n 个同学组成一个**临时班**，但是觉得班级人数太多了，因此给了你一份所有同学的成绩单，让你根据每个人的成绩按照如下步骤将大家分成不同**正规班**：

1. 若该**临时班**总人数小于 k，则其成为最后一个**正规班**（班号为 $s+1$，s 为已经分出来的正规班的数目，显然开始时 $s=0$），并结束分班。
2. 从第一个同学开始，依次将**连续**的 k 位同学分为一个**正规班**（若最后未分班的同学不足 k 人，则将其并到前一个班），班号依次为 $s+1, s+2, s+3, \ldots, s+m$（假设分出来了 m 个班）。
3. 将上一步分出来的 m 个班的**最高分**依次移出来再组成一个**临时班**（这几个同学将不再属于原来的班），并对该班再次分班（回到第 1 步）。

在分班时你还需注意以下几点:

- 学生的**相对顺序在分班前后保持不变**，例如学生 a 是 1 班的最高分，学生 b 是 2 班的最高分，两人在移出来到同一个临时班后 a 还应排在 b 前面。
- 只有**正规班**有班号，并且从 1 开始依次递增，**临时班**是没有班号的。
- 记录每个**正规班**的**最低分**，最后按照班号顺序将每个班的最低分输出。

输入描述

第一行两个整数 n 和 k

第二行 n 个整数 x ，表示每位同学的成绩

输出描述

多行输出，第 i 行输出 i 班的最低分

样例输入

```
10 3
10 23 21 89 11 43 56 99 60 78
```

样例输出

```
10
11
56
23
99
```

样例说明

先分成1班：$(10, 23, 21)$、2班：$(89, 11, 31)$、3班：$(56, 99, 60, 78)$，再将每个班最高分取出来组成临时班：$(23, 89, 99)$，该班人数不小于$k(k = 3)$，还要继续分班。

将该班分成4班：$(23, 89, 99)$，并取出最高分组成临时班：(99)，此时临时班只有1人(小于k)，结束分班，该临时班成为5班。

最后将1到5班最低分按顺序输出：$10, 11, 56, 23, 99$。

数据范围

对于 100% 的数据，$2 \leq n \leq 10^5$ ，$2 \leq k \leq 100$ ，$0 \leq x \leq 100$

HINT

在对上述过程进行模拟前，先理清楚思路，想好哪些数值是需要记录的，是否需要使用数组，循环的控制条件是什么等等。

AUTHOR: toush1

F - 进制转换

题目描述

给定一个 n 进制数 a，请你将它转换为 m 进制并输出。

特别的，令 $A, B, C, \ldots, Z$ 表示高进制下的 $10 \sim 35$。

例如，对于 16 进制下的数 $5A$，它表示十进制下的 90。

输入描述

一行,先输入一个字符串 a，表示需要进制转换的 n 进制数。再输入两个空格隔开的整数 n, m，含义见上。

输出描述

输出一行一个字符串，表示 n 进制下的数 a 在 m 进制下是多少。

样例输入1

```
8 10 2
```

样例输出1

```
1000
```

样例输入2

```
2HB4QH81T5O 35 13
```

样例输出2

```
199510435A68577
```

数据范围

假设数 a 对应十进制下的数 b，保证 $b \leq 10^{18}$。

HINT

如果你不知道如何输入，那么可以参照如下输入方式：

```
char ch[100];
int n,m;
scanf("%s%d%d",ch,&n,&m);
int len=strlen(ch);
```

