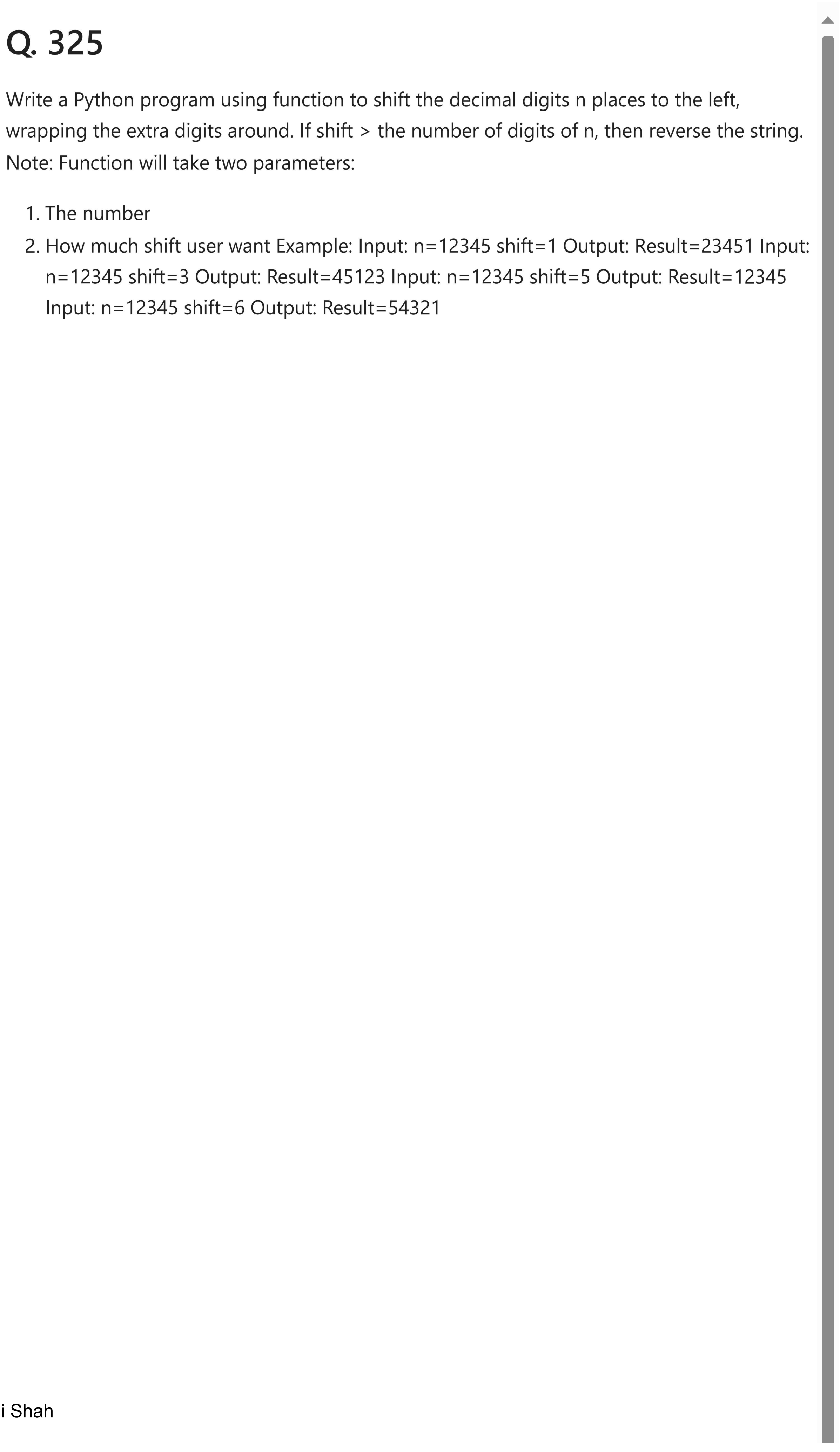
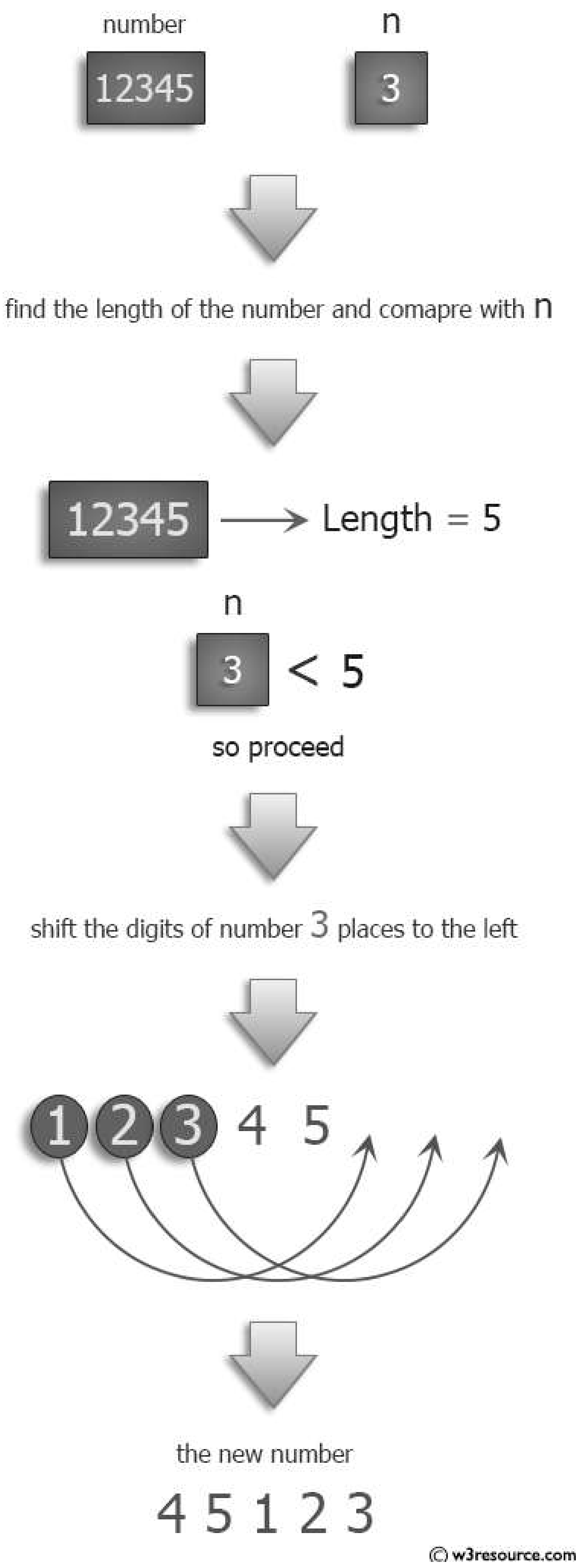


