SAPO 2018 Round 2 Questions FINAL

Question 1: Perfect 2

A **proper divisor** of an integer n is a positive integer strictly less than n which evenly divides n. For example, the proper divisors of 6 are 1, 2 and 3.

Let $D(n)$ denote the sum of the proper divisors of an integer n. So $D(6) = 1 + 2 + 3 = 6$, $D(4) = 1 + 2 = 3$, and $D(7) = 1$.

If $D(n)$ is less than n, we call n **deficient**. If $D(n)$ is equal to n, we call n **perfect**. If $D(n)$ is greater than n, we call n **abundant**.

Write a program that will read in two positive integers, A and B, and then display the number of **deficient**, **perfect** and **abundant** numbers (in this order) between A and B, inclusive of both A and B. The output values must be separated by a single space.

Examples:

Input:	4 6	Output:	2 1 0
Input:	11 12	Output:	1 0 1
Input:	25 30	Output:	4 1 1

In the first example, A = 4 and B = 6. So, we consider the numbers 4, 5 and 6. Both 4 and 5 are deficient, while 6 is perfect. Hence, we have 2 deficient numbers, 1 perfect number, and 0 abundant numbers between 4 and 6. The output, therefore, is 2 1 0.

Test your program with the following cases:

a) `100 110`
b) `20 80`
c) `500 1000`
d) `1000 10000`

Question 2: Pattern

Starting with 2 and using a common difference of 3, we can get the sequence 2, 5, 8, 11, 14, …

Starting with 2 and using common differences of 3 and -2, we can get the sequence 2, 5, 3, 6, 4, 7, … (we add 3, subtract 2, add 3 again, subtract 2, and so on).

Given the first few terms of a sequence, the program you must write should determine the smallest group of common differences, in order, that will generate the sequence if you are given the first term.

The first number in the input is the number of terms, n. This is followed by n numbers, the terms of the sequence, separated by spaces. The program should output the group of integers you have found, separated by spaces.

Examples:

Input: 5
2 5 8 11 14

Output: 3

Input: 6
2 5 3 6 4 7

Output: 3 -2

Input: 12
3 1 2 5 3 4 7 5 6 9 7 8

Output: -2 1 3

In the second example, `3 -2 3 -2` would also generate the sequence, but is not the smallest group, of common differences and thus would be incorrect. `-2 3` would produce `2 0 3 1 4 2` and thus would also not be correct.

Test your program with the following cases:

a) 8
1 5 9 13 17 21 25 29

b) 12
5 2 9 6 13 10 17 14 21 18 25 22

c) 11
10 13 15 18 14 19 22 24 27 23 28

d) 25
4 10 7 5 11 8 6 12 9 7 13 10 9 15 12 10 16 13 11 17 14 12 18 15 14

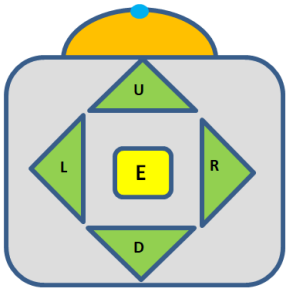
Question 3: Game Console

You are playing on a game console and must enter a phrase using the game controller. On the screen, you are given a grid of keys, as shown below, with the cursor initially hovering over the **A**-key.

Console

A	**B**	**C**	**D**	**E**	**F**
G	**H**	**I**	**J**	**K**	**L**
M	**N**	**O**	**P**	**Q**	**R**
S	**T**	**U**	**V**	**W**	**X**
Y	**Z**	**Space**	**-**	**.**	**Accept**

The controller contains an 'Enter' button (E), and 4 arrow buttons in the up (U), down (D), left (L) and right (R) directions. The 'Enter' button will enter the key which the cursor currently indicates. The arrow buttons move the cursor one position in the specified direction (so if the cursor is at **C**, with one press of the right button the cursor will then move to **D**).



Controller

Given a single uppercase phrase, the program you must write must determine the minimum number of button presses on the controller that it will take to enter in the given phrase on the game console. Once the phrase has been entered the **Accept** key must then be pressed to accept the phrase.

Examples:

Input: GAME

Output: 20

Input: `PHONE` Output: `27`

GAME takes 2 + 2 + 3 + 7 + 6 = 20 button presses. One of the sequences of button presses that gives this is DEUEDDEUURRRRERDDDDE (D = down arrow, E = enter, etc). There are other solutions but this is the solution that uses the minimum number of button presses.

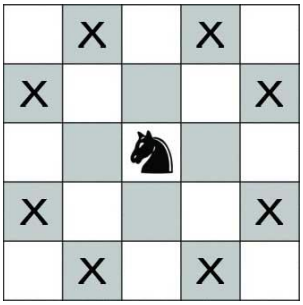
Test your program with the following cases:

a) `CAT`
b) `HAMSTER`
c) `HELLO WORLD`
d) `DOMO ARIGATO MR. ROBOTO`

Question 4: Knight Moves

Given a chessboard with N rows and M columns, a start position A and an end position B, the program you must write must calculate the fewest number of moves it takes a **knight*** to travel from A to B. In all the test cases, it is always possible for the knight to travel from A to B.

The first input line contains N and M. The second line contains two integers, the row and column, respectively, of square A. The third line contains two integers, the row and column, respectively, of square B. Note that the rows are numbered 1 to N, and the columns from 1 to M. All input values are separated by a space.



*In one move, a knight can move two squares vertically and one square horizontally, or two squares horizontally and one square vertically. The image shows all the legal moves that the knight can make.

Examples:

Input:
```
8 8
4 4
4 2
```
Output: `2`

Input:
```
3 3
1 1
2 1
```
Output: `3`

Input:
```
3 6
2 1
1 6
```
Output: `4`

In the third example, the chessboard is 3 by 6, the starting position of the knight is row 2 column 1 and the end position row 1 column 6.

Test your program with the following cases:

a)
```
11 2
1 1
11 2
```

b)
```
8 8
1 3
7 6
```

c)
```
40 50
4 9
38 47
```

d)
```
100 100
15 2
100 100
```