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.

   - Ask the user to enter a transaction amount
     - Enter the transaction amount:
   - If the number is greater than 0, print "Positive (Deposit)".
   - Else if the number is less than 0, print "Negative (Withdrawal)".
   - Else, print "Zero (No Transaction)".

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.

   - Ask the user to enter a passcode
     - Enter your numerical passcode:
   - Initialize a variable to hold the sum
   - Iterate through each character in the passcode
   - Display the result
     print("Sum of digits:", digit_sum)

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.

   - Ask the user to enter a transaction ID (as a number)
     - Enter the transaction ID:
   - Convert the number into a string
   - Reverse the string
   - Convert it back to a number
   - Display the reversed number
     - print("Reversed transaction ID:", reversed_id)

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.

   - Ask the user to enter a user ID
     - Enter your user ID:
   - If the number is less than 2, print "Not Prime".
   - Loop from 2 to the square root of the number:
   - If the number is divisible by any of these values, print "Not Prime" and exit.
   - If no divisors are found,
     - print("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.

   - Ask the user to enter number
     - Enter a number:
   - If the number is 0 or 1, return 1.
   - Else, return the number multiplied by the factorial of (number - 1).
   - Print the result.

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.

   - Ask the user to enter number
     - Enter a number:
   - Convert the number to a string to access individual digits
   - Initialize a sum variable
   - For each digit in the number
     - Raise the digit to the power of the total number of digits and add to sum
   - Compare and print result
     - If armstrong_sum == number:
       - print("Armstrong Number")
     - If not print("Not 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.

    - Ask the user to enter passwords
        - Enter your password:
    - Check if the string is too short (less than 2 ) to swap
        - If less than 2, print("Password too short to swap characters.")
        - If not swap first and last characters
    - Print the modified 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.

    - Ask the user to enter number
        - Enter a decimal number:
    - Initialize an empty string for binary representation
    - While the number is greater than 0
        - Get remainder
        - Add to binary string
        - Update the number
    - Reverse the binary string
    - Print the binary representation
        - print("Binary equivalent:", binary)

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.

    - Ask the user to enter sentence
        - Enter a sentence:
    - Split the sentence into words
    - Initialize variables to track the longest word
    - Iterate through each word
    - The loop checks each word’s length and updates the longest one found
    - Print the longest word
        - print("Longest word:", longest_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.

- Ask the user to enter word
  - Enter the first word:
  - Enter the second word:
- Removes spaces (useful for multi-word phrases).
- Check if sorted characters match
  - if sorted list both are identical
    print("The strings are anagrams.")
  - if sorted list both are not identical
    print("The strings are not anagrams.")