Q. 325

Write a Python program using function to shift the decimal digits n places to the left, wrapping the extra digits around. If shift > the number of digits of n, then reverse the string. Note: Function will take two parameters:

1. The number
2. How much shift user want Example: Input: n=12345 shift=1 Output: Result=23451 Input: n=12345 shift=3 Output: Result=45123 Input: n=12345 shift=5 Output: Result=12345 Input: n=12345 shift=6 Output: Result=54321



find the length of the number and comapre with n

so proceed

shift the digits of number 3 places to the left

the new number

In [1]:

```
#License: https://bit.ly/3oLErEI

def test(n, shift):
    s = str(n)
    if shift > len(s):
        return s[::-1]
    return s[shift:] + s[:shift]

print("Shift the decimal digits n places to the left. If shift > the number of digi

n = 12345
shift = 1
print("\nn =",n," and shift =",shift)
print("Result = ",test(n, shift))
n = 12345
shift = 2
print("\nn =",n," and shift =",shift)
print("Result = ",test(n, shift))
n = 12345
shift = 3
print("\nn =",n," and shift =",shift)
print("Result = ",test(n, shift))
n = 12345
shift = 5
print("\nn =",n," and shift =",shift)
print("Result = ",test(n, shift))
n = 12345
shift = 6
print("\nn =",n," and shift =",shift)
print("Result = ",test(n, shift))
```

```
Shift the decimal digits n places to the left. If shift > the number of digits of
n, reverse the string.:

n = 12345  and shift = 1
Result =  23451

n = 12345  and shift = 2
Result =  34512

n = 12345  and shift = 3
Result =  45123

n = 12345  and shift = 5
Result =  12345

n = 12345  and shift = 6
Result =  54321
```

