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.
   Step:

```
amount = float(input("Enter transaction amount: "))
if amount > 0:
   print("Positive transaction")
elif amount < 0:
   print("Negative transaction")
else:
   print("Zero 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.
   Step:

```
num = int(input("Enter passcode: "))
sum_digits = sum(int(digit) for digit in str(abs(num)))
print("Sum of digits:", sum_digits)
```

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.
   Step:

```
num = int (input ("Enter transaction ID: "))
reversed_num = int(str(abs(num)) [::-1])
print("Reversed ID:", reversed_num)
```

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.
   num = int(input("Enter user ID: "))
   Step:

```
if num > 1:
   for i in range(2, int(num ** 0.5) + 1):
      if num % i == 0:
         print("Not a prime number")
         break
```

```
    else:
        print("Prime number")
else:
    print ("Not a 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.
   Step:

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    return n * factorial(n - 1)

num = int(input("Enter a number: "))
print("Factorial is:", factorial(num))
```

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.
   Step:

```
num = int(input("Enter a number: "))
num_str = str(num)
n = len(num_str)
armstrong_sum = sum(int(digit) ** n for digit in num_str)

if armstrong_sum == num:
    print("Armstrong number")
else:
    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.

   Step:

```
password = input("Enter password: ")
if len(password) > 1:
    swapped = password[-1] + password[1:-1] + password[0]
else:
    swapped = password
print("Strengthened password:", swapped)
```

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.

Step:

```
decimal = int(input("Enter a decimal number: "))
binary = bin(decimal)[2:]
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.

Step:

```
sentence = input("Enter a sentence: ")
words = sentence.split()
longest = max(words, key=len)
print("Longest word:", longest)
```

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.

Step:

```
str1 = input("Enter first word: ").lower().replace(" ", "")
str2 = input("Enter second word: "). lower(). replace(" ", "")
if sorted(str1) == sorted(str2):
   print("Anagrams")
else:
   print("Not anagrams")
```