Q1

Advanced Coding

Q1. Problem statement:

Given two non-negative integers n1 and n2, where n1 <n2. The task is to find the total number of integers in the range interval [n1, n2] [both inclusive] which have no repeated digits.

For e.g.

Suppose n1 = 11 and n2 = 15.

There is the number 11, which has repeated digits, but 12, 13, 14, and 15 have no repeated digits. So, the output is 4.

Input	Output
11 -- Value of n1 15 -- Value of n2	4
101 -- Value of n1 200 -- Value of n2	72

```python
def checkRepetitive(start: int, end: int) -> list:
    count = 0
    for i in range(start, end+1):
        if len(list(set(str(i)))) == len(str(i)):
            count += 1

    return count


print(checkRepetitive(int(input()), int(input())))
```

Q2

Q2. Problem statement:

Given an array Arr[] of N integers and a positive integer K. The task is to cyclically rotate the array clockwise by K.

Note: Keep the first position of the array unaltered.

Example	Input	Output	Explanation

Example 1	5 -- Value of N
	{10, 20, 30, 40, 50} -- Elements of Arr[]
	2 -- Value of K
		40 50 10 20 30	Arr[] = {10, 20, 30, 40, 50} and K = 2 (Two cyclical rotations)
			After 1st rotation = {10, 50, 20, 30, 40}
			After 2nd rotation = {10, 40, 50, 20, 30}

Example 2	4 -- Value of N
	{10, 20, 30, 40} -- Elements of Arr[]
	1 -- Value of K
		40 10 20 30	Arr[] = {10, 20, 30, 40} and K=1 (One cyclical rotation)
			After 1st rotation = {10, 40, 20, 30}

```python
def rotate(arr: list, k: int) -> list:
    for i in range(k):
        arr.insert(0, arr.pop())

    return arr


print(rotate(list(map(int, input().split())), int(input())))
```

Q3

Advanced Coding

Q1. Given an array Arr[] of N integer numbers. The task is to rewrite the array by putting all multiples of 10 at the end of the given array.

Note: The order of the numbers which are not multiples of 10 should remain unaltered, and similarly, the order of all multiples of 10 should be unaltered.

For e.g.

Suppose N = 9 and Arr[]={10, 12, 5, 40, 30, 7, 50, 9, 10}

You have to push all multiple of 10 at the end of the Arr[]

Hence, the output is 12 5 7 9 10 40 30 50 10.

Input	Output
9 .... Value of N 10 12 5 40 30 7 50 9 10 ... Elements of Arr[]	12 5 7 9 10 40 30 50 10
9 ..... Value of N 100 21 5 6 3 7 11 89 10.... Elements of Arr[]	21 5 6 3 7 11 89 100 10

```python
def pushMultiples(arr: list) -> list:
    final = []
    multiples = []
    for ele in arr:
        if ele % 10 == 0:
            multiples.append(ele)
        else:
            final.append(ele)

    final.extend(multiples)
    return final


if __name__ == '__main__':
    print(pushMultiples(list(map(int, input().split()))))
```

Q4

Q2. Given an array Arr[N] of N integers and a positive integer K. The task is to divide the array into two sub-arrays from right after the Kth position and slide the left sub-array of K elements to the end.

Input	Output	Explanation
5 -- Value of N {10, 20, 30, 40, 50} -- Elements of Arr [] 2 -- Value of K	30 40 50 10 20	Arr[] = {10,20,30,40,50} and K=2 (2nd position) Divide array from after 2nd position and add left sub-array {10,20} to the end. So the output is 30 40 50 10 20
4 -- Value of N {10, 20, 30, 40} -- Elements of Arr [] 1 -- Value of K	20 30 40 10	Arr[] = {10, 20, 30, 40} and K=1 (1st position) Divide array from after 1st position and add left sub-array {10} to the end. So the output is 20 30 40 10
4 -- Value of N {10, 20, 30, 40} -- Elements of Arr[] 3 -- Value of K	40 10 20 30	Arr[] = {10, 20, 30, 40} and K=3 (3rd position) Divide array from after 3rd position and add left sub-array {10, 20, 30} to the end. So the output is 40 10 20 30

```python
def pushAtEnd(arr: list, k: int) -> list:
    return arr[k:] + arr[:k]


if __name__ == '__main__':
    print(pushAtEnd(list(map(int, input().split())), int(input())))
```

Q5

Advanced Coding

Q1. For hiring a car, a travel agency charges R1 rupees per hour for the first N hours and then R2 rupees per hour. Given the total time of travel in minutes is X. The task is to find the total traveling cost in rupees.

Note: While converting minutes into hours, ceiling value should be considered as the total number of hours.

For example: If the total travelling time is 90 minutes,

i.e. 1.5 hours, it must be considered as 2 hours.

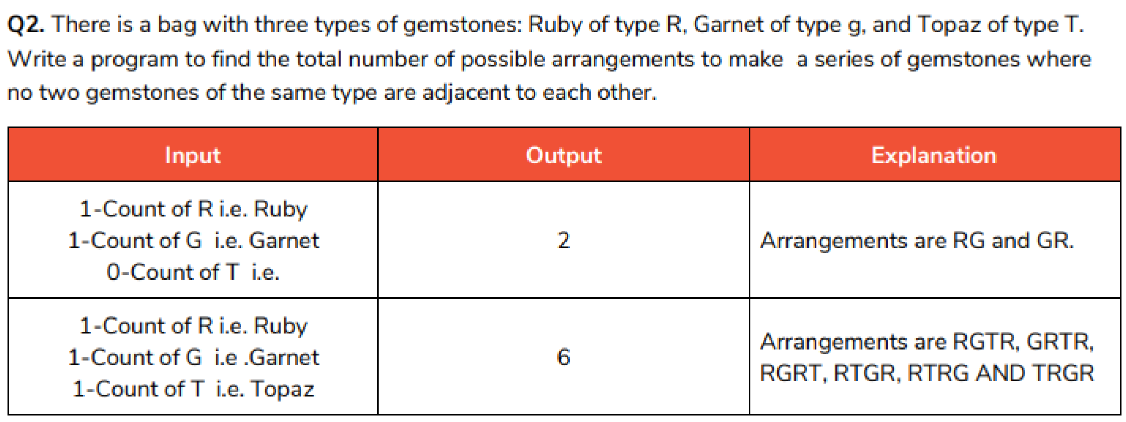
Input	Output	EXplanation
20 ---Value of R1 4 --- Value of N in hours 40 --- Value of R2 300 --- Value of X in minutes	120	Total travelling hours = 300/60 = 5 hours Rupees 20/hours for first 4 hours = 20 * 4 = 80 rupees Rupees 40/hours in 5th hour = 40 * 1 = 40 rupees Hence, the total travelling cost = 80 + 40 = 120 rupees
30 --- Value of R1 5 --- Value of N in hours. 35 --- Value of R2 500 -- Value of X in minutes	290	Total travelling hours = 500/60 = 8.33, Ceiling value of 8.33 = 9 hours Rupees 30/hours for first 5th hours = 30 * 5 = 150 rupees Rupees 35/hours in 5th hour = 35 * 4 = 140 rupees Hence, the total travelling cost = 150 + 140 = 290 rupees
30--- Value of R1 10--- Value of N in hours 35 ---- Value of R2 5 --- Value of X in minutes	30	Total travelling hours = 3/60 = 0.05, Ceiling value of 0.05 = 1 hour Rupees 30/hour for first 10 hours = 30 * 1 = 30 rupees

```python
import math

def calculateCost(r1: int, n: int, r2: int, x: int) -> int:
    x = math.ceil(x / 60)
    if x>=n:
        cost = n * r1
        x -= n
        cost += x * r2
    else:
        cost = x * r1
    return cost


if __name__ == '__main__':
    print(calculateCost(int(input()), int(input()), int(input()),
int(input())))
```

Q6



Q2. There is a bag with three types of gemstones: Ruby of type R, Garnet of type g, and Topaz of type T. Write a program to find the total number of possible arrangements to make a series of gemstones where no two gemstones of the same type are adjacent to each other.

Input	Output	Explanation
1-Count of R i.e. Ruby 1-Count of G i.e. Garnet 0-Count of T i.e.	2	Arrangements are RG and GR.
1-Count of R i.e. Ruby 1-Count of G i.e .Garnet 1-Count of T i.e. Topaz	6	Arrangements are RGTR, GRTR, RGRT, RTGR, RTRG AND TRGR

```python
def gemstoneArrangement(R: int, G: int, T: int, last='') -> int:
    if R == 0 and G == 0 and T == 0:
        return 1

    count = 0

    if last != 'R' and R>0:
        count += gemstoneArrangement(R-1, G, T, 'R')

    if last != 'G' and G > 0:
        count += gemstoneArrangement(R, G-1, T, 'G')

    if last != 'T' and T > 0:
        count += gemstoneArrangement(R, G, T-1, 'T')

    return count


if __name__ == '__main__':
    print(gemstoneArrangement(int(input()), int(input()), int(input())))
```