# 14级第三次练习赛解题报告

A新安当的神笔

# 题目描述：

新安当这天来了个稀客，没错就是传说中没节操水平超越了延帝的天臧散人，Arthur。  
老板景天感到非常荣幸，忍痛拿出了李三思家祖传的铁观音来请Arthur，作为答谢，Arthur掏出一支狼毫，在地上画了个惟妙惟肖的菱形，唰的一闪，地上出现了一枚等大的钻石。  
看着呆若木鸡口水直流的景天，Arthur微微一笑“景老板请吧……”

# 输入：

一组测试数据。  
第一行为一个整数n(0≤n≤20)，表示钻石的数目.  
接下来n行，每一行一个整数，为每块钻石的尺寸。

# 输出：

对于每个尺寸，输出对应的钻石，用\*填充，具体形式参见输出样例。

# 输入样例：

3  
2  
3  
5

# 输出样例：

\*

\*\*\*

\*

\*

\*\*\*

\*\*\*\*\*

\*\*\*

\*

\*

\*\*\*

\*\*\*\*\*

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

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

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

\*\*\*\*\*

\*\*\*

\*

分析：

这道题目和我们之前的“猫粮”“狗粮”是有共同特点的；我们可以用嵌套的for语句将 空格 与 \* 分别的打出。

关键是找到规律：

我们先要输入a;(表示有几组数据这时可以使用while(a--) )接下来输入n，发现结果总共有2\*n-1行；我们将其分为上下两大部分，然后再将上面（一共有n行）分为左（空格 \*）右（\*）部分，下面（一共有n-1行）分为左（空格 \*）右（\*）部分进行for的控制语句；

这样我们就可以找到规律。C:\Users\ADMINI~1\AppData\Local\Temp\SGPicFaceTpBq\48060\0652B9DD.gif

A代码：

#include<iostream>

using namespace std;

int main()

{

int a;

cin>>a;

while (a--)

{

int n;

cin>>n;

for(int i=1;i<=n;i++)

{

for(int j=1;j<=n-i;j++)

cout<<" ";

for(int j=1;j<=i;j++)

cout<<"\*";

for(int j=i-1;j>0;j-- )

cout<<"\*";

cout<<endl;

}

for(int i=n-1;i>=1;i--)

{

for(int j=1;j<=n-i;j++)

cout<<" ";

for(int j=1;j<=i;j++)

cout<<"\*";

for(int j=i-1;j>0;j-- )

cout<<"\*";

cout<<endl;

}

}

}

B逃出迷宫

### Problem Description

小蛮和龙幽两人在江湖上行侠仗义，锄强扶弱。老百姓非常感谢他们，但他们同时也结下了不少仇家。很明显，这些仇家都不是什么见得了人的人物，干得全都是下流手段。某天他们向小蛮和龙幽下了战帖，说要来一场光明正大的决斗，如果他们赢了，以后小蛮和龙幽就不得再干涉他们的事。龙幽觉得其中有诈，刚想拒绝。可是任性的小蛮却一口答应了下来，龙幽（一 一+）……  
等他们到了约定的地方却发现对方没有按时到，龙幽敏锐的察觉到事情不对，拉着小蛮就要离开，可这时虚空中传来了声音：“既来之，则安之。何必这么急着走呢……这个阵法是专门为你们留的，找不到破阵之法，你们就将永远被困在里面了，哈哈哈哈……”小蛮大声斥责他们无赖，龙幽无奈的回答“他们本来就是无赖(─.─|||，我们还是想办法出去吧”说完开始研究这个阵法……  
经过一番研究，龙幽发现此阵很奇特，每一步都得走特定的步数，一步错则得从头开始。经过一番尝试，他发现前几步满足下面这个数列……  
第一项：1 第二项：2 第三项：5 第四项：26 第五项：677••••••  
现在请你帮助他们逃出这座迷阵吧……

### Input

输入包含多组数据，每组数据为一行，为一个整数n（1<=n<=100）。

### Output

对于每一个n，输出他们第n步需要走的步数。现在只要求你给出这个步数的十位数的值。

### Sample Input

3

5

6

### Sample Output

0

7

3

分析：

第一项：1 第二项：2 第三项：5 第四项：26 第五项：677••••••

这样为我们看出（n>=2）后一项是前一项平方+1；因此可以利用循环将每个数求出，但是我们要求的是第n项的十位数；况且运算量很大，我们要在循环里加上条件使其不影响结果又能不超过运算范围；我们可以想到每次只要这个数的最后两项就可以了；因此可以利用这个数的后两位进行循环，只是要考虑这个数是不是大于100就行了，然后%100。输出结果时要先/10在%10就ok了。

B代码

#include<iostream>

using namespace std;

int main()

{

int n;

while(cin>>n)

