#A、链表合并

题目描述

给定两个单链表

一个节点数量为 n

$$L_1=a_1 \rightarrow a_2 \rightarrow \ldots\ldots \rightarrow a_{n-1} \rightarrow a_n$$

一个节点数量为 m

$$L_2=b_1 \rightarrow b_2 \rightarrow \ldots\ldots \rightarrow b_{m-1} \rightarrow b_m$$

如果 $n \geq m$,你的任务是将比较短的那个链表逆序,然后将之并入比较长的那个链表,得到一个形如 $a_1 \rightarrow a_2 \rightarrow b_m \rightarrow a_3 \rightarrow a_4 \rightarrow b_{m-1}$ ……的结果

例如给定两个链表分别为 $6 \rightarrow 7$ 和 $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$,合并结果为 $1 \rightarrow 2 \rightarrow 7 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 5$

输入格式

输入首先在第一行中给出两个链表 L_1 和 L_2 的头结点的地址,以及正整数 $N(\leq 10^4)$,即给定的结点总数

一个结点的地址是一个 5 位数的非负整数,空地址 `NULL` 用 `-1` 表示。

随后 N 行,每行按以下格式给出一个结点的信息:

```
Address Data Next
```

其中 `Address` 是结点的地址,`Data` 是不超过 10^5 的正整数,`Next` 是下一个结点的地址。题目保证没有空链表,并且较长的链表至少是较短链表的两倍长

输出格式

按顺序输出结果链表,每个结点占一行,格式与输入相同

输入样例

```
00100 01000 7
02233 2 34891
00100 6 00001
34891 3 10086
01000 1 02233
00033 5 -1
10086 4 00033
00001 7 -1
```

输出样例

```
01000 1 02233
02233 2 00001
00001 7 34891
34891 3 10086
10086 4 00100
00100 6 00033
00033 5 -1
```

本体思路为：首先读入两个链表，分别建立出来；随后只需按题目描述((i(链表一目前元素位置)+1)%2=0 且 j(链表二目前元素位置)>0)的顺序建立出第三个链表再输出即可

#D、Wilks Point

题目描述

Wilks\ Coefficient*Wilks Coefficient* 是一个可用于计算不同体重、性别的力量训练者的力量水平的计算工具

目前 IPF(international powerlifting federation)*IPF(international powerlifting federation)*在计算全场最佳运动员的时候都会使用，计算的方式是输入性别、体重

Wilks\ Point*Wilks Point* 是使用 Wilks\ Coefficient*Wilks Coefficient* 乘上按照 IPF*IPF* 标准的深蹲、卧推以及硬拉的总成绩计算得到

Wilks\ Coefficient*Wilks Coefficient* 的计算方法如下

$$\text{Wilks\ Coefficient} = \frac{500}{a+bx+cx^2+dx^3+ex^4+fx^5}$$

Wilks Coefficient=$\frac{500}{a+bx+cx^2+dx^3+ex^4+fx^5}$

其中 xx 为体重的KGKG 数值,abcdefabcdef 的数值根据性别从下表选取

	Male	Female
a	-216.0475144	594.31747775582
b	16.2606339	-27.23842536447
c	-0.002388645	0.82112226871
d	-0.00113732	-0.00930733913
e	0.00000701863	0.00004731582
f	-1.291e-8	-9.054e-8

-

当选手的 Wilks\ Point*Wilks Point* 分数在[0,200)[0,200)内,称之为 `Un-trained`

-

-

当选手的 Wilks\ Point*Wilks Point* 分数在[200,238)[200,238),称之为 `Novice`

-

-

当选手的 Wilks\ Point*Wilks Point* 分数在[238,326)[238,326),称之为 `Intermediate`

-

-

当选手的 Wilks\ Point*Wilks Point* 分数在[326,414)[326,414),称之为 `Advanced`

-

-

当选手的 Wilks\ Point*Wilks Point* 分数不低于414414,称之为 `Elite`

-

输入格式

第一行输入一个字符 `F` 或者 `M` 表示性别

第二行输入一行三个实数表示卧推、深蹲、硬拉的 KG*KG* 数值(三个实数 $0 \sim 5000$ ~ 500)

第三行输入一个实数表示体重 KG*KG* 数值(体重不超过 $30 \sim 20030$ ~ 200)

输出格式

第一行输出一个整数 Wilks\ Point*Wilks Point* 向上取整的结果

第二行输出 Wilks\ Point*Wilks Point* 对应的称号

输如样例 1

```
M
100 187.5 180
75
```

输出样例 1

```
334
Advanced
```

输如样例 2

```
F
57.25 80 90
60
```

输出样例 2

```
254
Intermediate
```

这道题其实很简单，只需要按照他的步骤，用一个 switch 语句，分两种情况讨论，一种为男生，用多个变量来计算后输出；另一种则为女生，只需要将男生里的数值改变一下，计算后输出即可。

