Lab Exam 2

<https://vjudge.net/contest/491098#problem/A>

কন্টেস্ট লিংক - <https://vjudge.net/contest/491098>

পাসওয়ার্ড – easyorhard

#### My submission

user name: Engr\_Matin

vjudge profile link: <https://vjudge.net/user/Engr_Matin>

## A - A

**Problem Statement**

There are *N* people numbered *1, 2, …., N*. We have *K* gifts.

Staring with Person *A*, gifts are distributed one by one sequentially. In other words, people receive gifts in the following order: *A, A+1, A+2, ….., N, 1, 2, ……*  Who will receive the last gift?

Formally, after Person *x(1≤x<N)* gets a gift, Person *x+1* will get a gift. After Person *N* gets a gift, Person *1* gets a gift. This goes on until all *K*a gift have been given.

**Constraints**

* *1≤N,K≤1000*
* *1≤A≤N*
* All values in input are integers.

**Input**

Input is given from Standard Input in the following format:

*N* *K* *A*

**Output**

Print a number of the person who will get the last card.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 3 3 2 | 1 |

The cards are given to Person *2, 3, 1* in this order.

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 1 100 1 | 1 |

**Sample 3**

| **Input** | **Output** |
| --- | --- |
| 3 14 2 | 3 |

#include<stdio.h>

int main()

{

int n, k, a, t;

scanf("%d %d %d", &n, &k, &a);

if(k%n==0)

{

k=n;

}

else

{

k=k%n;

}

if((k+(a-1))<=n)

{

t=k+(a-1);

}

else

{

t=k+(a-1)-n;

}

printf("%d", t);

return 0;

}

B – B

Abu and Babu are close friends. Recently they both got admitted into the same university. Now they want to live in the university halls.

They want to live in the same room. There are *n* rooms in the hall. At the moment the *i*-th room has *pi* people living in it and the room can accommodate *qi* people in total (*pi* ≤ *qi*). Your task is to count how many rooms has free place for both Abu and Babu.

**Input**

The first line contains a single integer *n* (1 ≤ *n* ≤ 100) — the number of rooms.

The *i*-th of the next *n* lines contains two integers *pi* and *qi* (0 ≤ *pi* ≤ *qi* ≤ 100) — the number of people who already live in the *i*-th room and the room's capacity.

**Output**

Print a single integer — the number of rooms where both friends can move in.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 3  1 1  2 2  3 3 | 0 |

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 3  1 10  0 10  10 10 | 2 |

#include<stdio.h>

int main()

{

int i, n, p[100], q[100];

scanf("%d", &n);

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

{

scanf("%d %d", &p[i], &q[i]);

}

int count=0;

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

{

if((q[i]-p[i])>=2)

{

count++;

}

}

printf("%d", count);

return 0;

}

C – C

**Problem Statement**

Masud has recently got his dream job at Voogle. To get to his job, he leaves home at *S* o'clock (on the *24*-hour clock) every day and returns home *T* o'clock every day.  
The date may change while he is out of home. Outside this time period, he always stays at home.

Determine whether Masud is away from home at *30* minutes past *X* o'clock.

**Constraints**

* *0≤S,T,X≤23*
* *S!=T*
* All values in input are integers.

**Input**

Input is given from Standard Input in the following format:

*S* *T* *X*

**Output**

If Masud is not at home at *30* minutes past *X*  o'clock, print Yes; otherwise, print No.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 7 20 12 | Yes |

Masud is away from home between *7* o'clock and *20* o'clock. At *30* minutes past *12* o'clock, he is not at home, so we print Yes.

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 20 7 12 | No |

Masud is away from home between *20* o'clock and *0* o'clock and between *0* o'clock and *7* o'clock (on the next day). At *30* minutes past *12* o'clock, he is home, so we print No.

