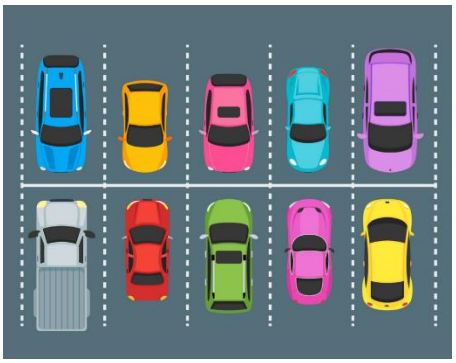
Challenge Set#1

(Advanced Python)



Problem#1

Problem J2: Occupy parking

Problem Description

You supervise a small parking lot which has N parking spaces.

Yesterday, you recorded which parking spaces were occupied by cars and which were empty.

Today, you recorded the same information.

How many of the parking spaces were occupied both yesterday and today?

Input Specification

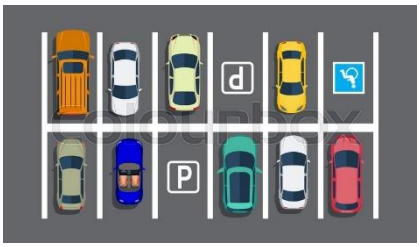
The first line of input contains the integer N ($1 \leq N \leq 100$). The second and third lines of input contain N characters each. The second line of input records the information about yesterday's parking spaces, and the third line of input records the information about today's parking spaces. Each of these $2N$ characters will either be `C` to indicate an occupied space or `.` to indicate it was an empty parking space.

Output Specification

Output the number of parking spaces which were occupied yesterday and today.

Sample Input 1

```
5
CC..C
.CC..
```



Output for Sample Input 1

```
1
```

Explanation of Output for Sample Input 1

Only the second parking space from the left was occupied yesterday and today.

Sample Input 2

```
7
CCCCCCC
C.C.C.C
```

Output for Sample Input 2

```
4
```

Explanation of Output for Sample Input 2

The first, third, fifth, and seventh parking spaces were occupied yesterday and today.

Problem#2

Use the **`time`** library and the Python Turtle to create a game that times the how long it takes the user to click on two randomly placed objects that appear on the screen (after a count down). Have the game repeat and report their time so that the user can see if the are improving. You might want to search up: "Measuring elapsed time using Python.

More exercise Below…

Problem#3

Problem Description
Antonia and David are playing a game.

Each player starts with 100 points.

The game uses standard six-sided dice and is played in rounds. During one round, each player rolls one die. The player with the lower roll loses the number of points shown on the higher die. If both players roll the same number, no points are lost by either player.

Write a program to determine the final scores.

Input Specification
The first line of input contains the integer n $(1 \leq n \leq 15)$, which is the number of rounds that will be played. On each of the next n lines, will be two integers: the roll of Antonia for that round, followed by a space, followed by the roll of David for that round. Each roll will be an integer between 1 and 6 (inclusive).

Output Specification
The output will consist of two lines. On the first line, output the number of points that Antonia has after all rounds have been played. On the second line, output the number of points that David has after all rounds have been played.

Sample Input
```
4
5 6
6 6
4 3
5 2
```

Output for Sample Input
```
94
91
```

Explanation of Output for Sample Input
After the first round, David wins, so Antonia loses 6 points. After the second round, there is a tie and no points are lost. After the third round, Antonia wins, so David loses 4 points. After the fourth round, Antonia wins, so David loses 5 points. In total, Antonia has lost 6 points and David has lost 9 points.

Problem#4

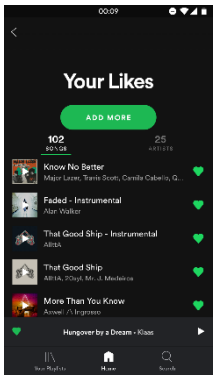
Problem J2: Do the Shuffle

Problem Description

Those tiny music machines that play your digital music are really computers that keep track of and play music files. The *CCC music player* (C^3MP) is currently in development and will be hitting the stores soon! In this problem, you have to simulate a C^3MP.

The C^3MP music player will hold 5 songs in memory, whose titles will always be "A", "B", "C", "D" and "E". The C^3MP also keeps track of a *playlist*, which is an ordering of all the songs. The C^3MP has 4 buttons that the user will press to rearrange the playlist and play the songs.

Initially, the C^3MP playist is "A, B, C, D, E". The 4 control buttons do the following:



- Button 1: move the first song of the playlist to the end of the playlist.
  For example: "A, B, C, D, E" will change to "B, C, D, E, A".
- Button 2: move the last song of the playlist to the start of the playlist.
  For example, "A, B, C, D, E" will change to "E, A, B, C, D".
- Button 3: swap the first two songs of the playlist.
  For example, "A, B, C, D, E" will change to "B, A, C, D, E".
- Button 4: stop rearranging songs and output the playlist.

You need to write a program to simulate a CCC music player. Your program should repeatedly ask for two positive integers b and n. Here b represents the button number that the user wants to press, $1 \leq b \leq 4$, and n represents the number of times that the user wants to press button b. You can assume that n always satisfies $1 \leq n \leq 10$.

The input will always finish with the pair of inputs ($b = 4, n = 1$) when this happens, you should print the order of songs in the current playlist and your program should end. You can assume that the user will only ever press button 4 once.

Sample session (user input in *italics*)	**Explanation**
	(initial playlist is "A, B, C, D, E")
`Button number:` *2*	
`Number of presses:` *1*	($b = 2, n = 1$ so "A, B, C, D, E" changed to "E, A, B, C, D")
`Button number:` *3*	
`Number of presses:` *1*	($b = 3, n = 1$, so "E, A, B, C, D" changed to "A, E, B, C, D")
`Button number:` *2*	
`Number of presses:` *3*	($b = 2, n = 3$, so "A, E, B, C, D" changed to "B, C, D, A, E")
`Button number:` *4*	
`Number of presses:` *1*	($b = 4, n = 1$) When this happens, you should output the playlist.

Output for Sample Input Session

```
B C D A E
```