Q. 326

Write a Python programme that accepts a string and calculate the number of uppercase letters, lowercase letters and number of digits. For example, Input: Hello Pyth@n is 100% easy Output: Uppercase letters : 2 Lowercase letters : 14 Digits : 3

In [3]:

```
def count_chars(input_string):
    uppercase_count = 0
    lowercase_count = 0
    digit_count = 0
    for char in input_string:
        if char.isupper():
            uppercase_count += 1
        elif char.islower():
            lowercase_count += 1
        elif char.isdigit():
            digit_count += 1

    print("Uppercase letters:", uppercase_count)
    print("Lowercase letters:", lowercase_count)
    print("Digits:", digit_count)

# Get input from the user
user_input = input("Enter a string: ")

# Call the function with the user input
count_chars(user_input)
```

```
Enter a string: Hello Pyth@n is 100% easy
Uppercase letters: 2
Lowercase letters: 14
Digits: 3
```

Q. 327.

Write a python program to check the validity of a Password. Primary conditions for password validation:

1. Minimum 8 characters.
2. The alphabet must be between [a-z]
3. At least one alphabet should be of Upper Case [A-Z]
4. At least 1 number or digit between [0-9]
5. At least 1 character from [_ or @ or $]Examples : Input : Ram@_f1234Output : ValidPasswordInput : Rama_fo$ab Output: Invalid Password Explanation: Number is missing Input: Rama#fo9c Output: Invalid Password Explanation: Must consist from _ or @ or $

In [6]:

```
def is_valid_password(password):
    length_condition = len(password) >= 8
    uppercase_condition = False
    digit_condition = False
    special_condition = False

    for char in password:
        if char.isupper():
            uppercase_condition = True
        elif char.isdigit():
            digit_condition = True
        elif char in '_@$':
            special_condition = True

    return (length_condition and uppercase_condition and digit_condition and specia
            "Valid Password" if length_condition and uppercase_condition and digit_
            else "Invalid Password")

# Get input from the user
user_password = input("Enter a password: ")
```

```
# Check the validity of the password and print the result
is_valid, message = is_valid_password(user_password)
print(message)
```

```
Enter a password: Rama#fo9c
Invalid Password
```

Q. 328

Write a Python program to return another string similar to the input string, but with its case inverted. For example, input of “Mr. Ed” will result in “mR. eD” as the output string. Note: Use of built in swapcase function is prohibited.

In [7]:

```
def invert_case(input_string):
    inverted_string = ""
    for char in input_string:
        if char.islower():
            inverted_string += char.upper()
        elif char.isupper():
            inverted_string += char.lower()
        else:
            inverted_string += char  # No change for non-alphabetic characters

    return inverted_string

# Get input from the user
user_input = input("Enter a string: ")

# Get the inverted case string and print the result
inverted_result = invert_case(user_input)
print("Inverted Case:", inverted_result)
```

```
Enter a string: Mr. Ed
Inverted Case: mR. eD
```

Q. 329

Dr. Prasad is opening a new world class hospital in a small town designed to be the first preference of the patients in the city. Hospital has N rooms of two types – with TV and without TV, with daily rates of R1 and R2 respectively. However, from his experience Dr. Prasad knows that the number of patients is not constant throughout the year, instead it follows a pattern. The number of patients on any given day of the year is given by the following formula – (6-M)^2 + |D-15| , where M is the number of month (1 for jan, 2 for feb ...12 for dec) and D is the date (1,2...31). All patients prefer without TV rooms as they are cheaper, but will opt for with TV rooms only if without TV rooms are not available. Hospital has a revenue target for the first year of operation. Given this target and the values of N, R1 and R2 you need to identify the number of TVs the hospital should buy so that it meets the revenue target. Assume the Hospital opens on 1st Jan and year is a non-leap year. Constraints Hospital opens on 1st Jan in an ordinary year 5 <= Number of rooms <= 100 500 <= Room Rates <= 5000 0 <= Target revenue < 90000000 Input Format: • First line provides an integer N that denotes the number of rooms in the hospital. • Second line provides the rates of rooms with TV (R1). • Third line provides the rates of rooms without TV