其中，输入的字符串存在ch数组的第0位到第$len-1$位。

如果要使用上述代码，请添加头文件`string.h`，即在程序开头添加`#include<string.h>`。

Author : yzh

G - 相反数

题目描述

灵梦在早苗的程序设计课堂上学会了将一个十进制数在二进制表示下取相反数。所以回去之后，灵梦迫不及待地拿出了一台魔法计算机，已知这台计算机存一个数需要的位数是n。

举个例子：当$n = 8$时，$(13)_{10} = (00001101)_2, (-13)_{10} = (11110011)_2$

现在早苗给他出了一道难题：如果输入是在十六进制下，那么对应的十进制数取相反数的二进制结果是多少呢？

在本题中，十进制下的负数在二进制表示下最左边的第一位一定是符号位，0表示正，1表示负。

灵梦慌了，她没学过十六进制，于是找来了你，想让你帮她写一个程序，好水完早苗老师的作业。

输入描述

第一行一个十进制整数n，含义如题所述。

第二行一个二进制下长度为n的十六进制数。

输出描述

一行，一个n位数，表示处理之后的二进制数。

样例输入1

```
8
0D
```

样例输出1

```
11110011
```

样例输入2

```
16
F63F
```

样例输出2

```
0000100111000001
```

数据范围

对于所有数据，$n \leq 2^{20}, 4|n$。

数据保证十六进制数转成二进制后位数不超过n。

HINT

一般的进制转换常规做法是先从某一进制转为十进制，再从十进制转为另一进制，但从十六进制转为二进制或许有更简单的方法。

AUTHOR: Stockholm

H - 强迫症

题目描述

小H是个不折不扣的强迫症。对于长度为n的数列$a_1, a_2, \ldots, a_n$，小H希望这个数列看起来很“整齐”，否则他就会发作强迫症。在小H的概念中，如果这个数列是“整齐”的，则这个数列必须至少满足下列两种条件之一：

$a_1 \ \& \ a_2 \ \& \ \ldots \ \& \ a_n = m$

$a_1 \mid a_2 \mid \ldots \mid a_n = m$

其中，符号$\&$代表按位**与**运算，符号$|$代表按位**或**运算。

同时小H认为数列里的每个数都不能太大，即$0 \leq a_i < 2^k$，否则也不够“整齐”。

小H想**分别在这两种条件之一**下计算有多少种长度为n的“整齐”的数列，但这样的数列太多了，一个一个数根本数不过来。聪明的你能否告诉小H，究竟有多少种这样的数列呢？

由于答案可能很大，请输出答案对$10^9 + 7$取模后的结果。

输入描述

一行3个整数n, k, m,其中n代表数列长度、k, m的含义见题目描述。

输出描述

一行2个整数，表示分别在2种条件下，所求得的答案对$10^9 + 7$取模后的结果。

样例输入1

```
2 2 1
```

样例输出1

```
3 3
```

样例输入2

```
555 55 5555
```

样例输出2

```
19680059 206895921
```

样例解释

样例1解释：

在$0 \leq a_1, a_2 < 4$范围内：

满足$a_1 \& a_2 = 1$条件的(a_1, a_2)分别为$(1, 1), (1, 3), (3, 1)$,共3种。

满足$a_1 | a_2 = 1$条件的(a_1, a_2)分别为$(0, 1), (1, 0), (1, 1)$,共3种。

数据范围

$2 \leq n \leq 10^4, 0 \leq k \leq 63, 0 \leq m < 2^k$

HINT

关于位运算：

例如，如果想获取2^{60}的精确值，请不要使用`1<<60`而应当`1LL<<60`，因为整数常量在C语言默认为int类型（32位），但`1LL`是$long\ long$类型（64位），前者会有溢出问题。

AUTHOR：*Inf*

I - 跳棋

题目描述

由于蛙宝喜欢蹦蹦跳跳，所以Blore为他做了一款叫"小跳蛙"的跳棋。

Blore有个 $1 \times n$ 大小的棋盘，棋盘上的格子由左到右从 1 到 n 依次编号。棋盘上有一些相同的棋子。规定1表示某一位置上有棋子，0表示某一位置上没有棋子。

他可以进行两种操作：

1. 将位置为 i 棋子移到 $(i-2)$ 处，需满足条件：① $i-2 \geq 1$ ② $(i-1)$ 处存在棋子 ③ $(i-2)$ 处没有棋子。
2. 将位置为 i 棋子移到 $(i+2)$ 处，需满足条件：① $i+2 \leq n$ ② $(i+1)$ 处存在棋子 ③ $(i+2)$ 处没有棋子。