**Sample 3**

| **Input** | **Output** |
| --- | --- |
| 23 0 23 | Yes |

#include<stdio.h>

int main()

{

int s, t, x;

scanf("%d %d %d", &s, &t, &x);

if(t>s)

{

if((x\*60+30)>s\*60 && (x\*60+30)<t\*60)

{

printf("Yes");

}

else

{

printf("No");

}

}

else if(t<s)

{

if((x\*60+30)>s\*60 && (x\*60+30)<(t+24)\*60)

{

printf("Yes");

}

else if((x\*60+30)>s\*60 || (x\*60+30)<t\*60)

{

printf("Yes");

}

else

{

printf("No");

}

}

return 0;

}

D – D

Bjorn Ironside likes to play chess, and so does his enemy King Harald.

Once they have played *n* games in a row. For each game it's known who was the winner — Bjorn Ironside or King Harald. None of the games ended with a tie.

Now Bjorn Ironside wonders, who won more games, he or King Harald? Help him determine this.

**Input**

The first line of the input contains a single integer *n* (1 ≤ *n* ≤ 100 000) — the number of games played.

The second line contains a string *s*, consisting of *n* uppercase English letters 'A' and 'D' — the outcome of each of the games. The *i*-th character of the string is equal to 'A' if the Bjorn Ironside won the *i*-th game and 'D' if King Harald won the *i*-th game.

**Output**

If Bjorn Ironside won more games than King Harald, print "Anton" (without quotes) in the only line of the output.

If King Harald won more games than Bjorn Ironside, print "Danik" (without quotes) in the only line of the output.

If Bjorn Ironside and King Harald won the same number of games, print "Friendship" (without quotes).

**Examples**

**Input**

6  
ADAAAA

**Output**

Anton

**Input**

7  
DDDAADA

**Output**

Danik

**Input**

6  
DADADA

**Output**

Friendship

**Note**

In the first sample, Bjorn Ironside won 6 games, while King Harald — only 1. Hence, the answer is "Anton".

In the second sample, Bjorn Ironside won 3 games and King Harald won 4 games, so the answer is "Danik".

In the third sample, both Bjorn Ironside and King Harald won 3 games and the answer is "Friendship".

#include<stdio.h>

int main()

{

long long i, n, c1=0, c2=0;

scanf("%lld", &n);

char s[100000];

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

{

scanf(" %c", &s[i]);

}

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

{

if(s[i]=='A')

{

c1++;

}

else if(s[i]=='D')

{

c2++;

}

}

if(c1>c2)

{

printf("Anton");

}

else if(c2>c1)

{

printf("Danik");

}

else if(c2==c1)

{

printf("Friendship");

}

return 0;

}

E – E

**Problem Statement**

Takahashi and Aoki decided to train themselves by running.  
Takahashi repeats the following schedule: "Run for *A* seconds at *B* meters per second and then rest for *C* seconds."  
Aoki repeats the following schedule: "Run for *D* seconds at *E* meters per second and then rest for *F* seconds."  
When *X* seconds have passed since they simultaneously started running, which of Takahashi and Aoki is ahead?

**Constraints**

* *1≤A,B,C,D,E,F,X≤100*
* All values in input are integers.

**Input**

Input is given from Standard Input in the following format:

*A* *B* *C* *D* *E* *F* *X*

**Output**

When *X* seconds have passed since they simultaneously started, if Takahashi goes ahead of Aoki, print Takahashi; if Aoki goes ahead of Takahashi, print Aoki; if they have advanced the same distance, print Draw.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 4 3 3 6 2 5 10 | Takahashi |

During the first *1010* seconds after they started to jog, they move as follows.

* Takahashi walks for *4* seconds, takes a rest for *3* seconds, and walks again for *3* seconds. As a result, he advances a total of *(4+3)×3=21* meters.
* Aoki walks for *6* seconds and takes a rest for *4* seconds. As a result, he advances a total of *6×2=12* meters.

