PREVIOUS TCS CODEVITA4 PROBLEMS

Problem 1 -- Logic Pyramid

Identify the logic behind the series

6 28 66 120 190 276....

The numbers in the series should be used to create a Pyramid. The base of the Pyramid will be the widest and will start converging towards the top where there will only be one element. Each successive layer will have one number less than that on the layer below it. The width of the Pyramid is specified by an input parameter N, In other words, there will be N numbers on the bottom layer of the pyramid.

The Pyramid construction rules are as follows
1. The first number in the series should be at the top of the Pyramid
2. Last N number of the series should be on the bottom-most layer of the Pyramid, with N^{th} number being the right-most number of this layer.
3. Numbers less than 5-digits must be padded with zeroes to maintain the sanctity of a Pyramid when printed. Have a look at the examples below to get a pictorial understanding of what this rule actually means.

Example

If input is 2, output will be

```
 00006
00028 00066
```

If input is 3, output will be

```
  00006
 00028 00066
00120 00190 00276
```

Formal input and output specifications are stated below

Input Format:
The first line of input will contain number N that corresponds to the width of the bottom-most layer of the Pyramid

Output Format:
The Pyramid constructed out of numbers in the series as per stated construction rules

Constraints:
0 < N <= 14 (If input is not in range print "Invalid Input")

Sample Input 1:
2
Sample Output 1:

00006
00028 00066

Sample Input 2:
3
Sample Output 2:
00006
00028 00066
00120 00190 00276

Sample Input 3:
5
Sample Output 3:
00006
00028 00066
00120 00190 00276
00378 00496 00630 00780
00946 01128 01326 01540 01770

Test Cases

SNo	Input	Output
1	3	00006 00028 00066 00120 00190 00276
2	15	Invalid Input
3	6	00006 00028 00066 00120 00190 00276 00378 00496 00630 00780 00946 01128 01326 01540 01770 02016 02278 02556 02850 03160 0348 6
4	1	00006
5	5	00006 00028 00066 00120 00190 00276 00378 00496 00630 00780 00946 01128 01326 01540 01770
6	0	Invalid Input
7	4	00006 00028 00066 00120 00190 00276 00378 00496 00630 00780

SNo	Input	Output
8	2	00006 00028 00066

```
#include<stdio.h>
int main()
{
   long long n,i,c,d,e[1000],l=0,k,t,m=0,b;
   scanf("%lld",&n);
   t=(n*(n+1))/2;
   c=0;d=6;
   for(i=0;i<t;i++)
   {
       c=c+(16*i)+d;
       e[l]=c;
       l++;
   }
   if(n<=0 || n>=14)
   printf("Invalid Input");
   else
   {
       for(i=0;i<n;i++)
       {
           k=i+1;
           for(b=i;b<n-1;b++)
           printf(" ");
           while(k>0)
           {
               printf("%05lld ",e[m]);
               m++;
               k--;
           }
           printf("\n");
       }
       }
   return 0;
}
```

Problem 2 -- The Great Chase

Welcome to the Planet Zandar, the second most prominent planet in the Milky Way Galaxy (of course after our own Earth).

The planet is in a distress condition, a Group of Galactic pirates, Zorons have stolen the Trident Crystal, which is the main source of energy of the planet, and are escaping the Galaxy. The Nova Corps, the military agency of Zandar, have gathered intelligence that the Zoronion spacecraft can run in cosmic leaps of exactly D units, (it means that the

spacecraft will move D units from its position in every leap/turn) and is currently I units away from Zandar.

The Zandarian Space crafts can run in cosmic leaps of exactly Z units. The Commander of Nova Corps wants to know the smallest number of leaps required to catch Zorons (Note that it is possible to catch the pirates only when they will be at the same point in the cosmic universe). The Zorons, even though are clever thieves, travel in one direction, and keep jumping exactly D units without stopping at any point. The Nova Corps can dial in the number of jumps they need to make (each of them exactly Z units), and reach the place almost instantly. They can then wait there until the Zorons arrive, and recover the Trident Crystal.

However, their wizard has told them that there may be situations where it is impossible for the Nova corps to be at the same distance as the Zorons.

