### Complete Python Programs with Documentation

## 1. Sum of Two Numbers

num1 = int(input("Enter first number: "))  
num2 = int(input("Enter second number: "))  
print("Sum:", num1 + num2)

**Objective:** Calculate the sum of two integers.

**Approach:** Take two inputs, add using +, display result.

**Challenges:** Ensuring input is integer.

**Solution:** Used int(input()).

**Sample Input/Output:** Input: 5, 7 Output: 12

**Learning Outcome:** Learned arithmetic and input handling.

## 2. Odd or Even Checker

num = int(input("Enter a number: "))  
if num % 2 == 0:  
 print("Even")  
else:  
 print("Odd")

**Objective:** Check if a number is odd or even.

**Approach:** Use % 2 to find remainder.

**Challenges:** Negative numbers.

**Solution:** Conditional handles all integers.

**Sample Input/Output:** Input: 13 Output: Odd

**Learning Outcome:** Conditional logic and modulus usage.

## 3. Factorial Calculation

n = int(input("Enter a number: "))  
factorial = 1  
for i in range(1, n + 1):  
 factorial \*= i  
print("Factorial:", factorial)

**Objective:** Compute factorial of n.

**Approach:** Multiply numbers 1 to n in loop.

**Challenges:** Large numbers, zero input.

**Solution:** Loop multiplication, handle n=0.

**Sample Input/Output:** Input: 5 Output: 120

**Learning Outcome:** Loops and mathematical operations.

## 4. Fibonacci Sequence Generator

n = int(input("Enter number of terms: "))  
a, b = 0, 1  
fibonacci = []  
for i in range(n):  
 fibonacci.append(a)  
 a, b = b, a + b  
print("Fibonacci sequence:", fibonacci)

**Objective:** Generate first n Fibonacci numbers.

**Approach:** Loop, add previous two numbers.

**Challenges:** Handling n=0 or 1.

**Solution:** Initialize first two numbers, use list.

**Sample Input/Output:** Input: 7 Output: [0,1,1,2,3,5,8]

**Learning Outcome:** Sequence generation, lists.

## 5. Reverse a String

text = input("Enter a string: ")  
print("Reversed:", text[::-1])

**Objective:** Reverse string.

**Approach:** Use slicing [::-1].

**Challenges:** Empty string, special characters.

**Solution:** Slicing handles all cases.

**Sample Input/Output:** Input: Hello Output: olleH

**Learning Outcome:** String manipulation.

## 6. Palindrome Checker

text = input("Enter a string: ")  
print(text == text[::-1])

**Objective:** Check if string reads same forwards/backwards.

**Approach:** Compare string with reversed version.

**Challenges:** Case sensitivity and spaces.

**Solution:** Can use .lower() and remove spaces if needed.

**Sample Input/Output:** Input: radar Output: True

**Learning Outcome:** Logical reasoning and string comparison.

## 7. Leap Year Checker

year = int(input("Enter a year: "))  
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):  
 print(True)  
else:  
 print(False)

**Objective:** Check leap year.

**Approach:** Conditional logic based on divisibility.

**Challenges:** Correct leap year rules.

**Solution:** Combined conditions.

**Sample Input/Output:** Input: 2024 Output: True

**Learning Outcome:** Conditionals and logical operators.

## 8. Armstrong Number Checker

num = int(input("Enter a number: "))  
digits = str(num)  
power = len(digits)  
total = sum(int(d)\*\*power for d in digits)  
print(num == total)

**Objective:** Check Armstrong number.

**Approach:** Sum digits raised to power of number of digits.

**Challenges:** Varying number of digits.

**Solution:** Used string conversion and comprehension.

**Sample Input/Output:** Input: 153 Output: True

**Learning Outcome:** String to integer conversion, loops.

## 9. Custom Encryption-Decryption System

def encrypt(text, key):  
 result = ""  
 for char in text:  
 if char.isalpha():  
 shift = key % 26  
 if char.isupper():  
 result += chr((ord(char)-ord('A')+shift)%26 + ord('A'))  
 else:  
 result += chr((ord(char)-ord('a')+shift)%26 + ord('a'))  
 else:  
 result += char  
 return result  
  
def decrypt(cipher\_text, key):  
 return encrypt(cipher\_text, -key)  
  
message = input("Enter message: ")  
key = int(input("Enter key: "))  
encrypted = encrypt(message, key)  
print("Encrypted:", encrypted)  
decrypted = decrypt(encrypted, key)  
print("Decrypted:", decrypted)

**Objective:** Encrypt and decrypt messages using a custom algorithm.

**Approach:** Caesar cipher, shift letters by key, preserve non-letters.

**Challenges:** Uppercase/lowercase handling, symbols.

**Solution:** ASCII and modulo arithmetic, conditional checks.

**Sample Input/Output:** Input: Hello World!, Key: 4 Encrypted: Lipps Asvph! Decrypted: Hello World!

**Learning Outcome:** Encryption principles, string manipulation, modular arithmetic.