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.

   a. Obtain the input amount
   b. Check for the number <0, if yes, declare it to be withdrawal
   c. Else if, number>0, then declare it to be deposit
   d. Else, it will be definitely be zero, then declare as error occurred.

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.
   a. Obtain the user input as save in variable(num)
   b. Initialize the sum variable to zero
   c. Use a while loop till n is not zero
      i. Obtain last digit(n=num/10) and store in temporary variable
      ii. Add the temp value to sum variable(sum+=n)
      iii. Do Floor division and remove the last digit(num//=10)
   d. Return the sum variable

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.
   a. Obtain the number n and declare a variable rev = 0
   b. Use a while loop till n is not zero
      1. Obtain last digit(n=num/10) and store in temporary variable
      2. Add the temp value to reversed variable (rev=rev*10+n)
      3. Do Floor division and remove the last digit (num//=10)
   c. Return the rev reversed variable

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.
   a. Obtain the user id
   b. Use a flag declared as false
   c. Use a for loop to iterate from 2 till the number-1.
   d. Inside the iteration check for divisibility (num%i==0)
   e. If the number is divisible, make the flag true
   f. After loop is complete, return prime if variable is false, else return not 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.
   a. Obtain the number and declare a fact=1 variable
   b. Use a for loop to iterate from 1 to the input number
   c. Multiply each number to the fact variable and save (fact*=i)
   d. Return the fact value after the iteration is complete.

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.
   a. Obtain the user input and declare a sum variable
   b. Obtain the number of digits in the number
   c. Undergo the iteration of each digit and exponentially calculate it with the number of digits (sum+=i**n)
   d. Return the sum of the exponential sum of digits after iteration

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.
   a. Obtain the user input and convert the string to list

   b. Use a temporary variable to swap the first and last elements
   c. Using the function join(), convert it into string and return

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.
   a. Obtain the user input
   b. Use a for loop to iterate till the number becomes zero or one
   c. Get the modulo answer by 2 and store the reminder in a list
   d. Convert the list into string and then to an integer and return

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.
   a. Obtain the user input as a string
   b. Whenever there is a white space in between, skip it and take the character(s) till it reaches a white space
   c. Store those words in a list
   d. Declare a max variable with “”
   e. Using len() function, check for the length and whichever is greater than the actual max words.
   f. Return the max word

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.
   a. Obtain the two strings
   b. Check the length of both the strings and verify
   c. Convert the strings to lists
   d. Use a membership operator and iterate throughout any one list
   e. If the membership fails in any character, return not anagrams
   f. If the iteration is complete, then return as anagram