As the planet is out of power currently, their supercomputers are shut down and they are not able to calculate the required information. As you are there from Earth they have approached you for help.

Note: Assume that the Cosmic universe is one dimensional.

Input
An integer T for the number of test cases, followed by T test cases each one consisting of three numbers
1) I :- initial position of Zorons
2) D:- distance covered in a single cosmic leap by Zoronion space craft.
3) Z:- distance covered by Zandarian space crafts.

Output
A single number, the number of leaps required to catch the pirates, and if it is not possible to catch them, the output will be -1
Constraints $1 \leq I,D \leq 1012$ $1 \leq Z \leq 109$

Example 1
Input:
2
9 5 12
5 7 9
Output:
2
6
Explanation: The first line is 2, so T is 2, and there are 2 test cases.
In the first test case, The Zorons will initially be at 9 and then they will leap to 14,19 24..... The Nova Corps can take leaps of 12 and will catch them nearest at a distance 24, taking 2 leaps 12 and 24.
In the second test case, The Zorons will initially be at 5 and then they will leap to 12,19

26, 33, 40, 47, 54..... The Nova Corps can take leaps of 9 and will catch them nearest at 54, taking 6 leaps.
Hence the output has 2 lines, 2 and 6.

Example 2
Input:
1
10 15 20
Output:
2
Explanation: The first line is 1, so T is 1, and there is 1 test case.
The Zorons will initially be at 10, and jump in jumps of 15, landing at 25,40
The Nova Corps take leaps of 20, and arrive at 20, 40. Hence, they can meet at 40 after 2 leaps. The output is 2.

SNo	Name	Input	Output
1	TC3	1 20 10 15	2
2	TC2	2 9 5 12 5 7 9	2 6
3	TC6	4 20 30 40 10 20 30 21 54 65 89 94 99	2 1 51 93
4	TC4	2 45 56 47 89 90 91	-1 -1
5	TC1	1 10 15 20	2
6	TC5	3 65 87 95 5 45 56 12 15 49	19 25 -1

```
n=int(input())
for j in range(n):
    arr=[int(i) for i in input().split()]
    s=1
    t=arr[0]
    c=0
```

```
while s!=0:

    t=t+arr[1]
    for k in range(2,arr[2]+1):
       if t%k==0 and arr[2]%k==0:
           c+=1

    if c==0:
        print("-1")
        break
    s=t%arr[2]
    if s==0:
        print(int(t/arr[2]))
```

Problem 3 - Random Walk

You are playing a game with closed eyes. You take alternate forward and backward steps. There are banana peels on front and back.
Given the initial direction, the magnitude of forward movement, the magnitude of backward movement, the time taken to move one meter, the position of banana peel in front and position of banana peel in back.
Your task is to find out if you will slip over the banana skin on the forward path or banana skin on the backward path and in how much time. It is also possible that by your good fortune you might not step over either banana skins. Write a program to calculate the outcome.

Input Format:

The first line consists of a single character (F or B) that denotes the initial direction.
The second line consists of a single integer that denotes the magnitude of forward movement in meters.
The third line consists of a single integer that denotes the magnitude of backward movement in meters.
The fourth line consists of a single integer that denotes time taken to cover 1 meter.
The fifth line consists of a single integer that denotes the distance from your starting position and the banana skin in the forward direction.
The sixth line consists of a single integer that denotes the distance from your starting position and the banana skin in the backward direction.

Output Format:

Print "Hurray" (without the quotes) if you don't slip on any banana peel or else output contains 2 lines.
In the first line print, the time taken to slip over the forward or backward banana skin.
In the second line print F if he slips over forward banana skin and B if he slips over the backward banana skin.

Sample Input 1:

B
14

12
7
25
4

Sample Output 1:
28
B

Sample Input 2:
B
8
8
5
9
18

Sample Output 2:
Hurray

SNo	Name	Input	Output
1	T5	F 1 12 3 22 28	102 B
2	ST2	B 8 8 5 9 18	Hurray
3	T6	F 8 8 5 7 9	35 F
4	T4	F 2 3	675 B

SNo	Name	Input	Output
		9 12 15	
5	T8	F 13 8 3 43 46	417 F
6	T3	B 11 4 6 10 17	156 F
7	ST1	B 14 12 7 25 4	28 B
8	T7	B 14 12 6 2 3	18 B

Problem 4 -- Date Time

Arun and his sister Usha are challenging each other with some mathematical puzzles. Usha, the cleverer one, has come up with the idea of giving Arun 12 distinct digits from 0 to 9, and have him form the largest date time in 2018 with them. Arun is a little nervous and asks you to help him with a computer program.

