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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

My Answers (Krithiksha):

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.

i. Get the amount
ii. Check if the amount is equal to 0, if yes , print ‘zero’
iii. Otherwise, check if amount is greater than 0, if yes , print ‘Positive’
iv. Otherwise, print ‘negative’

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.

i. Get the given number
ii. Initialize a variable sum_result=0
iii. Loop through the number
iv. And add the number with sum_result +=number
v. Return the sum_result

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.

i. Get the number
ii. Convert the number as string (as string have slicing operation , we can do the reserve using slicing)
iii. Reverse the string (string[::-1])
iv. Convert the string into integer
v. And return the result

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.

i. Get the number
ii. Condition 1:-
iii. Check the number is less than 2 , if yes , print ‘the number is not prime”
iv. Condition 2:-
v. If above condition fails, loop the num from 2 to square root of the number

vi. And check if the number is divisible by any of the num from 2 to square root of the number
vii. If yes , print 'the number is not prime"
viii. Otherwise, if the above 2 condition fails, print 'the number is prime"

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.

i. Create a function called factorial
ii. Put the base case , to check if number is equal to 0, return 1 --- this condition is used to stop the recursion
iii. Put the recursive case, to return factorial using the condition (number * factorial(number -1) --- this condition will recursively calls the function again and again and perform the factorial operation until it reaches the stop condition
iv. Call the factorial function by passing the given number

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.

i. Get the given number (153)
ii. Split the number (1 + 5+ 3)
iii. Cube the numbers (1 + 125 + 27)
iv. Add those numbers and store in calculated_arm variable(153)
v. If calculated_arm is equals given number , then given number is Armstrong number
vi. Otherwise it is not a Armstrong number

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.

i. Get the given_string
ii. Initialize a empty temporary variable 'temp' - to store the value while swapping 2 characters
iii. Assign 1st character in temp variable → temp = given_string[0]
iv. Update last character of the string to 1st character of the string → given_string[0] = given_string[-1]
v. Update first character of the string which is stored in temp variable to 1st character of the string → given_string[-1] = temp
vi. Return the swapped string . given_string

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.

i. Get the number
ii. To find the binary value of the decimal -> need to divide by base(2) of binary
iii. Get the Quotient and remainder by dividing givenNumber by 2
iv. Save the remainder
v. Store the Quotient in tempQuotient variable
vi. now get the Quotient and remainder by dividing tempQuotient by 2
vii. repeat the steps from 3 to 6, until you get the quotient as one (1)
viii. concatenate the remainder together and return it

eg:
Number: 10
10 mod 2 → quotient =5 remainder=0
5 mod 2 → quotient =2 remainder=1
2 mod 2 → quotient =1 remainder=0

Result : binary value of decimal 10 is 1010

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.

i. Get the sentence
ii. Loop through the sentence and find the length of each word in the sentence and
iii. Get the maximum length word
iv. Find the maximum length word in the sentence and
v. Return the word as result

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

i. Get the 2 strings
ii. Sort the 2 strings
iii. Check if the sorted 2 strings are equal or not, if yes, print 'strings are anagram'
iv. Otherwise, print 'strings are Not anagram'