（总结：就是一道模拟题，高级一点可以学习超哥用数组，过分简单，就不贴主要代码了）。

#C、迷失的牛

题目描述

FJ*FJ* 弄丢了奶牛贝茜，他需要找到她！

幸运的是，农场只有一条长长的路，FJ*FJ* 知道贝茜一定在这条路上的某个地方。如果我们把路径想象成一条数轴，那么FJ*FJ* 目前处于位置 x*x* 贝茜目前处于位置 y*y*(FJ*FJ* 不知道)。如果FJ*FJ* 知道贝茜在哪里，他可以直接走|x-y| ∣ *x*−*y* ∣ 到她身边。不幸的是，外面很黑，FJ*FJ* 什么也看不见。他能找到贝茜的唯一方法就是来回走动，直到他最终到达她的位置。

为了找出在搜索中来回走动的最佳策略，FJ*FJ* 查阅了计算机科学研究文献，有趣的是，这个确切的问题不仅在过去被计算机科学家研究过(The\ Lost\ Cow\ Problem*The Lost Cow Problem*)。

这个解决方案是先移动到位置 x+1*x*+1，然后反向并移动到位置 x−2*x*−2，然后移动到 x+4*x*+4，依此类推。正如他在研究解决丢失的奶牛问题的算法时所读到的那样，这种方法保证了他在最坏的情况下移动 99 倍的|x-y| ∣ *x*−*y* ∣ 。

输入格式

输入 x,y(0 \leq x,y \leq 1000)*x*,*y*(0≤*x*,*y*≤1000)

输出格式

输出 FJ*FJ* 移动的总距离

输入样例 1

```
3 6
```

输出样例 1

```
9
```

样例解释 1

3 \rightarrow 4 \rightarrow 1 \rightarrow 63→4→1→6 一共移动了 99 个距离

输入样例 2

```
3 19
```

输出样例 2

```
46
```

样例解释 2

3 \rightarrow 4 \rightarrow 1 \rightarrow 7 \rightarrow -5 \rightarrow 193→4→1→7→−5→19 一共移动了 99 个距离

根据题意，我们只需模拟 F“J 跳”来“跳”去的过程，并在每一次“跳”后储存“跳”的距离，每一次“跳”后都需要判断是否“跳”过头，如果跳过头那就减去“跳”过头的距离。

#B、充电器

题目描述

小苏有两部手机和一个充电器,两部手机开始分别有 $a1$ 和 $a2$ 的电量

小苏能够在任一分钟开始时将充电器连接到任一部手机上

每分钟,手机要消耗 2 的单位电量(如果没连接到充电器)或充 1 的单位电量(如果连接到充电器)

小苏来玩游戏,如果两部手机的电量都是正值,那么游戏将一直进行下去

如果某分钟开始时,一部手机的电量是 1,那么它必须连接充电器,否则游戏结束

如果某部手机的电量是 0,那么游戏也立即结束

你的任务是确认游戏最长能持续多少时间

> *游戏进行中两部手机都必须工作,而且不能暂停或关机,允许手机的电量超过 100*

输入格式

仅一行,两个整数 $a1$ 和 $a2$,分别表示两部手机开始的电量

输出格式

一个整数,表示游戏最长的持续时间

输入样例 1

```
3 5
```

输出样例 1

```
6
```

样例解释 1

开始两部手机电量分别为 3,5

前第 1 分钟,充电器接第 1 部手机电量分别为 4,3

前第 2 分钟,充电器接第 1 部手机电量分别为 5,1

前第 3 分钟,充电器接第 2 部手机电量分别为 3,2

前第 4 分钟,充电器接第 2 部手机电量分别为 1,3

前第 5 分钟,充电器接第 1 部手机电量分别为 2,1

前第 6 分钟,充电器接第 2 部手机电量分别为 0,2 游戏结束

输入样例 2

```
4 4
```

输出样例 2

```
5
```

数据范围

对于 100%的数据 1<=a1,a2<=100

对于这道题，我们可以使用模拟法，一步步推。我们可以用 while 来实现。如果 a1,a2（以下称之为 a,b）都大于 0，就继续执行，否则退出。每循环一次，计数器就加一。还需要加条件判断。因为要使使用时长最多，所以如果 a 手机电量小于 b，充电器就给 a 充。反之 b 充。a 充时 a 手机电量+1，b 手机电量-2，反之 b 手机电量+1，a 手机电量-2。当 a，b 都<=1 时，break 以跳出 while，输出计数器，结束程序。

值得一提的是，这个程序有特判分 10 分。需要在输入时判断 a，b 是否都是 1，如果是，直接输出 0。因为这种情况下一部手机只能撑一分钟不到。

贴条件判断部分代码：

（c 表示的是充电器连接到的手机，0 等于连接 a，1 等于连接 b）