Usha will give Arun 12 distinct digits. He needs to create a date-time combination in the year 2018, the date in the MM/DD form (all four digits must be present), and the time in the format HH: MM (all four digits must be present). The date may be from 01/01 to 12/31 and the time may be from 00:00 to 23:59 (in the 24-hour format). The digits provided may be used only once in the answer that Arun gives.

If more than one date time combination may be formed, Arun needs to give the latest valid date time possible in the year 2018.

Constraints

Single digits (any of 0-9)

Input Format

A line consisting of a sequence of 12 (not necessarily distinct) single digits (any of 0-9) separated by commas. The sequence will be non-decreasing.

Output

The maximum possible valid date time in the year 2018. The output must be in the format

MM/DD HH:MM

If no date time can be constructed, the output should be 0

Explanation

Example1 :

Input

0,0,1,2,2,2,3,5,9,9,9,9

Output

12/30 22:59

Explanation

The 12 digits to be used by Arun are given.

The maximum valid date time using only the digits given, and with each digit used at most once is

12/30 22:59

This is the output.

Example 2

Input

3,3,3,3,3,3,3,3,3,3,3,3

Output

Explanation

As no digit less than 3 is present in the input, a valid month cannot be formed. Hence no valid Date time can be formed with the input digits.

Test Cases

SNo	Input	Output
1	1,2,3,4,5,6,7,8,9,8,1,2	12/29 18:58
2	3,3,3,3,3,3,3,3,3,3,3,3	0
3	7,8,9,4,5,6,1,2,3,7,8,9	0
4	0,0,1,2,2,2,3,5,9,9,9,9	12/30 22:59
5	4,5,2,8,7,2,3,7,1,2,6,8	12/28 23:58

Problem 5 --- Rook Moves

A Chessboard position is accurately captured by Forsyth-Edwards notation and is abbreviated as FEN. A FEN "record" defines a particular game position, all in one text line and using only the ASCII character set. A FEN record contains six fields. A complete description of the FEN format to represent Chess positions can be found here

For the purpose of this problem only consider first of the six fields of FEN. Before we describe the problem, let us look at how FEN maps to a board position. The following 5 images show board positions and its corresponding FEN representation.

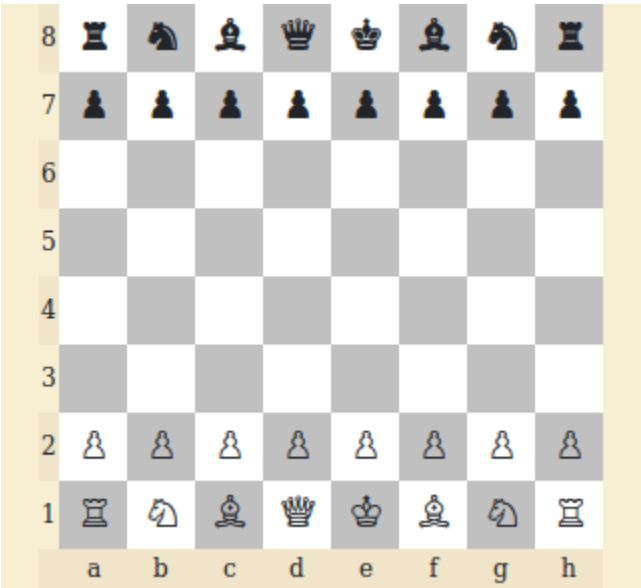

Figure 1.