Since Takahashi goes ahead, Takahashi should be printed.

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 3 1 4 1 5 9 2 | Aoki |

**Sample 3**

| **Input** | **Output** |
| --- | --- |
| 1 1 1 1 1 1 1 | Draw |

#include<stdio.h>

int main()

{

int a, b, c, d, e, f, x, s1, s2;

scanf("%d %d %d %d %d %d %d", &a, &b, &c, &d, &e, &f, &x);

if(x%(a+c)>=a)

{

s1=((x/(a+c))\*a+a)\*b;

}

else if(x%(a+c)<a)

{

s1=((x/(a+c))\*a+x%(a+c))\*b;

}

if(x%(d+f)>=d)

{

s2=((x/(d+f))\*d+d)\*e;

}

else if(x%(d+f)<d)

{

s2=((x/(d+f))\*d+x%(d+f))\*e;

}

if(s1>s2)

{

printf("Takahashi");

}

else if(s2>s1)

{

printf("Aoki");

}

else if(s2==s1)

{

printf("Draw");

}

return 0;

}

## F - F

Matilda is practising driving. Since, she is not a very good driver, she drives in a weird pattern.

She drives on a *n* by *m* board. Let (*x*, *y*) be the *y*-th cell from left in the *x*-th row . Matilda starts at (1, 1). Then she drives rightwards to (1, *m*). Then she goes down two cells to (3, *m*) . After than she moves left to (3, 1) and so on.

Your task is to draw her path. If she visits a cell, it should be represented as hash characters ('#') and the unvisited cells should be filled with dots ('.').

Refer to the sample tests in order to understand the pattern.

**Input**

The only line contains two integers: *n* and *m* (3 ≤ *n*, *m* ≤ 50).

*n* is an **odd** number.

**Output**

Output *n* lines. Each line should contain a string consisting of *m* characters. Do not output spaces.

**Sample 1**

| **Inputcopy** | **Outputcopy** |
| --- | --- |
| 3 3 | ###  ..#  ### |

**Sample 2**

| **Inputcopy** | **Outputcopy** |
| --- | --- |
| 3 4 | ####  ...#  #### |

**Sample 3**

| **Inputcopy** | **Outputcopy** |
| --- | --- |
| 5 3 | ###  ..#  ###  #..  ### |

**Sample 4**

| **Inputcopy** | **Outputcopy** |
| --- | --- |
| 9 9 | #########  ........#  #########  #........  #########  ........#  #########  #........  ######### |

#include<stdio.h>

int main()

{

int n, m, i, j;

scanf("%d %d", &n, &m);

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

{

for(j=1; j<=m; j++)

{

if(i%2!=0)

{

printf("#");

}

else if(i%2==0 && i%4!=0 && j!=m)

{

printf(".");

}

else if(i%2==0 && i%4!=0 && j==m)

{

printf("#");

}

else if(i%4==0 && j==1)

{

printf("#");

}

else if(i%4==0 && j<=m)

{

printf(".");

}

}

printf("\n");

}

return 0;

}

G – G

**Problem Statement**

How many integers not less than *A* and not more than *B* are there?

**Constraints**

* *1≤A≤100*
* *1≤B≤100*
* *A* and *B* are integers.

**Input**

Input is given from Standard Input in the following format:

*A* *B*

**Output**

Print the number of integers not less than *A* and not more than *B*.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 2 4 | 3 |

We have three integers not less than *2* and not more than *4*: *2*, *3*, *4*, so we should print *3*.

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 10 100 | 91 |

**Sample 3**

| **Input** | **Output** |
| --- | --- |
| 3 2 | 0 |

We have no integers not less than *3* and not more than *2*, so we should print *0*.

#include<stdio.h>

int main()

{

int a, b, i, n=0;

scanf("%d %d", &a, &b);

for(i=a; i<=b; i++)

{

n++;

}

printf("%d", n);

return 0;

}

H – H

Ragnar Lothbrok is a restless warrior and family man who longs to find and explore new countries across the sea , his most important wish is to find lands which are Good for planting . That's why he asked you to help him to express his feelings about lands.

Ragnar likes the Inception so much, and like that his feelings are complicated. They have *n* layers. The first layer is hate, second one is love, third one is hate and so on...

For example if *n* = 1, then his feeling is "I hate it" or if *n* = 2 it's "I hate that I love it", and if *n* = 3 it's "I hate that I love that I hate it" and so on.

Please help Ragnar.

**Input**

The only line of the input contains a single integer *n* (1 ≤ *n* ≤ 100) — the number of layers of love and hate.

**Output**

Print Ragnar Lothbrok's feeling in one line.

**Examples**

**Input**

1

**Output**

I hate it

**Input**

2

**Output**

I hate that I love it

**Input**

3

**Output**

I hate that I love that I hate it

#include<stdio.h>

int main()

{

int i, n;

scanf("%d", &n);

char a[1200]= "I hate ";

char b[15]="that I love ";

char c[15]="that I hate ";

char d[5]="it";

for(i=2; i<=n; i=i+1)

{

if(i%2==0)

{

strcat(a,b);

}

else if(i%2!=0)

{

strcat(a,c);

}

}

strcat(a,d);

puts(a);

return 0;

}

I – I

**Problem Statement**

Samantha takes an coding exam. She will *100* and get a **grade** according to the marks she gets in the exam, as follows:

* Newbie, if her marks are not less than *0* but less than *40*;
* Pupil, if her marks are not less than *40* but less than *70*;
* Specialist, if her marks are not less than *70* but less than *90*;
* Expert, if her marks are not less than *90*.

Samantha took this test and got *X* points.

Find the minimum number of extra points needed to go up one grade higher. If, however, her grade was Expert, print **expert** as there is no higher grade than that.

**Constraints**

* *0≤X≤100*
* *X* is an integer.

**Input**

Input is given from Standard Input in the following format:

*X*

**Output**

Print the answer.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 56 | 14 |

She got *56* points and was certified as Pupil. In order to reach the next rank of Specialist, she needs at least *14* more points.

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 32 | 8 |

**Sample 3**

| **Input** | **Output** |
| --- | --- |
| 0 | 40 |

**Sample 4**

| **Input** | **Output** |
| --- | --- |
| 100 | expert |

She got full points and was certified as Expert. There is no rank higher than that, so print expert.

#include<stdio.h>

int main()

{

int x, p;

scanf("%d", &x);

if(x>=0 && x<40)

{

p=40-x;

printf("%d", p);

}

else if(x>=40 && x<70)

{

p=70-x;

printf("%d", p);

}

else if(x>=70 && x<90)

{

p=90-x;

printf("%d", p);

}

else if(x>=90 && x<=100)

{

printf("expert");

}

return 0;

}

J – Digits

**roblem Statement**

Let *xyz* denote the *33*-digit integer whose digits are *x*, *y*, *z* from left to right.

Given a *3*-digit integer *abc* none of whose digits is *0*, find  *abc+bca+cab*.

**Constraints**

* *abc* is a *3*-digit integer *abc* none of whose digits is *0*.

**Input**

Input is given from Standard Input in the following format:

*abc*

**Output**

Print the answer.

**Sample 1**

| **Input** | **Output** |
| --- | --- |
| 123 | 666 |

We have *123+231+312=666*.

**Sample 2**

| **Input** | **Output** |
| --- | --- |
| 999 | 2997 |

We have *999+999+999=2997*.

#include<stdio.h>

int main()

{

int n, a, b, c, t;

scanf("%3d", &n);

c=n%10;

b=(n/10)%10;

a=(n/100)%10;

t=100\*a+10\*b+c+100\*b+10\*c+a+100\*c+10\*a+b;

printf("%d", t);

return 0;

}