(1) Given a decimal number N (<= 10^9), convert it into a binary number.

```
Sample Input                Sample Output
11
                            1011
```

(2) Given a binary number, convert it into a decimal number.

```
Sample Input                Sample Output
1011
                            11
```

(3) Given A, B (A, B <= 10^9) print the sum of digits of (A+B)

```
Sample Input                Sample Output
123 19                      7
2021 13                     9
```

Explanation:

For sample-1 : 123 + 19 = 142, sum of digits of 142 = 1 + 4 + 2 = 7
For sample-2 : 2021 + 13 = 2034, sum of digits of 2034 = 2 + 0 + 3 + 4 = 9

(4) Given a string, can you find if it is palindrome or not ?

```
Sample Input                Sample Output
ABBA                        YES
ABABA                       YES
XYZ                         NO
ABCA                        NO
```

(5) Given a string, is it possible to rearrange the letters to make it a palindrome ?

```
Sample Input                Sample Output
AABCB                          YES
ABAB                           YES
ABC                            NO
```

Explanation:

For sample 1, you can rearrange AABCB to ABCBA which is a palindrome.
For sample 2, you can rearrange ABAB to ABBA which is a palindrome.
For sample 3, no matter how you rearrange, it will not be a palindrome.

(6) Given two strings A, B. Can you rearrange the letters of A to transform it into B ?

Sample Input	Sample Output
ABBC	
BABC	YES

(7) Given a large integer N (with at most 100 digits), for all its suffix, you have to find the remainder when divided by M.

For example, N = 12345, M = 6
Suffixes are:
12345
2345
345
45
5
For all suffixes, dividing by 6, we have to output the remainder:
3
5
3
3
5
You must solve it in O(# of digits of N) time complexity.

(8) Given N (<= 100) , find the number of digits of N! (factorial).

Sample Input	Sample Output
3	1
4	2
10	7
100	158

Note that you can not simply calculate N! and then find the number of digits, as N! may become very large and it is not possible to store them.
The idea of log() function may come handy in this case. Try reading some materials on log() function if you are stuck on this problem.

(9) Given N, print all the permutations of numbers 1, 2, …, N

Sample Input	Sample Output
3	
	1 2 3
	1 3 2

```
                                        2 1 3
                                        2 3 1
                                        3 1 2
                                        3 2 1
```

(10) Given N, print all non-empty subsets of S = {1, 2, 3, …., N}

```
Sample Input                            Sample Output
3
                                        1
                                        2
                                        3
                                        1 2
                                        1 3
                                        2 3
                                        1 2 3
```

(11) Can you print all the **distinct** permutations “HAPPINESS” ?

(12) Generate all words of 3 lowercase letters. For example - aaa, cat, dog, one are all 3 letter words. How many of them are there ?

(13) Given A and B, find A*B. But you can not use the multiplication operator. It is easy to do it in O(B). But we want to make it better than that. **The time complexity of your program must be O(logB)**.

Hint: Can you recall how we calculated base^{power} in O(log(power)) time complexity ?

```
Sample Input                            sample Output
10 12
                                        120
```

(14) Suppose on a distant planet far away from here, there are coins of only 7, 13, 19 units. If you look closely, you can never pay someone 16 units. But you can pay 27 units (using 2 coins of 7 and one coin of 13). Given the cost of a flying car is **C.** Can you pay the exact C amount ?

```
Sample Input                            Sample Output
27
                                        YES
                                        7, 7, 13

16
                                        NO
```

(15) Given a binary string $s(s_1s_2......s_n)$ of size n and q queries. In each query you will be given two integers l,r and you’ve to answer how many 1’s are there in the substring $s_ls_{l+1}......s_r$.

```
Sample Input                                Sample Output

    10
    0101101001
    5
    1 3                                         1
    2 7                                         4
    1 10                                        5
    2 5                                         3
    1 7                                         4
```

Explanation:

For the 1st query the substring is : 010 , so the answer is 1.

For the 4th query the substring is : 1011 , so the answer is 3.

You need to solve the problem in O(n+q) complexity.

(16) You'll be given an integer array $a(a_1a_2....a_n)$ of size n and q queries. In each query, you'll be given 3 integers l,r,x, which means you'll have to add x to each number in the range l to r. After all the queries you've to print the final array.

```
Sample Input                             Sample Output
    7
    2 1 0 3 -2 0 5
    3
    2 3 -2
    3 5 4
    1 6 10                                 12 9 12 17 12 10 5
```

Explanation:

After 1st query the array becomes, 2 -1 -2 3 -2 0 5

After 2nd query the array becomes, 2 -1 2 7 2 0 5

After 3rd query the array becomes, 12 9 12 17 12 10 5

You've to solve the problem in O(n+q) complexity.

(17) Given a string $s=(s_1s_2…..s_n)$ of length $n(n <= 10^5)$ consisting of lowercase english letters, for each character that appeared at least once, print the character and it's frequency.

```
Sample Input                                    Sample Output
    bcbbxyx
                                                  b = 3
                                                  c = 1
                                                  x = 2
                                                  y = 1
```

You've to solve it in O(n+alpha) complexity, where alpha = 26.

(18) There is a 2D grid, in each cell there is either a VIP(denoted by V) or a guard(denoted by G) or an ordinary person(denoted by O). A vip is called safe if he has at least one guard at the adjacent positions(above, below, left, right) to him. You have to count the number of safe vips.

```
Sample Input                                         Sample Output

5 5
VGOOV                                                    5
OVGVV
GGOOO
OOVVO
GVVVG
```

(19) You have an array of size $n(n<=10^5)$, in each move you can increase any element of the array by one. Find out the minimum number of moves you have to make in order to make the array increasing.(Each element is larger than it's previous one).

```
Sample Input                                       Sample Output

7
-2 -4 -1 3 4 0 8                                   9
```

Explanation: optimal solution here would be to increase the 2nd number(-4) by 3, increase the 3rd number (-1) by 1 and increase the 6th number(0) by 5.

```
5
2 4 7 8 10                                          0
```

You should solve it in O(n).

(20) You’re given a sorted array of size n(n<=10^5). Output the count of distinct numbers in it.

Sample Input	Sample Output
10	
2 2 5 5 7 8 9 10 11 11	7

You should solve it in O(n).