# C4 - 2021级程序设计基础第四次上机

# A - 成绩分段打印

## 题目描述

输入一个成绩$score$，按照如下要求输出：

- 当$90 \leq score \leq 100$时，输出$A$；
- 当$80 \leq score < 90$时，输出$B$；
- 当$70 \leq score < 80$时，输出$C$；
- 当$60 \leq score < 70$时，输出$D$；
- 当$0 \leq score < 60$时，输出$F$。

## 输入描述

一行一个正整数$score$表示成绩。

## 输出描述

一行一个字符，要求见题目描述。

## 输入样例1

```
78
```

## 输出样例1

```
C
```

## 输入样例2

```
81
```

## 输出样例2

```
B
```

## 数据范围

对于$100\%$的数据，$0 \leq score \leq 100$。

---

# B - 天数

## 题目描述

某只蒟蒻想要知道某一年的某一个月有多少天QAQ。

## 输入格式

第一行，一个整数$T(1 \leq T \leq 10^4)$，表示数据的组数。

接下来$T$行，每行两个整数$y, m$表示年份和月份，$2000 \leq y \leq 9999, 1 \leq m \leq 12$。

## 输出格式

$T$行，每行一个整数，第$i$行表示第$i$组数据的$y_i$年$m_i$月的天数。

## 样例输入

```
2
2000 2
2001 5
```

## 样例输出

```
29
31
```

## HINT

记得判断闰年~

---

# C - 一天不用switch我浑身难受

## 题目描述

请在不改变下列程序功能的情况下，将下列代码的`if-else`结构重写为`switch`结构，提高程序的性能并提交代码。

```
#include<stdio.h>
unsigned seed;
long long a, b;
long long ans = 0;
int n, i;
int op;
int scheme[35];
int main() {
    scanf("%d%u", &n, &seed);
    for (i = 0; i <= 31; ++i) scanf("%d", &scheme[i]);
    while(n--) {
    seed ^= seed << 13;
    seed ^= seed >> 17;
    seed ^= seed << 5;
```

```
    op = scheme[seed & 31] & 31;
    a = n ^ 114514191981011;
    b = n ^ 191981011451411;
    if (op == 0) ans += a + b;
    else if (op == 1) ans += a - b;
    else if (op == 2) ans += a * b;
    else if (op == 3) ans += a / b;
    else if (op == 4) ans += a % b;
    else if (op == 5) ans += a & b;
    else if (op == 6) ans += a | b;
    else if (op == 7) ans += a ^ b;
    else if (op == 8) ans -= a ^ b;
    else if (op == 9) ans -= a | b;
    else if (op == 10) ans -= a & b;
    else if (op == 11) ans -= a % b;
    else if (op == 12) ans -= a / b;
    else if (op == 13) ans -= a * b;
    else if (op == 14) ans -= a - b;
    else if (op == 15) ans -= a + b;
    else if (op == 16) ans &= a * b;
    else if (op == 17) ans &= a / b;
    else if (op == 18) ans &= a + b;
    else if (op == 19) ans &= a - b;
    else if (op == 20) ans &= a | b;
    else if (op == 21) ans &= a ^ b;
    else if (op == 22) ans &= a % b;
    else if (op == 23) ans &= a & b;
    else if (op == 24) ans ^= a / b;
    else if (op == 25) ans ^= a * b;
    else if (op == 26) ans ^= a - b;
    else if (op == 27) ans ^= a + b;
    else if (op == 28) ans ^= a ^ b;
    else if (op == 29) ans ^= a | b;
    else if (op == 30) ans ^= a & b;
    else if (op == 31) ans ^= a % b;
    }
    printf("%lld", ans);
    return 0;
}
```