{

long long x=1;

for(int i=1;i<=n-1;i++)

{

x=x\*x+1;

if(x>100)

x=x%100;

}

if(x<10)

cout<<"0"<<endl;

else

cout<<(x/10)%10<<endl;

}

}

C召唤英灵

# 题目描述

第X次圣杯战争即将拉开，作为参与其中的魔术师之一，Darling首先要做的，是绘制召唤法阵。  
已知绘制法阵需要若干种化学元素，原本这对于Darling并非什么难事，但是Darling近期有点忙，因此她来找你帮忙。  
你的任务很简单，给你所有原子序数，请你将Darling需要的元素名称写下来并交给化学药品店。

# 输入

一组测试数据。  
第一行包含一个数n（1≤n），表示Darling绘制法阵需要的元素个数。  
第二行包含n一个整数，用空格隔开，为每种元素对应的原子序数。

# 输出

输出一行，为所需的全部元素名称（化学符号），元素与元素之间用一个空格隔开。（如果元素不在前四周期，则该元素名称为Usl）。  
具体参见输出样例

# 输入样例

3  
1 17 50

# 输出样例

H Cl Usl

分析：

这道题用switch就可以，只是。。。~~~~(>\_<)~~~~ 

C代码

#include<iostream>

using namespace std;

int main()

{

int n;

cin>>n;

while(n--)

{

int a;

cin>>a;

switch(a)

{

case 1: cout<<"H"<<" ";break;

case 2: cout<<"He"<<" ";break;

case 3: cout<<"Li"<<" ";break;

case 4: cout<<"Be"<<" ";break;

case 5: cout<<"B"<<" ";break;

case 6: cout<<"C"<<" ";break;

case 7: cout<<"N"<<" ";break;

case 8: cout<<"O"<<" ";break;

case 9: cout<<"F"<<" ";break;

case 10: cout<<"Ne"<<" ";break;

case 11: cout<<"Na"<<" ";break;

case 12: cout<<"Mg"<<" ";break;

case 13: cout<<"Al"<<" ";break;

case 14: cout<<"Si"<<" ";break;

case 15: cout<<"P"<<" ";break;

case 16: cout<<"S"<<" ";break;

case 17: cout<<"Cl"<<" ";break;

case 18: cout<<"Ar"<<" ";break;

case 19: cout<<"K"<<" ";break;

case 20: cout<<"Ca"<<" ";break;

case 21: cout<<"Sc"<<" ";break;

case 22: cout<<"Ti"<<" ";break;

case 23: cout<<"V"<<" ";break;

case 24: cout<<"Cr"<<" ";break;

case 25: cout<<"Mn"<<" ";break;

case 26: cout<<"Fe"<<" ";break;

case 27: cout<<"Co"<<" ";break;

case 28: cout<<"Ni"<<" ";break;

case 29: cout<<"Cu"<<" ";break;

case 30: cout<<"Zn"<<" ";break;

case 31: cout<<"Ga"<<" ";break;

case 32: cout<<"Ge"<<" ";break;

case 33: cout<<"As"<<" ";break;

case 34: cout<<"Se"<<" ";break;

case 35: cout<<"Br"<<" ";break;

case 36: cout<<"Kr"<<" ";break;

default: cout<<"Usl"<<" ";break;

}

}

}

D盗墓笔记之禁婆的考验

推开那扇坑爹的石门，忽然一只冷冰冰的手搭在了Thor的肩上。回头，一张脸惊得他毛骨悚然——居然是——Arthur？！  
长着一张Atrhur的脸的禁婆露出一个妩媚的微笑。  
想过我这一关？先来玩个小游戏吧。很简答，给你一个十进制的数字n，你能求出它在m进制下的表示里数字k的个数么？  
于是加法与求模都不会做的Thor又逗比了，怎么办？看你喽~

# 输入

多组测试数据。  
每组数据为一行，包含三个整数n,m,k（2≤m≤8，0≤k≤9）。保证n在int范围内。

# 输出

对于每组数据，输出一行，包含一个整数，为k的个数。

# 输入样例

9 2 1  
8 8 1

# 输出样例

2  
1

分析：

进制换算我们要不断地进行除法，

例如10换算为二进制数

10/2=5……0;

5/2=2……1;

2/2=1……0;

1/2=0……1;

按照上面的过程要进行求模运算；除法运算；还要判断余数是否等于我们要求的k；直到商为0即可。

这样我们就找到了规律O(∩\_∩)O~

D代码

#include<iostream>

using namespace std;

int main()

{

int n,m,k,a,b,c;

while(cin>>n>>m>>k)

{

c=0;

a=1;

while(a!=0)

{

b=n%m;

a=n/m;

n=a;

if(b==k)

c++;

}

cout<<c<<endl;