This board position depicts initial position before any side has made a move. In FEN format this board position is represented as

rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w

Let's say, White plays e4. Then the board position looks like shown below

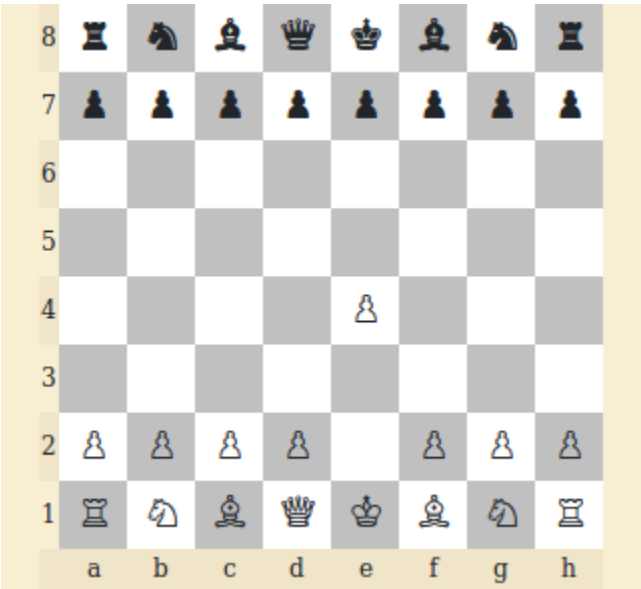

Figure 2.

This board position depicts the Chessboard after White has played e4. In FEN format this board position is represented as

rnbqkbnr/pppppppp/8/8/4P3/8/PPPP1PPP/RNBQKBNR b

Similarly, 3 more half-moves are depicted in the following diagrams

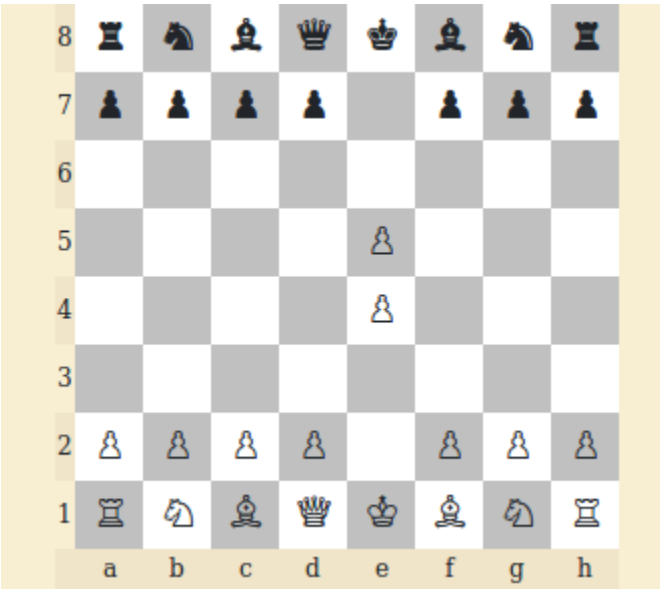

Figure 3.

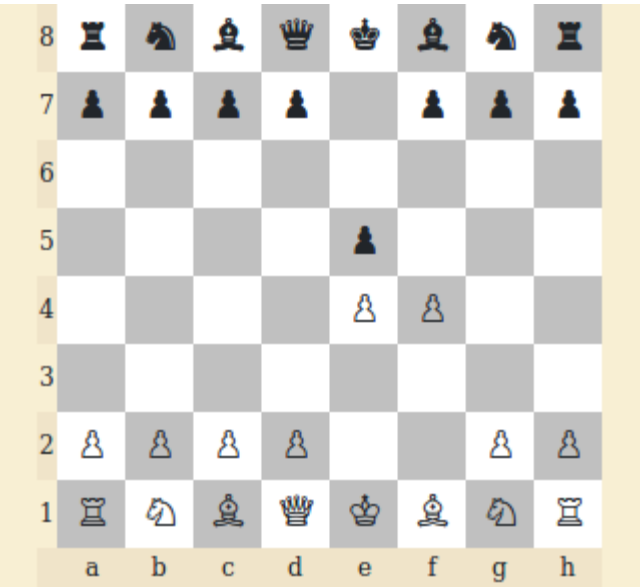

Figure 4.

