1. **Scenario:** You are developing a banking application that categorizes transactions based on the amount entered.
   Write logic to determine whether the amount is positive, negative, or zero.

   Logic-1:

   1. Get the input amount from user.

   2. If input amount is greater than the zero means its positive, lessthen that zero means negative, if it is equal to zero means zero.

2. **Scenario:** A digital locker requires users to enter a numerical passcode. As part of a security feature, the system checks the sum of the digits of the passcode.
   Write logic to compute the sum of the digits of a given number.

   Logic-2:

   1. Get the digits from user.
   2. Set the sum of the digits(security passcode).
   3. split it as a individual digits and sum()
   4. If user digits are equal to sum of the digits means ‘open’ otherwise ‘not open’

3. **Scenario:** A mobile payment app uses a simple checksum validation where reversing a transaction ID helps detect fraud.
   Write logic to take a number and return its reverse.
   Logic-3:

   Start with a number.

   Take the last digit using % 10.

   Add it to a new number (by shifting digits using * 10).

   Remove the last digit from the original number using // 10.

   Repeat until the original number becomes 0.

4. **Scenario:** In a secure login system, certain features are enabled only for users with prime-numbered user IDs.
   Write logic to check if a given number is prime.
   Logic-4:

   1. Get the user id(num) fro user:
   2. Num should be greater then one .becuse, it divisible by 1 itself.
   3. find the prime or not from range(2, square root of n + 1)
   4. If num % iterated value is equal to zero once in this range means , ‘its not a prime number’ otherwise 'prime number'

5. **Scenario:** A scientist is working on permutations and needs to calculate the factorial of numbers frequently.
   Write logic to find the factorial of a given number using recursion.

```
Logic-5:
        1. Get number from user.
        2. if number lessthen or equal to 1 means return 1(0,1 factorial 1)
        3. else return number*factorial(num-1)
```

6. **Scenario:** A unique lottery system assigns ticket numbers where only Armstrong numbers win the jackpot.
   Write logic to check whether a given number is an Armstrong number.

```
Logic-6:
        1. get the ticket number From user.
        2. Fix the amstrong number to check.
        3. Check whether a given number is equal to Armstrong number or not.
                if yes, return ‘you won the lottery’,
                else return ‘Better luck next time’
```

7. **Scenario:** A password manager needs to strengthen weak passwords by swapping the first and last characters of user-generated passwords.
   Write logic to perform this operation on a given string.

```
Logic-7:
        1. Get password from user.
        2. if password length less then 2(single letter), return 1
        3. otherwise, split it as a last index,middle part,first index. and concatenate all in
                required order.
```

8. **Scenario:** A low-level networking application requires decimal numbers to be converted into binary format before transmission.
   Write logic to convert a given decimal number into its binary equivalent.

```
Logic-8:
        1. Get the input From user.
        2. And typecast into binary.(bin()).
```

9. **Scenario:** A text-processing tool helps summarize articles by identifying the most significant words.
   Write logic to find the longest word in a sentence.

```
Logic-9:
        1. Get the input from user.
        2. get input as a 'sentence'.
        3. split the word in that sentence
        4. then use attribute max to find the longest word based on length.
```

10.**Scenario:** A plagiarism detection tool compares words from different documents and checks if they are anagrams (same characters but different order).

Write logic to check whether two given strings are anagrams.

Logic:-10

1. sorted(str): Sorts characters alphabetically.

2. If sorted versions of both strings are equal, they are anagrams.