}

}

E盗墓笔记之怒海潜沙

# 题目描述

西沙水下，一阵暗流，Thor与天真一行人被冲散了。  
幸运的是，他被冲到了一扇门前。  
不幸的是，刚刚到达这扇门前，门就合上了。上面的机关被重置为初始状态。  
经过仔细的观察，Thor发现，机关为一个圆盘，分为内外两圈。内圈均匀刻着a个小格，外圈均匀刻着b个小格，每个刻度有一个独一无二的图腾与之对应。  
他还发现，每经过一分钟，内外两圈会同时顺时针旋转一个单位。从身边墙壁上的铭文，他得知，当轮盘再次回到初始状态时，门会重新打开。他想知道从门被合上到再次打开的这一个轮回需要多少时间，这个任务就交给你了。

# 输入

多组测试数据。  
对于每组测试数据，输入两个数a,b（1≤a，b≤30000），用空格隔开，具体含义见题目描述。

# 输出

对于每组数据，输出一行，包含一个整数，表示一个轮回需要的时间。

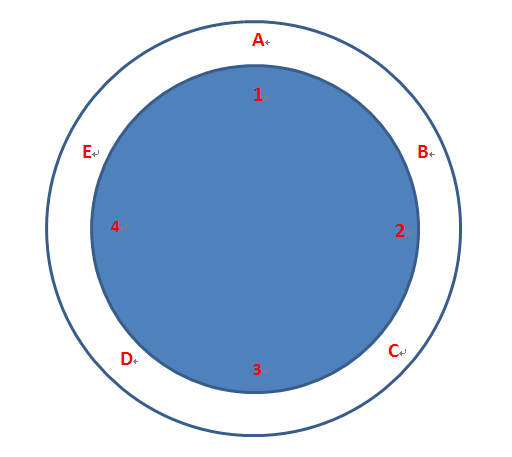
# 输入样例

3 4  
4 8  
12 16

# 输出样例

12  
8  
48

# HINT

机关？大概......就是这个样子吧......OTL  


分析：

大概就是求两个数的最小公倍数吧。O(∩\_∩)O~O(∩\_∩)O~

E代码

（1）辗转相减先求出最大公约数（但是我的方法好像不是很正规）

#include<iostream>

using namespace std;

int main()

{

int a=0,b=0,c=0,d=0;

while(cin>>a>>b)

{

c=a;

d=b;

while(a!=b)

{

if(a-b>0)

a=a-b;

else

b=b-a;

}

cout<<a\*(c/a)\*(d/a)<<endl;

}

}

（2）辗转相除法先求出最大公约数

#include<iostream>

using namespace std;

int main()

{

int a,a1,b,b1,c,d;

while(cin>>a>>b)

{

a1=a;

b1=b;

if(a<b)

{

c=a;

a=b;

b=c;

}

while(a%b!=0)

{

d=b;

b=a%b;

a=d;

}

cout<<b\*(a1/b)\*(b1/b)<<endl;

}

}

F盗墓笔记之逃亡

# 题目描述

电闪雷鸣，警笛大作——不明人士的举报使得乘着列车去往长白雪山的Thor与天真不得不搞了一次夜半逃亡之旅。给你一上帝视角，将他们跑路的路线与警方的追查路线简化为一平面直角坐标系上的两条直线，由于警方人手的优越性，只要Thor与天真有可能出现在对方的路线上，就会被包围逮捕。他们只能找一条完全与警方没有交集的路线，才可以安全逃离。现请给你这两条路线的一般表示（Ax+By+C=0），请你判断他们是否会遭到逮捕。

# 输入

多组测试数据。  
每组测试数据分为两行。第一行包含三个整数A1,B1,C1，用空格隔开，为Thor和天真的逃跑路线。第一行包含三个整数A2,B2,C2，用空格隔开，为警方的追捕路线。  
保证输入数据可以构成直线。

# 输出

对于每组数据，输出一行。如果Thor与天真会被逮捕，输出“Dead End”。如果他们安全逃离，则输出“Safe and Sound”

# 输入样例

1 1 1  
1 1 2  
1 1 1  
1 2 1

# 输出样例

Safe and Sound  
Dead End

分析：

判断两条直线是否相交，若相交就输出Dead End；

不相交就输出Safe and Sound。

a\*b1==a1\*b && a\*c1!=a1\*c O(∩\_∩)O~

F代码

#include<iostream>

using namespace std;

int main()

{

int a,b,c,a1,b1,c1;

while(cin>>a>>b>>c>>a1>>b1>>c1)

{

if((a\*b1==a1\*b)&&(a\*c1!=a1\*c))

cout<<"Safe and Sound"<<endl;

else

cout<<"Dead End"<<endl;