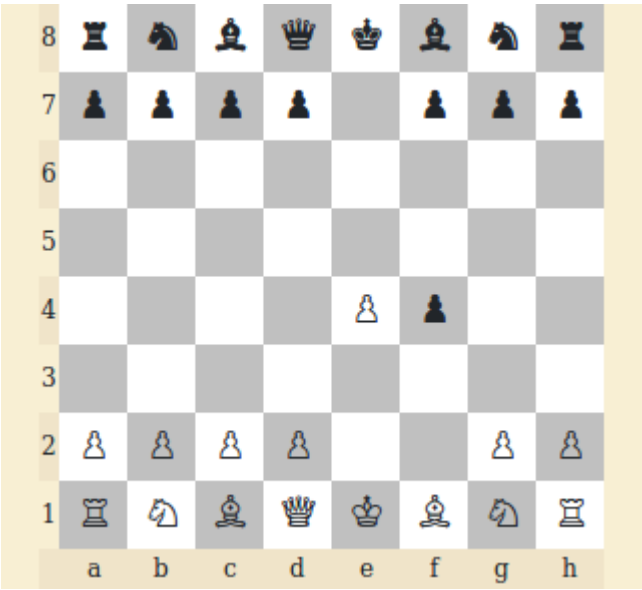

Figure 5.

The FENs corresponding to Figure 3, 4 and 5 are represented as

3. rnbqkbnr/pppp1ppp/8/4P3/4P3/8/PPPP1PPP/RNBQKBNR w
4. rnbqkbnr/pppp1ppp/8/4p3/4PP2/8/PPPP2PP/RNBQKBNR b
5. rnbqkbnr/pppp1ppp/8/8/4Pp2/8/PPPP2PP/RNBQKBNR w

Wikipedia describes the first field of FEN format as follows

Each rank is described, starting with rank 8 and ending with rank 1; within each rank, the contents of each square are described from file "a" through file "h". Following the Standard Algebraic Notation (SAN), each piece is identified by a single letter taken from the standard English names (pawn = "P", knight = "N", bishop = "B", rook = "R", queen = "Q" and king = "K").[1] White pieces are designated using upper-case letters ("PNBRQK") while black pieces use lowercase ("pnbrqk"). Empty squares are noted using digits 1 through 8 (the number of empty squares), and "/" separates ranks.

The second field denotes whose move it is now. "w" depicts that it is White's turn to play and "b" indicates that it is Black's turn to play

CodeVita Problem Statement

Given a board position in FEN format, your task is to find out all the move(s) that Rook(s) of the playing side can make.

Input Format:

1. The first line contains a single FEN record, which corresponds to a particular board position and also indicates whose turn it is.

Output Format:

1. The output must be printed as follows
    1. All legal moves that Rook(s) can make must be in the format "[]"
    2. Where is move represented in the format "[fromSquare][toSquare]"
2. See Example section for better understanding of output format
3. Follow Output printing specification to print the output in the required format

Constraints:

1. Since we focus on only the first two parts of the FEN, we are essentially ignoring the possibility of Castling. **Hence our test cases don't contain FENs which give rise to such positions.**

Sample Input1:

2k5/8/8/3B4/2PRN3/3B4/8/4K3 w

Sample Output1:

[]

Sample Input2:

2k5/2p5/8/8/2PRN3/8/8/2K5 w

Sample Output2:

[d4d8, d4d7, d4d6, d4d5, d4d3, d4d2, d4d1]

Print Specification:

1. Should start with "[" and end with "]"
2. If more than one move is possible, moves should be separated by a comma followed by whitespace
3. Moves of a single rook should be printed in *Move Format*. Scan the board from 8th rank to 1st rank from a-file to h-file. Whichever square gets hit first, that move should be printed first.
4. If more than one rook exists for side to move, then start scanning for rook from 8th rank to 1st rank, left to right i.e. from a-file to h-file. Whichever rook appears first, print all moves for that rook first.
5. Verify your understanding of how printing should happen against examples shown above