**在极少数情况下，由于评测机会飘，如果优化后仍TLE，请尝试重新提交几次，就可以了。**

以下样例作为测试对拍，可以检查你改完的程序是否破坏了程序的正确性。

# 样例输入

```
3 8
0 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 26 27 28 29 30 31
```

# 样例输出

```
-3469337213682
```

## HINT

最先提醒一些同学：**可别从头写一遍了，代码都给出来了，直接复制粘贴到IDE上再慢慢改他不香吗?**

**提醒：不要忘记break!**

如果有幸能学到计组，会得知switch和if-else的底层实现方式不同，switch编译到汇编层面用到了“跳转表”的结构，保证了对于每一种情况能直接偏移到该情况的汇编代码头，而不是经过冗长的if语句判断跳来跳去。当然，不排除你自己本地的编译器可能足够聪明，能够把冗长乏味的if优化成底层跳转表，学校这里的评测姬做不到呢。

注：本题没有在为*Nintendo*打广告，因为开花学长没有买*Switch*。

*p.s.* 题目思路来源于胡珽

*Author*：计组*OO*两开花

---

# D - 成绩分析

## 题目描述

老师要统计C3上机的情况。

已知有3个班级，共$N$名学生。

三个班级的编号为$A$，$B$和$C$。每班有若干位学生（测试数据保证每班人数至少1人，至多50人）。

现在给出每位同学的班级和成绩，请问：

（1）哪个班平均分最高？请输出班号。

（2）A,B,C三个班的最高分和最低分分别是多少?

## 输入描述

第一行为一个正整数$N(1 \leq N \leq 150)$

接下来$N$行。

每行第一个输入为字符$A$，$B$或$C$，表示班级。

每行第二个输入为一个整数$X$，表示该同学的分数。（$0 \leq X \leq 100$）

## 输出描述

共4行。

第一行输出一个字符，为$A$，$B$或$C$，表示平均分最高的班级。（数据保证班级平均分不相同）

第二行有两个由空格分开的整数，表示$A$班的最高分和最低分。

第三、四行分别表示$B$班和$C$班的情况，格式与第二行相同。

## 样例输入

```
6
A 99
A 100
B 99
B 97
C 100
C 100
```

## 样例输出

```
C
100 99
99 97
100 100
```

*AUTHOR*：Mentor_D

---

# E - Black and White

## 题目描述

某只蒟蒻得到了一个由小写字母`w`、`b`组成的字符串，ta觉得可爱的字符串由`m`个连续的`w`和`n`个连续的`b`组成，`m`和`n`均可为$0$。

如`wwb`、`wbbb`都是可爱的，`wbw`、`bww`都是不可爱的。

**注意：w、b的数量m、n都可以为0。** 如`wwww`、`bb`等都是可爱的字符串。

给你一个由小写字母`w`、`b`组成的字符串，请判断字符串是不是可爱的。

## 输入格式

一行，由小写字母`w`、`b`组成的字符串，字符串长度$\leq 100$。

## 输出格式

如果这个字符串是可爱的，输出`"Yes"`，否则输出`"No"`，输出均不含引号。

## 样例输入1

```
wwwbw
```

## 样例输出1

```
No
```

## 样例输入2

```
wwwbbb
```

## 样例输出2

```
Yes
```

---

# F - 愿此行，终抵群星

## 题目背景

我们的目标是星辰大海。

## 题目描述

三月七是一个喜欢夜观天象的女孩，她为每个星星都用$1, 2, \cdots, n$编上了号，但她总是为自己记不下来每个星星的位置而苦恼。

某天她突发奇想，如果把夜空看作一个二维平面，以某处为原点建立平面直角坐标系$xOy$的话，那么每个星星都会有唯一的坐标表示。

但是星星的数量是如此之多，以至于她仍然记不住。于是，她找你帮忙看看，这些星星的坐标能否用一个关于$x$并且次数不超过$n-1$次的多项式表达出来。

## 输入描述

第一行一个整数$t$，表示有多少组数据。

对于每组数据：

第一行一个正整数$n$；

接下来$n$行，每行两个整数$x_i, y_i$，表示第$i$个星星的坐标表示，两个数之间用一个空格分开。

## 输出描述

每组数据输出一行。

对于第$k$组数据：

首先输出`Case #k:`，其中$k$为数据组数。

紧接着，如果能够用一个关于$x$的不超过$n-1$次的多项式表示，则输出`Through the star sea.`；否则输出`Stop somewhere.`。

## 样例输入

```
3
3
1 0
2 1
3 4
4
-2 -26
-1 -10
1 -2
```

```
2 2
6
1 4
3 128
5 1844
5 11248
9 42884
11 123584
```

## 样例输出

```
Case #1: Through the star sea.
Case #2: Through the star sea.
Case #3: Stop somewhere.
```

## 样例解释

第一组数据的多项式：$f(x) = x^2 - 2x + 1$

第二组数据的多项式：$f(x) = x^3 - 2x^2 + 3x - 4$

第三组数据无法用$n-1$次多项式表示。

## 数据范围

对于$100\%$的数据：$t \leq 100, n \leq 1000, -10^6 \leq x_i, y_i \leq 10^6$。

数据保证任意两个点的坐标不会重合；所有坐标均按照$x$从小到大顺序给出；当$x$相同时，按$y$从小到大给出。

## HINT

三月七教你学线性代数：

一个关于$x$的$n-1$次多项式$f(x) = \sum_{i=0}^{n-1} a_i x^i$对于任意横坐标$x$，有且仅有一个唯一确定的$y = f(x)$与之相对应，而不会出现相同的横坐标但纵坐标不同的情况，**因此只要有$n$个不同的横坐标就能确定一个次数不超过$n-1$次的多项式。**

温馨提示：

- 注意输出格式。
- 因为只需要判断存在性，所以直接使用不求系数的做法即可。

*AUTHOR: Stockholm*

---

# G - 炸弹人

## 题目背景

《炸弹人》是一款经典FC游戏，玩家可以操控角色在屏幕中移动，并放置炸弹消灭一定区域内的敌人。不过本题中的炸弹不同于原版，它十分强力，范围巨大。

# 题目描述

平面直角坐标系中有$n$个敌人，他们的坐标已知，且不会移动，消灭第$i(i=1,2,\ldots,n)$个敌人可以获得$s_i$积分。

玩家分别在$m$个坐标放置了"强力炸弹"，每个炸弹都会消灭以之为中心的"十字"中的所有敌人（也就是说，所有与炸弹**横坐标相等或纵坐标相等**的敌人都被消灭了）。请编程求解玩家一共获得了多少分。

# 输入描述

第一行两个正整数$n, m$表示**敌人**数量和**炸弹**数量。

接下来$n$行，每行三个整数表示第$i$个**敌人**的横坐标$x_i$，纵坐标$y_i$，和对应分数$s_i$。

再接下来$m$行，每行两个整数表示第$i$个**炸弹**的横坐标$x_i'$，纵坐标$y_i'$。

# 输出描述

一行一个整数表示总得分。

# 样例输入1

```
4 2
0 0 1
0 1 1
1 0 1
1 1 1
2 0
0 2
```

# 样例输出1

```
3
```

# 样例输入2

```
4 2
0 0 1000000000
0 1 1000000000
1 0 1000000000
1 1 1000000000
2 0
0 2
```

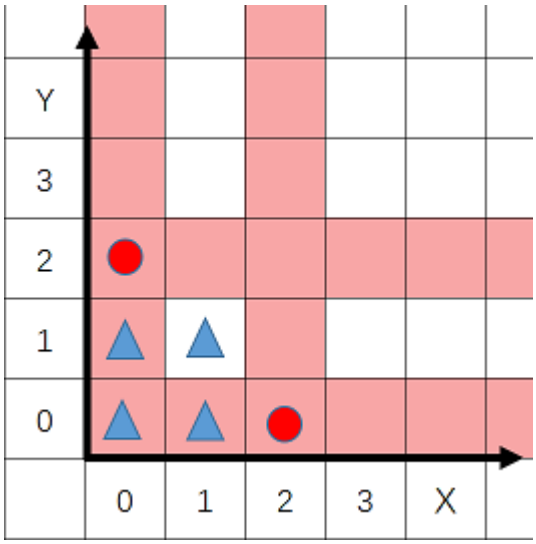
# 样例输出2

```
3000000000
```

# 样例解释

第一个炸弹可以消灭$(0,0),(1,0)$两个敌人，第二个炸弹可以消灭$(0,0),(0,1)$两个敌人。故除去$(1,1)$处的敌人外，其余3个敌人均被消灭。

可以配合示意图食用（三角为敌人，圆形为炸弹，红色为轰炸区域）




## 数据范围

$1 \leq n,m \leq 1000, 0 \leq x_i,y_i,s_i \leq 10^9$

## HINT

本题的一种思路：用第一层循环枚举敌人。对于每个敌人，用第二层循环枚举炸弹，统计该敌人是否被轰炸。

*AUTHOR:inf*

# H - 有效字符

## 题目描述

在可打印字符中（$ascii$码值为$32-126$），称字符$0-9, a-z, A-Z$为**有效字符**，其余的称为**无效字符**。定义**语句**为有着特定格式的字符串，该字符串以**有效字符**开头，以遇到的第一个句号(.)、感叹号(!)或问号(?)结尾，不满足该格式的字符串不能算作一条语句。现有一行只含有可打印字符的字符串，请你找到当中**有效字符数最多**的语句（当两个语句有效字符数一样时选择**靠前**的语句）。

## 输入描述

一行输入，为待处理的字符串。

## 输出描述

三行，输出有效字符数最多的语句的信息

第一行为该语句是第几条语句

第二行为该语句的有效字符数

第三行为该语句的内容

## 样例输入

```
emm...  This example 1s nonsence.
```

## 样例输出

```
2
21
This example 1s nonsence.
```

## 数据范围

- 设字符串总共有 $n$ 个字符， $2 \leq n \leq 10^6$ ，并且至少含有一条语句。

## HINT

- 在进行字符串操作时注意结尾符 `'\0'` 的问题
- 也许你会用到 `string.h` 中的库函数：
    - `char *strcpy(char *dest, const char *src)` ：把 src 字符串（包括结尾符）复制到 dest 中
    - `unsigned int strlen(const char *str)` ：计算字符串 str 的长度，直到结尾符，但不包括结尾符。

*AUTHOR: toush1*

---

# I - Monica的羊

## 题目描述

在数了一整晚的星星之后，$Monica$的数数能力大大提升了。

现在$Monica$准备在羊圈中挑选一只羊加入她的梦境中，可是每只羊看起来都差不多，于是$Monica$选择用排除法把一些羊排除掉。

$Monica$给每只羊都赋予了一个唯一的编号，编号从1到$n$，并且让它们按编号顺序首尾相接围成一圈。一开始，编号为1的羊的下一只羊编号为2，其他的依此类推。特别的，编号为$n$的羊的下一只羊编号为1。$Monica$从编号为$n-1$的羊开始数，每数到第$k$只羊就让它出圈，直到只剩下最后一只羊。

## 输入描述

第一行两个整数$n, k$。

## 输出描述

输出一行一个整数$x$。表示最后剩下的羊的编号。

## 样例输入1

```
3 2
```

## 样例输出1

```
1
```

## 样例输入2

```
15 4
```

## 样例输出2

```
11
```

## 样例输入3

```
605 24
```

## 样例输出3

```
281
```

## 样例解释

第一个样例中

$2 \to 3$，编号为3的羊出圈。

$1 \to 2$，编号为2的羊出圈。

最后剩下编号为1的羊。

## 数据范围

对于$100\%$的数据

$2 \leq n \leq 10000$

$2 \leq k \leq 100$

## Hint

如何表示一只羊出局了呢，想想之前C3-G中的技巧。

*AUTHOR: Monica*

---

# J - HDT

## 题目描述

Blore想写一个酒馆战棋战斗阶段的模拟器，不过为了减少大家的码量，他对模型做了一定的简化（和实际游戏的内容有些不同）。

模拟战斗阶段的过程可以归纳为以下几步：

1. **初始化**：总共有两名玩家参与战斗，记为玩家A和玩家B。每名玩家最多拥有7位随从，每个随从有且仅有两个属性值，为**攻击力**和**血量**。
2. **判定游戏结束**：如果有一名玩家**所有随从死亡**则游戏结束。如果结束时**玩家A和玩家B**所有随从均死亡输出 `Draw!` ，否则，如果**玩家B**所有随从均死亡输出 `A Win!`，**玩家A**所有随从均死亡输出 `B Win!`
3. **攻击随从和被攻击随从的确定**：攻击方（**初始为玩家A**）的左起当前**没有进行过攻击**的并且**未死亡**的随从随机攻击一名敌方随从（**请使用题目提供的代码片段确定被攻击的目标**，具体使用方法见下面 攻击目标确定方法 部分）

   如果攻击方所有随从都进行过攻击，那么从左数第一个存活的随从重新开始新一轮攻击。
4. **攻击结束**：记发动攻击的随从攻击力和血量为$ATK_\alpha, HP_\alpha$，被攻击的随从攻击力和血量为$ATK_\beta, HP_\beta$，则一轮攻击后**两个随从均受到伤害**，即

$$HP_\alpha = \max(0, HP_\alpha - ATK_\beta)，\ HP_\beta = \max(0, HP_\beta - ATK_\alpha)$$

注：若某个随从血量降为0（死亡），则会被移除，不参与后续战斗阶段，**不能成为发起攻击的随从，不能被指定为攻击的目标，不计入攻击目标的需求变量。**

5. **进入下一轮次**：交换攻击方（例如当前发起攻击的为玩家A，那么下一次玩家B发起攻击），返回第（2）步。

**攻击目标确定方法**：

（请把下面的内容**复制**到提交的代码中，确保**仅在随从要发起攻击**时灵活使用，请勿主动修改变量Next的值）

```
/*
* 需求变量: 被攻击方剩余随从数量leftMinion（不包含死亡随从）
* 得出结果: 被攻击的目标随从位置
*/
unsigned long Next = 1;
int main()
{
    //do something...
    Next = Next * 810975 + 922768;
    goal=((unsigned)(Next / 65536) % leftMinion + 1);
    //do somrthing...
    return 0;
}
// 由于被攻击方剩余随从数量为7，故leftMinion的值为7
// goal的值为6，表示被攻击方 存活着的 左数第6个随从将被攻击
```

# 输入

第一行一个数$T$，表示有 $T$ 组数据

对于每组数据

第一行，一个数$N$，表示玩家A有$N$个随从。

第二行到第 $(N+1)$ 行（共$N$行），每行两个数$ATK_i$，$HP\_i$，表示玩家A从左至右第$i$个随从的攻击力和血量。

第 $(N+2)$ 行，一个数$M$，表示玩家B有$M$个随从。

第 $(N+3)$ 行到第 $(N+M+2)$ 行（共$M$行），每行两个数$ATK_j$，$HP\_j$，玩家B从左至右第 $j$ 个随从的攻击力和血量。

# 输出

对于每一组数据输出一行，如果结束时玩家A和玩家B所有随从均死亡输出`Draw!`，否则：如果玩家B所有随从均死亡输出`A Win!`，玩家A所有随从均死亡输出`B Win!`

# 输入样例

```
2
2
10 10
10 10
3
10 10
1 1
10 10
2
10 10
10 10
3
10 10
1 1
10 10
```

# 输出样例

```
B Win!
Draw!
```

# 样例说明

第1轮：A 发起攻击 1 -> 3

第2轮：B 发起攻击 1 -> 2

此时游戏结束，A随从全部死亡，输出`B Win!`

第1轮：A 发起攻击 1 -> 1

第2轮：B 发起攻击 2 -> 2

第3轮：A 发起攻击 2 -> 3

此时游戏结束，A、B随从全部死亡，输出`Draw!`

其中确定攻击目标的代码片段被调用了五次，下面是这五次调用的部分变量的值：

```
leftMinion = 3, goal = 3
leftMinion = 1, goal = 1
leftMinion = 3, goal = 1
leftMinion = 1, goal = 1
leftMinion = 1, goal = 1
```

## 数据范围：

$1 \leq T \leq 10, 1 \leq N, M \leq 7, 1 \leq ATK, HP \leq 100$

*Author*：*Blore*