(R2). • Fourth line provides the revenue target. Output: Minimum number of TVs the hospital needs to buy to meet its revenue target. If it cannot achieve its target, print the total number of rooms in the hospital. Test Case Example-1 : Input 20 1500 1000 7000000 Output 14 Explanation Using the formula, the number of patients on 1st Jan will be 39, on 2nd Jan will be 38 and so on. Considering there are only twenty rooms and rates of both type of rooms are 1500 and 1000 respectively, we will need 14 TV sets to get revenue of 7119500. With 13 TV sets Total revenue will be less than 7000000 Example-2 : Input 10 1000 1500 10000000 Output 10 Explanation In the above example, the target will not be achieved, even by equipping all the rooms with TV. Hence, the answer is 10 i.e. total number of rooms in the hospital.

In [9]:

```
# Python3 program for the above approach

# Function that returns number of
# patient for a day in a month
def getPatients( M, D):

        return ((6 - M) * (6 - M)) + abs(D - 15);

# Function that count the TVs with
# given amount of revenue target
def countTVs( n, r1, r2, target):


        # Days in each month
        days = [ 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 ];

        # Check all possible combinations
        for tvs in range(n + 1):

                # Stores the current target
                current_target = 0;

                for m in range(1, 13):

                        for d in range(1, 1 + days[m]):

                                # Number of patients
                                # on day d of month m
                                np = getPatients(m, d);

                                # Patients cannot be
                                # exceed number of rooms
                                np = min(np, n);

                                # If the number of patient is
                                # <= count of rooms without tv
                                if (np <= n - tvs) :

                                        # All patients will opt
                                        # for rooms without tv
                                        current_target += np * r2;


                                # Otherwise
                                else :

                                        # Some will opt for
                                        # rooms with tv and
                                        # others without tv
                                        current_target += ((n - tvs) * r2 + ((np -



                # If current target meets
                # the required target
                if (current_target >= target):
                        break;



        # Pr the count of TVs
        print(min(tvs, n));


# Driver Code
N = 10
R1 = 100;
R2 = 1500;
target = 10000000;

# Function Call
countTVs(N, R1, R2, target);

# This code is contributed by phasing17.
```

```
10
```

Q. 330

Write a Python program to create a Caesar encryption. Note: In cryptography, a Caesar cipher, also known as Caesar's cipher, the shift cipher, Caesar's code or Caesar shift, is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a right shift of 3, A would be replaced by D, E would become H, and so on. The method is named after Julius Caesar, who used it in his private correspondence. For Example: Input Text : LJIET ENG Shift : 3 Cipher: OMLHW HQJ

In [10]:

```
def caesar_encrypt(plain_text, shift):
    encrypted_text = ""
    for char in plain_text:
        if char.isalpha():
            # Determine if the character is uppercase or lowercase
            is_upper = char.isupper()
            # Apply the Caesar shift to the character
            shifted_char = chr((ord(char) - ord('A' if is_upper else 'a') + shift)
            encrypted_text += shifted_char
        else:
            # Keep non-alphabetic characters unchanged
            encrypted_text += char

    return encrypted_text

# Get input from the user
input_text = input("Enter the text to encrypt: ")
shift_amount = int(input("Enter the shift amount: "))

# Encrypt the input text and print the result
```

```
encrypted_result = caesar_encrypt(input_text, shift_amount)
print("Cipher:", encrypted_result)
```

```
Enter the text to encrypt: LJIET ENG
Enter the shift amount: 3
Cipher: OMLHW HQJ
```

Q. 331.

Write a program to check if two strings are balanced. For example, strings s1 and s2 are balanced if all the characters in the s1 are present in s2 and length of s1 & s2 should be same. The character's position doesn't matter. Example : s1 = hello s2 = olleh Balanced

In [11]:

```
def are_strings_balanced(s1,s2):
    if len(s1)!=len(s2):
        return False
    for char in s1:
        if char not in s2:
            return False
    return True
s1='hello'
s2='olleh'
if are_strings_balanced(s1,s2):
    print('balanced')
else:
    print('not balanced')
```

```
balanced
```

In [ ]: