**Q1.** Write a program that takes cost price and selling price as input and displays whether the transaction is a Profit or a Loss or Neither.

INPUT FORMAT :

The first line contains the cost price. The second line contains the selling price.

OUTPUT FORMAT

Print "Profit" if the transaction is a profit or "Loss" if it is a loss. If it is neither profit nor loss, print "Neither".

Solution of Q1 :

c=int(input())

s=int(input())

if(c<s):

print("Profit")

if(c>s):

print("Loss")

if(c==s):

print("Neither")

Q2 . Let us assume paper as the plane and a letter as a curve on the plane, then each letter divides the plane into regions. For example letters "A", "D", "O", "P", "R" divide the plane into two regions so we say these letters each have one hole. Similarly, letter "B" has two holes and letters such as "C", "E", "F", "K" have no holes. We say that the number of holes in the text is equal to the total number of holes in the letters of the text. Write a program to determine how many holes are in a given text.  
  
Input Format:  
The only line contains a non-empty text composed only of uppercase letters of English alphabet.  
  
Output Format:  
output a single line containing the number of holes in the corresponding text.  
  
Example-  
Input: DRINKEATCODE

Output: 5

Solution Q2:

list\_st = []

zero\_hole = ["C","E","F","G","H","I","J","K","L","M","N","S","T","U","V","W","X","Y","Z"]

one\_hole = ["A","D","O","P","Q","R"]

two\_hole = ["B"]

x = input()

c = 0

for j in x:

if(j in one\_hole):

c = c+1

if(j in two\_hole):

c=c+2

print(c)

Q3. In this assignment, given a number **n**, you have to write a program that generates a dictionary **d** which contains **(i, i\*i)**, where **i** is from **1 to n (both included)**.  
Then you have to just print this dictionary **d**.  
  
Example:  
Input: 4  
  
will give output as  
{1: 1, 2: 4, 3: 9, 4: 16}  
  
**Input Format:**  
Take the number **n** in a single line.  
  
**Output Format:**  
Print the dictionary **d** in a single line.

Solution Q3:

n=int(input())

d=dict()

for i in range(1,n+1):

d[i]=i\*i

print(d)

Q4: Create a list of numbers from 1 to 50 named list\_1. The numbers should be present in the increasing order: Ex list\_1 = [1,2,3,4,5,....,50] i.e. index zero should be 1, index one should be 2, index two should be 3 and so on.

Given an input let's say a, you have to print the number of elements of list\_1 which are divisible by a, excluding the element which is equal to a.

Input: Number a

Output: In a single line, the number of elements which are divisible by a with total count.

Solution of Q4:

l = []

for i in range(1,51):

l.append(i)

a= int(input())

a = int(a)

count = 0

for i in l:

if(i%a==0 and i!=a):

count+=1

print(i)

print(count)

Q5: With the number of **rows i.e. r** and **columns i.e. c** as the input and your job is to create a matrix of size rxc. Also, the matrix should have elements starting from **1** to **rxc with an increment of one**in row manner.  
  
**Example:** if r = 2 and c = 3  
then the output is   
1 2 3  
4 5 6  
  
**Input Format:**  
Two numbers **r** and **c** in a single line separated by a space.  
  
**Output Format:**  
Elements of the generated matrix.  
Each row should be printed in a new line with each element separated by a space.

Solution Q5:

a,b=map(int,input().split())

count=1

m = []

for i in range(1,a+1):

l = []

for j in range(1,b+1):

l.append(count)

count+=1

m.append(l)

for i in range(a):

for j in range(b):

if(j==b-1):

print(m[i][j], end="")

else:

print(m[i][j], end=" ")

if(i!=a-1):

print()

Q 6: There is a robot which wants to go the charging point to charge itself.   
The robot moves in a 2-D plane from the original point (0,0).  The robot can move toward **UP, DOWN, LEFT** and **RIGHT** with given steps.  
The trace of robot movement is shown as the following:

**UP 5**

**DOWN 3**

**LEFT 3**

**RIGHT 2**

Then, the output of the program should be: **2**

The numbers after the direction are steps. Write a program to compute the distance between the current position after a sequence of movement and original point. If the distance is a float, then just print the nearest integer (use **round()** function for that and then convert it into an integer).  
  
**Input Format:**  
The first line of the input contains a number n which implies the number of directions to be given. The next n lines contain the direction and the step separated by a space.  
  
**Output Format:**  
Print the distance from the original position to the current position.

Solution of Q6:

import math

pos = [0,0]

n = int(input())

for i in range(n):

s = input()

movement = s.split(" ")

direction = movement[0]

steps = int(movement[1])

if direction=="UP":

pos[0]+=steps

elif direction=="DOWN":

pos[0]-=steps

elif direction=="LEFT":

pos[1]-=steps

elif direction=="RIGHT":

pos[1]+=steps

else:

pass

print(int(round(math.sqrt(pos[1]\*\*2+pos[0]\*\*2))))

Q7: Write a Python program to add the digits of a positive integer repeatedly until the result has a single digit.  
  
Input Format:  
The first line of the input contains a number n.  
  
Output:  
Print the resultant number  
  
Example:  
Input: 48  
  
Output: 3  
  
Explanation: If you add digits 4 and 8, you will get 12. Again adding 1 and 2 will give 3 which is a single digit and hence the answer.

Solution Q7:

def add\_digits(num):

return (num - 1) % 9 + 1 if num > 0 else 0

num = int(input())

print(add\_digits(num))

Q8 :With a given list **L**, write a program to print this list L after removing all duplicate values with original order reserved.

**Example:**  
  
If the input list is  
  
12 24 35 24 88 120 155 88 120 155  
  
Then the output should be  
  
12 24 35 88 120 155  
  
**Explanation:**  
Third, seventh and ninth element of the list **L** has been removed because it was already present.  
  
**Input Format:**  
In one line take the elements of the list **L** with each element separated by a space.  
  
**Output Format:**  
Print the elements of the modified list in one line with each element separated by a space.

Solution of Q8:

def removeDuplicate( li ):

newli=[]

seen = set()

for item in li:

if item not in seen:

seen.add( item )

newli.append(item)

return newli

li=[]

li= list(map(int, input ().split ()))

x = removeDuplicate(li)

for i in x:

print(i,end=" ")

Q9: You are provided with a number **D** containing only digits 0's and 1's. Your aim is to convert this number to have all the digits same. For that, you will change exactly one digit i.e. from **0 to 1** or from **1 to 0**. If it is possible to make all digits equal (either all 0's or all 1's) by flipping exactly 1 digit then output "YES", else print "NO" (quotes for clarity).  
  
Input Format:

The first line of the input contains the number **D** made of only digits 1's and 0's.  
  
Output:

Print 'YES' or 'NO' depending on whether its possible to make it all 0s or 1s or not.   
  
Example-1:  
  
Input: 101  
Output: YES  
  
Example-2:  
  
Input: 11  
Output: NO  
  
Explanation:  
In the first example, it is possible to make all the digits same by flipping the middle digit from 0 to 1. In the second example it is not possible.

Solution of Q9:

numbers = []

ls = []

x = input()

li = str(x)

for j in li:

ls.append(int(j))

numbers.append(ls)

for j in numbers:

count\_z = 0

count\_o = 0

for k in j:

if(k==1):

count\_o += 1

if(k==0):

count\_z += 1

if((count\_o == 1) or (count\_z == 1)):

print("YES")

else:

if((count\_o == 0) or (count\_z == 0)):

print("NO")

else:

print("NO")

Q10: Given a square matrix with **n** rows and **n** columns, you have to write a program to rotate this matrix such that each element is shifted by one place in a clockwise manner.  
For example, given the following matrix  
  
1 2 3  
4 5 6  
7 8 9  
  
The output should be  
  
4 1 2  
7 5 3  
8 9 6  
  
Input Format:  
The first line of the input contains a number n representing the number of rows and columns.  
After this, there are **n** rows with each row containing **n** elements separated by a space  
  
Output Format:  
Print the elements of the modified matrix with each row in a new line and all the elements in each row is separated by a space.

Solution of Q10:

# Python program to rotate a matrix

# Function to rotate a matrix

def rotateMatrix(mat):

if not len(mat):

return

"""

top : starting row index

bottom : ending row index

left : starting column index

right : ending column index

"""

top = 0

bottom = len(mat)-1

left = 0

right = len(mat[0])-1

while left < right and top < bottom:

# Store the first element of next row,

# this element will replace first element of

# current row

prev = mat[top+1][left]

# Move elements of top row one step right

for i in range(left, right+1):

curr = mat[top][i]

mat[top][i] = prev

prev = curr

top += 1

# Move elements of rightmost column one step downwards

for i in range(top, bottom+1):

curr = mat[i][right]

mat[i][right] = prev

prev = curr

right -= 1

# Move elements of bottom row one step left

for i in range(right, left-1, -1):

curr = mat[bottom][i]

mat[bottom][i] = prev

prev = curr

bottom -= 1

# Move elements of leftmost column one step upwards

for i in range(bottom, top-1, -1):

curr = mat[i][left]

mat[i][left] = prev

prev = curr

left += 1

return mat

# Utility Function

def printMatrix(mat,n):

for i in range(n):

for j in range(n):

if(j==n-1):

print(mat[i][j],end="")

else:

print(mat[i][j],end=" ")

if(i!=n-1):

print()

n = int(input())

matrix = []

for i in range(1,n+1):

l = list(map(int, input ().split ()))

matrix.append(l)

matrix = rotateMatrix(matrix)

# Print modified matrix

printMatrix(matrix,n)