Lab 3D

Problem 1: Formula 1 Grid Shifts

Charles Leclerc, driving for Ferrari, has just won thrilling races in Monza and Monaco. To celebrate, he needs to apply a series of strategic maneuvers on a grid-based track to demonstrate his tactical prowess. The track is represented by a 2D grid of size `m × n`, and each cell in the grid denotes a specific checkpoint. Task: Given a 2D grid of size `m × n` and an integer `k`, representing the number of shift operations, perform the following:

Shift Operation:

- The element at `grid[i][j]` moves to `grid[i][j + 1]`.
- The element at `grid[i][n - 1]` moves to `grid[i + 1][0]`.
- The element at `grid[m - 1][n - 1]` moves to `grid[0][0]`.

Return the 2D grid after applying the shift operation `k` times.

Input:

- The first line contains three integers, `m` , `n` and `k` representing the number of rows and number of columns and no of shifts
- Then a 2D grid of size `m × n` where each element is an integer

Output:

- A 2D grid of size `m × n` after `k` shift operations.

Constraints:

- `n ≤ 100` and `m ≤ 100`
- `-10^6 ≤ arr[i] ≤ 10^6`
- `0 ≤ k ≤ 50`

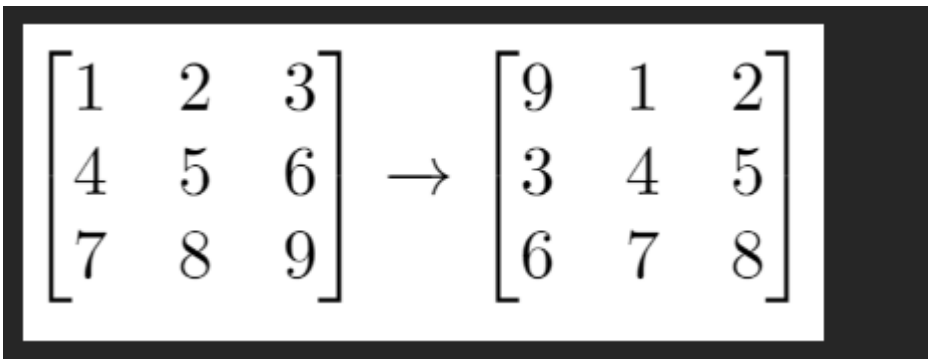
Example:

Input:

```
3 3 1
1 2 3
4 5 6
7 8 9
```

Output:

```
9 1 2
3 4 5
6 7 8
```



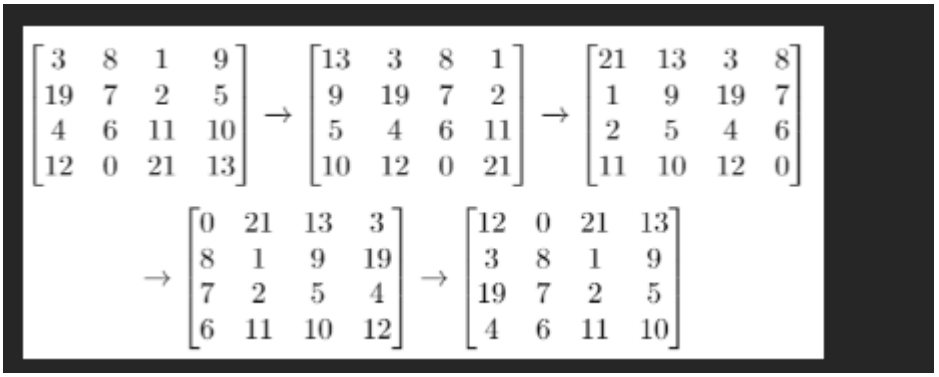
Example 2:

Input:

```
4 4 4
3 8 1 9
19 7 2 5
4 6 11 10
12 0 21 13
```

Output:

```
12 0 21 13
3 8 1 9
19 7 2 5
4 6 11 10
```



Problem 2: Wizard Chess Challenge

Objective: Your task is to simulate a simplified version of Wizard Chess on an 8x8 chessboard. In this game, Harry and Ron are strategizing the placement of their knights and pawns. You need to determine if any knight can attack another knight or a pawn, and print the appropriate result.

The board has exactly 4 knights and 8 pawns placed randomly. Your goal is to determine the outcome of a potential move by the knights:

- If any knight can attack another knight in a single move, print 1.
- If no knight attacks another knight, but a knight can attack a pawn, print 2.
- If no knight can attack any piece, print 3.

Input:

- Eight lines of input, each containing exactly 8 characters, representing the chessboard. The characters can be K (knight), P (pawn), or E (empty square).

Output:

- A single line containing 1, 2, or 3 based on the rules outlined above.

Constraints:

- The board will always contain exactly 4 knights and 8 pawns.

Good luck!

Example:

Input:

```
KPEEEEEE
PKEEEEEE
EEPPEEEE
EEEEEEEE
EEPKEEEE
EEEEEEEE
EEEEEPEE
EEEEKPEE
```

Output:

```
2
```

Explanation:

In this case, the knight at position (2, 2) can attack the pawn at position (3, 4) in a single move.

Input:

```
KPEPEKEE
EEPKEEEE
EEEEEEEE
EEEPEEEE
EEEKPEEE
EEPPEEEE
EEEEEEEE
EEEEEEEE
```

Output:

```
1
```

Explanation:

In this case, the knight at position (1, 6) can attack the knight at position (2, 4) in a single move.

Problem 3: The Traveler and his resources

The Traveler and his companion are exploring various realms and collecting daily resources. His companion has a challenge for the Traveler: determine if there is any contiguous period where the sum of the collected resources exactly matches a specified target value. Additionally, companion wants to ensure that the contiguous subarray has at least two elements, meaning the starting and ending indices should not be the same.

Given an array representing daily resource values and a target value, your task is to determine if there exists any contiguous subarray whose sum equals the target value. If such a subarray exists, print the starting and ending indices of the first such subarray (0-based indices) where the indices are different. If no such subarray exists, print -1 -1.

Input:

- The first line contains an integer, `N` representing the number of days the Traveler tracked the resource values.
- The second lines contains `N` integers, each representing the resource value collected on a particular day.
- The third line contains an integer `target` representing the target sum the Traveler needs to find.

Output:

- Print the starting and ending indices of the first contiguous subarray whose sum equals the target value, with the condition that the starting and ending indices are different. If no such subarray exists, print -1 -1. If multiple such indices exist print the first one occuring lexicographically

Constraints:

- 1 ≤ `N` ≤ 1000.
- -10^6 ≤ `Arr[i]` ≤ 10^6
- -10^6 ≤ `target` ≤ 10^6

Example 1:

Input:

```
5
1 2 3 -1 5
5
```

Copy

Output:

```
0 3
```

The first occuring sum of 5 is from index 0 to index 3, 1 + 2 + 3 + -1 =5

Example 2:

Input:

```
4
1 2 -1 4
7
```

Copy

Output:

```
-1 -1
```

Good luck!