现在，给定一个棋盘的初始状态A，给定一个棋盘的目标状态B，Blore想知道他能否在有限次操作后使棋盘从初始状态变为目标状态。

输入描述

第一行一个数字 T，表示有 T 组输入

对于每一组数据：

第一行一个数 n，含义如题干所示。

第二行 n 个数（中间无空格且仅含0或1），表示棋盘的初始状态。

第三行 n 个数（中间无空格且仅含0或1），表示目标状态。

输出描述

对于每一组数据输出一行。

如果可以在有限次操作后使棋盘从初始状态变为目标状态，输出`YES`。否则输出`NO`。

样例输入

```
2
4
1110
1011
4
1110
0111
```

样例输出

```
YES
NO
```

样例说明

在样例1中，只需将位置2的棋子移动到位置4即可变为目标状态。

在样例2中，可以证明，无论怎么移动都不能变为目标状态。

数据范围

对于20%的数据，$n \leq 10$ 。

对于60%的数据，$n \leq 1000$ 。

对于100%的数据，$n \leq 10^5$，$T \leq 10$ 。

HINT

你可以使用`scanf("%1d",&x);`来读入单个数字。

AUTHOR : Blore

J - Set Studio

题目背景

拥有雄厚技术实力的42Lab，与多个高科技企业和研究机构合作，研发具有高智能高性能的人工智能载体，又称“人形”，推动着人工智能和量子技术的发展，2057年，42Lab与I.O.P.公司达成深度合作，进行次世代人工智能的开发工作，这一项目名称代号"Project Neural Cloud"。

你现在是42Lab项目所属的一位高级计算机软件工程师，人工智能的研发需要雄厚的数学基础，当前你承担的任务正是其中的一小部分，构建“人形”的“心智”对于数学基础集合论的认知，项目代号：”Set Studio”。当然这个项目是迭代性的，这只是第一次迭代，你只需要实验基于26个标准ASCII码小写字母构成的集合全集实现原子化的基本运算。

由于全集也只包括26个小写字母，所以一个集合最多只有26个元素，作为天才软件工程师的你想到了很好的并且大学的Freshman学过程序设计位运算都能理解的集合存储方式。

比如 $a \sim z$ 的 ASCII 码减去 a 的 ASCII 码的值为 $0 \sim 25$，如果将其与一个二进制数的位相关联，就可以用一个唯一确定的整数存储一个集合。

比如集合 $\{a,b,c,d\}$，写成二进制数从低到高的第$0,1,2,3$ 位，对应二进制数 $(1111)_2=(15)_{10}$，也就是该集合可以用整数 15 来存储。

然后不难用位运算定义集合的基本运算和关系，比如集合 $\{a,b\}$ 要怎么判断是集合 $\{a,b,c\}$的子集呢，在位运算的背景下需要有这样一个基本认知，如果 $A\subset B$，应该有 $A\cup B=B$，最后转换成位运算实现的集合交并运算，也就是对应整数的与或运算。

题目描述

最初的 Set Studio 由于涉及到补集运算所以定义全集 $U=\{a,..,z\}$即全体26个小写字母。

现在需要实现两大主要功能：

- 操作0：输入一个非负整数，输出其存储的集合。
- 操作1：输入两行小写串代表两个集合A,B，输入指令的代号，输入指令对应集合运算的结果，指令表如下表所示：

指令代号	集合运算
0	$A\cap B$
1	$A\cup B$
2	$\overline{A}$
3	$A-B$
4	$\overline{A\cap B}$
5	$\overline{A\cup B}$

输入描述

本题涉及到的输入输出较为复杂，请仔细阅读说明

本题有多组数据读入。

首先输入一个操作数 op：

- $op=0$，输入一个正整数代表要解析其存储集合的数。
- $op=1$，输入两个个正整数和一个整数，两个正整数代表集合 A,B 的元素个数$|A|,|B|$，整数代表运算指令代号。然后输入两行小写字母串，代表集合 A,B的元素。
- $op=-1$，运行中止程序结束。

输出描述

对于每一组数据需要输出目前的组号：如`Case 1:`

每组输出的内容分操作数讨论：

- $op=0$，输出整数对应的集合。
- $op=1$：
    - 首先判断两个集合是否相等，如果两个集合相等则输出 `Set A is equal to set B!`，不进行操作数对应的集合运算，进行下一组读入。

- 集合不相等，则判断集合 A, B 是否存在子集关系，如果 $A \subset B$ 则输出 `Set A is the subset of set B!`，反之则输出 `Set B is the subset of set A!`，如果两种情况都不是则输出 `No subset relationship!`。
- 进行集合运算后，根据运算指令代码，先输出描述文本（见下表），然后再在同一行输出运算结果的集合。

关于集合的输出格式，操作数 $op = 0, 1$通用，用大括号'{','}'括起来，中间的元素用逗号隔开，元素按照小写字母 $a \sim z$ 的顺序升序输出。如果有出现运算结果为空集的，请输出 `Empty set`。

集合运算输出描述文本表：

集合运算指令代号	输出描述文本
0	`The intersection of set A and set B is:`
1	`The union of set A and set B is:`
2	`The complement of set A is:`
3	`The difference of set A and set B is:`
4	`The complement of the intersection of set A and set B is:`
5	`The complement of the union of set A and set B is:`

样例输入1

```
0 13
1 2 2 1
ab
ba
1 2 3 0
ab
cab
-1
```

样例输出1

```
Case 1:
{a,c,d}
Case 2:
Set A is equal to set B!
Case 3:
Set A is the subset of set B!
The intersection of set A and set B is:{a,b}
```

样例输入2

```
1 4 5 0
nide
mxbjk
1 3 6 1
```

```
moa
neiusj
1 18 1 2
ajhiowkmlzxcqegfdv
a
1 4 5 3
abcd
cdefg
1 3 4 4
jaw
jdwa
1 7 10 5
wajdokl
zciefjplmn
-1
```

样例输出2

```
Case 1:
No subset relationship!
The intersection of set A and set B is:Empty set
Case 2:
No subset relationship!
The union of set A and set B is:{a,e,i,j,m,n,o,s,u}
Case 3:
Set B is the subset of set A!
The complement of set A is:{b,n,p,r,s,t,u,y}
Case 4:
No subset relationship!
The difference of set A and set B is:{a,b}
Case 5:
Set A is the subset of set B!
The complement of the intersection of set A and set B is:
{b,c,d,e,f,g,h,i,k,l,m,n,o,p,q,r,s,t,u,v,x,y,z}
Case 6:
No subset relationship!
The complement of the union of set A and set B is:{b,g,h,q,r,s,t,u,v,x,y}
```

样例解释

样例一：

- 第一组数据 $13 = (1101)_2$，对应集合$\{a, c, d\}$
- 第二组数据：都表示集合 $\{a, b\}$，集合相等所以不再进行集合运算
- 第三组数据：集合 $A = \{a, b\}$ 是集合 $B = \{a, b, c\}$ 的子集，有 $A \subset B$，集合A, B对应的整数是3 和 7，做与运算 $3\&7 = 3$，于是有 $A \cap B = \{a, b\}$。

样例二则是涵盖了6种集合运算，并且涵盖了集合子集关系和运算为空集的可能。

数据范围

- 组数：$1 \leq k \leq 5000$
- $op = 0$ 时读入的整数 $0 \leq s \leq 2^{26} - 1$

- $op = 1$ 时：
    - 读入两个集合的元素个数：$1 \leq |A|, |B| \leq 26$（保证不会读入空集）
    - 两行小写串不保证小写字母按$a \sim z$升序排列，但是保证符合集合的元素唯一性，不会出现重复的小写字母。

HINT

关于读入，并不想卡大家，所以在此给出相关代码结构，避免你们纠结换行符与字符读入的问题，请直接复制该结构然后在此基础上写程序的功能：

特别是不要纠结为啥要用两个 `getchar()` ，因为 Windows 生成的数据输出文件换行符是 `\n\r` 所以需要把这两个换行符都读进去才能保证下一行的小写字母能正常读取。

```
char elem;
unsigned sa, sb;
while (~scanf("%d", &op))
{
    if (op == 0) scanf("%u", &s);
    else if (op == 1)
    {
        scanf("%d%d%d\n", &na, &nb, &set_op);
        sa = 0, sb = 0;
        for (int i = 0; i < na; ++i) { scanf("%c", &elem); /* your code*/ }
        getchar();
        getchar(); // 因为Windows生成数据换行符是\n\r需要两个getchar()
        for (int i = 0; i < nb; ++i) { scanf("%c", &elem); /* your code*/ }
    }
    // your code
}
```

需要注意的一点特别是取反运算时，由于不存在 26 位的整数要么 32 或者 64 位，要考虑避免高于 26 位的二进制位置为 1 时对集合输出的干扰。

彩蛋：事实上学完数据结构后，各位就能理解Set Studio第二次迭代的工作，通过引入数据结构（栈、字符串）来处理能通过逆波兰表达式表示的更具一般性的任何集合运算表达式。

AUTHOR：*Shederay*