SNo	Input	Output
1	2r5/8/8/2R5/8/8/8/8 b	[c8c7,c8c6,c8c5,c8h8,c8g8,c8f8,c8e8,c8d8,c8b8,c8a8]
2	2k5/2p5/8/8/2PRN3/8/8/2K5 w	[d4d8,d4d7,d4d6,d4d5,d4d3,d4d2,d4d1]
3	2k5/2p5/8/8/2PrN3/8/8/2K5 b	[d4d8,d4d7,d4d6,d4d5,d4d3,d4d2,d4d1,d4e4,d4c4]
4	R1k5/2p5/8/8/2PN4/8/8/2K5 w	[a8a7,a8a6,a8a5,a8a4,a8a3,a8a2,a8a1,a8c8,a8b8]
5	2kp4/2p5/8/8/2PrN3/8/8/2K5 b	[d4d7,d4d6,d4d5,d4d3,d4d2,d4d1,d4e4,d4c4]
6	2k5/8/8/3B4/2PRN3/3B4/8/4K3 w	[]
7	2k5/2p5/8/R7/2PN4/8/8/2K5 w	[a5a8,a5a7,a5a6,a5a4,a5a3,a5a2,a5a1,a5h5,a5g5,a5f5,a5e5,a5d5,a5c5,a5b5]

PROBLEM 6 : Exam Efficiency

In an examination with multiple choice questions, the following is the exam question pattern.

- X1 number of One mark questions, having negative score of -1 for answering wrong
- X2 number of Two mark questions, having negative score of -1 and -2 for one or both options wrong
- X3 number of Three mark questions, having negative score of -1, -2 and -3 for one, two or all three options wrong
- Score Required to Pass the exam : Y
- For 1,2 and 3 mark questions, 1,2 and 3 options must be selected. Simply put, once has to attempt to answer all questions against all options.

Identify the minimum accuracy rate required for **each type of question** to crack the exam.

Calculations must be done up to 11 precision and printing up to 2 digit precision with ceil value

Input Format:

First line contains number of one mark questions denoted by X1,
Second line contains number of two mark questions denoted by X2
Third line contains number of three mark questions denoted by X3
Fourth line contains number of marks required to pass the exam denoted by Y.

Output Format:

Minimum Accuracy rate required for one mark question is 80%
Minimum Accuracy rate required for two mark question is 83.33%
Minimum Accuracy rate required for three mark question is 90%

Note: - If the mark required to pass the exam can be achieved by attempting without attempting any particular type of question then show message similar to, One mark question need not be attempted, so no minimum accuracy rate applicable

See Example Test cases for better understanding.

Sample Input 1:

20
30
30
120

Sample Output 1:

One mark questions need not be attempted, so no minimum accuracy rate applicable.
Minimum Accuracy rate required for two mark question is 58.33%
Minimum Accuracy rate required for three mark question is 72.22%

Sample Input 2:

20
30
30
170

Sample Output 2:

Minimum Accuracy rate required for one mark question is 100.0%
Minimum Accuracy rate required for two mark question is 100.0%
Minimum Accuracy rate required for three mark question is 100.0%

Test Cases

SNo	Input	Output
1	20 30 30 120	One mark questions need not be attempted, so no minimum accuracy rate applicable. Minimum Accuracy rate required for two mark question is 58.33% Minimum Accuracy rate required for three mark question is 72.22%
2	100 100 100 550	Minimum Accuracy rate required for one mark question is 75.0% Minimum Accuracy rate required for two mark question is 87.5% Minimum Accuracy rate required for three mark question is 91.67%
3	120 140 170 750	One mark questions need not be attempted, so no minimum accuracy rate applicable. Minimum Accuracy rate required for two mark question is 71.43% Minimum Accuracy rate required for three mark question is 84.31%
4	30 70 70 300	One mark questions need not be attempted, so no minimum accuracy rate applicable. Minimum Accuracy rate required for two mark question is 71.43% Minimum Accuracy rate required for three mark question is 80.95%
5	20 30 30 170	Minimum Accuracy rate required for one mark question is 100.0% Minimum Accuracy rate required for two mark question is 100.0% Minimum Accuracy rate required for three mark question is 100.0%
6	20 30 30 160	Minimum Accuracy rate required for one mark question is 75.0% Minimum Accuracy rate required for two mark question is 91.67% Minimum Accuracy rate required for three mark question is 94.44%