Algorithms Laboratory (CS29203)
Assignment 0: PDS brush-up (not to be graded)
Department of CSE, IIT Kharagpur

1st August 2024

Question-1

(40 points)
Consider a set of n number of dice arranged in two rows as shown in the figure below, where both top and bottom rows have the same number of dice. The displayed numbers of the corresponding dice in the two rows may or may not match. For example in the first part of the figure, the top row displays the numbers $2, 1, 2, 4, 2, 2$, whereas the bottom row displays the numbers $5, 2, 6, 2, 3, 2$. Our goal is to exchange the dice in the same column(s) so that the displayed numbers either in the top or the bottom row become the same, and this needs to be performed with *minimum number of exchanges*. In the bottom part of the figure below, exchange of dice are performed in 2nd and fourth column, which results in the top row displaying the same number (which is 2).

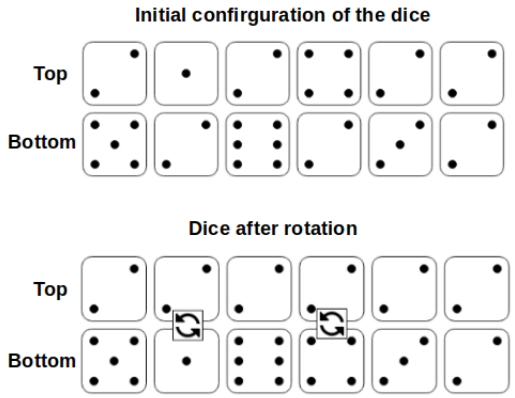
Your code should take the numbers in the top and bottom rows as two integer arrays, and return the minimum number of exchanges required as an integer number. If there is no such exchange possible to make the two rows equal, then return -1.

Ex1:

```
Input: top = [2,1,2,4,2,2], bottom = [5,2,6,2,3,2]
Output: 2
```

Ex2:

```
Input: top = [3,5,1,2,3], bottom = [3,6,3,3,4]
Output: -1
```



Question-2

(20 points)
Suppose that you are given a string consisting of lowercase English alphabets and parentheses. The parentheses are balanced, that is there are matched number of opening and closing braces. Your task is to reverse the strings in each pair of matching

parentheses, starting from the innermost one. The final result should not have any parentheses.

Ex1:

```
Input string: "(algorithm)"
Output string: mhtirogla
```

Ex2:

```
Input string: "(u(copy)i)"
Output string: icopyu
```

Ex3:

```
Input string: "(re(mp(li))oc)"
Output string: compiler
```

Question-3

(30 points)
Consider a text file having the following format, which is typically used in computer graphics applications to store 3D point and geometry:

```
# This is a file for geometric data
# Version 2.3.1
# VCGLib generated
Points 4
Triangles 3
12.5  3.36  10.2
7.3  20.91  9.9
11.2  14.02  8.9
23.1  0.56  4.5
1 2 3
1 2 4
2 3 4
```

In the file, the lines beginning with '#' are comment lines, which should be ignored for processing. Next, the line starting with *'Points'* denotes the number of 3D points, which is written as an integer number next to it separated by a space. The next line starting with *'Triangles'* denotes the number of triangles that are formed from the points, and the integer number is written next to it separated by a space. Then the 3D points are listed in the preceding lines as x, y and z coordinates separated by spaces. For example in this file, there are 4 points, which are written in the next 4 lines. After the points, triangle vertices are stored as numbers denoting the indices of the declared points. For example, this file has 3 triangles where the first triangle has the first, second, and third declared points as the three vertices. That means, the vertices of the first triangle are $(12.5, 3.36, 10.2), (7.3, 20.91, 9.9), (11.2, 14.02, 8.9)$. Similarly the vertices for the second triangle are $(12.5, 3.36, 10.2), (7.3, 20.91, 9.9), (23.1, 0.56, 4.5)$, and so on.

Your task is to read a given text file of the same format as described before, read the points and triangles, and compute the *surface normal* for each triangle. The surface normal of a triangle is computed as the cross product of two sides of the triangle. That is, if a triangle has vertices $P_1(x_1, y_1, z_1), P_2(x_2, y_2, z_2), P_3(x_3, y_3, z_3)$, then the following vectors are computed: $U = P_2 - P_1$, $V = P_3 - P_1$. Then the x, y, and z components of the normal are computed as:

$$N_x = U_y V_z - U_z V_y$$
$$N_y = U_z V_x - U_x V_z$$
$$N_z = U_x V_y - U_y V_x,$$

where U_x, U_y, U_z represents the x, y, and z component of the vecotor U (and same for V as well).

After computing the surface normal for each triangle, you have to create a new text file, and write the surface normal information in the file as $N_x N_y N_z$ format, as well as display the file content in the output window.

Example:

```
Enter the file name to be read: myGeometryPoints.txt
Enter the output file name: mySurfaceNormals.txt

Number of points in the file: 10
Number of triangles in the file: 15

Computing the surface normals...

Surface normals have been computed and written to the file mySurfaceNormals.txt

List of surface normals:
-0.34  0.45  3.67
1.25  4.21  -1.33
7.32  2.11  9.76
.
.
.
```