}

}

G李逍遥的仙剑客栈

## Description

有一天天臧散人Arthur到了渝州东南的仙剑客栈，发现李逍遥这小子粗心大意在上酒的时候，有的桌上多上了酒，有的桌上没上酒。唉，谁让Arthur心软呢，不忍心看李逍遥被他婶婶骂，决定帮他——收拾一排桌子。  
给你一个数n表示有多少个桌子，接下来给你一段整数序列表示每个桌上需要的酒量（假设这些桌子在一条直线上，且每个桌子之间的距离都是1），正数表示多放了几瓶酒，负数表示应该放多少瓶酒。请你帮Arthur算一下他提着酒走的最短路程是多少。对了，Arthur体力太渣，一次只能拿一瓶酒。

## Input

第一行一个数T表示有T组数据。  
接下来T组数据，每组数据有2行。  
第一行一个数n(1<=n<=1000)，表示桌子数量，  
接下来第二行有一段数列Ai(|Ai|<=1000)，表示每个桌子上应该放的酒。保证数列总和为0。

## Output

对于每组测试数据，输出一个数，表示Arthur拿着酒走的最少总路程。

## Sample

Input：

2

3

-1 2 -1

6

-1 -1 -1 3 -1 1

Output：

2

7

## Hint

其实就是每瓶酒走的路程之和。

分析：

例如：

桌 1、 2、3、4

酒 3、-1、3、-5

我们要做的就是把每个数变为0

（1）将1下的3全部移动到2下的-1，此时2下变成3-1=2那么路程a=3；

（2）将2下的2全部移动到3下的3，此时3下变成2+3=5 那么路程b=2；

（3）将3下的5全部移动到4下的-5，结束移动

那么路程c=5

（4）全部路程=3+2+5=10；

又例如

桌 1、2、3

酒 -3 、-1、4

1. 将1下-3全部移到2下的-1，此时2下变成-3-1=-4；

那么路程a=3；

1. 将2下-4全部移动到3下的4，结束移动；

那么路程b=4；

1. 总路程=3+4=7；

那么我们就找到了规律O(∩\_∩)O~

G代码

#include<iostream>

#include<cmath>

using namespace std;

int main()

{

int n=0; int c=0; int d=0;

cin>>n;

while(n--)

{

int a=0;

int b=0;

cin>>a;

for(int i=1;i<=a;i++)

{

cin>>c;

c+=d;

b=b+abs(c);

d=c;

}

cout<<b<<endl;

}

}

H jhljx学函数

# Problem Description

jhljx听说大家学了函数，决定考察大家的基本功。  
给你两个数a和b，请用函数来实现交换这两个数，使得a的值为b,b的值为a。  
**本题必须用函数来完成。**  
**不要在函数中先输出b,再输出a。保证a的值是b，b的值是a。请不要水过。**

# Input

输入多组数据。  
每组数据一行，为两个数a和b。(a和b在int范围内)

# Output

输出进行交换后a和b的值。

# Sample Input

1 2

# Sample Output

2 1

# Hint

函数声明的方法：

## 方法1

int fuc(int);  
int main()  
{

}

int fuc(int a)  
{

}

## 方法2

int fuc(int a)  
{

}  
int main()  
{

}  
关于函数的值传递和引用传递

## 值传递

值传递是将数值传递给一个函数，但是函数中得到的数值只是原数值的一个副本。函数中对它进行操作，不会改变main函数中传递进来的那个参数的实际值。  
void fuc(int m,int n)  
{  
m++;n++;  
cout<<m<<" "<<n<<endl;  
}  
int main()  
{  
int a,b;  
cin>>a>>b;  
fuc(a,b);  
cout<<a<<" "<<b<<endl;  
}  
如果输入2 3，输出第一行为3 4，第二行是2 3。第一行的3 4，是在函数中进行运算的结果。而第二行的2 3，是原来a,b的值，说明调用fuc函数进行值传递时，没有改变原有a和b的值。

## 引用传递

引用传递是将数值传递给一个函数，函数中对数值进行操作会改变原来的值。  
**void fuc(int &m,int &n)**  
{  
m++;n++;  
cout<<m<<" "<<n<<endl;  
}  
int main()  
{  
int a,b;  
cin>>a>>b;  
fuc(a,b);  
cout<<a<<" "<<b<<endl;  
}  
如果输入2 3，就会输出两行3 4。这说明在函数中a,b原来的值就已经改变了。

分析：

利用函数进行两个数的交换

H代码

#include<iostream>

using namespace std;

void change(int a,int b)

{

int c;

c=a;

a=b;

b=c;

cout<<a<<" "<<b<<endl;

}

int main()

{

int x,y;

while(cin>>x>>y)

{

change(x,y);

}

}

I\* Ryan's Derivatives

J\*\* RecTangles！RecTangles！RecTangles！

K\*\*\* Talus的会面

。。。。。（待续）。。。。。