1. Nested Function for Sum

```
def outer_function(a, b):
    def inner_function():
        return a + b
    return inner_function() + 5

print(outer_function(3, 4))  # Output: 12
```

---

2. Largest of Three Numbers Using Helper Function

```
def max_of_two(x, y):
    return x if x > y else y

def max_of_three(a, b, c):
    return max_of_two(max_of_two(a, b), c)

# Example
print(max_of_three(10, 25, 18))  # Output: 25
```

---

3. Sum and Product Using args

```
def sum_of_numbers(*args):
    return sum(args)
def multiply_numbers(*args):
    result = 1
```

```
    for num in args:
        result *= num
    return result

# Example
print(sum_of_numbers(1, 2, 3, 4))  # Output: 10
print(multiply_numbers(1, 2, 3, 4))  # Output: 24
```

---

4. Fibonacci Series (Recursive)

```
def fibonacci(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        series = fibonacci(n - 1)
        series.append(series[-1] + series[-2])
        return series

# Example
print(fibonacci(10))  # Output: Fibonacci series up to 10 terms
```

---

5. Menu-Driven Factorial and Fibonacci (Non-Recursive)

```
def factorial(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

def fibonacci_non_recursive(n):
    a, b = 0, 1
    fib_series = []
    for _ in range(n):
        fib_series.append(a)
        a, b = b, a + b
    return fib_series
choice = int(input("Enter 1 for Factorial, 2 for Fibonacci: "))
n = int(input("Enter a number: "))

if choice == 1:
    print("Factorial:", factorial(n))
elif choice == 2:
    print("Fibonacci Series:", fibonacci_non_recursive(n))
else:
    print("Invalid choice")
```

---

6. Even/Odd or Prime Checker

```
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True

num = int(input("Enter a number: "))
print("Even" if num % 2 == 0 else "Odd")
print("Prime" if is_prime(num) else "Not Prime")
```

---

7. Reverse Number and String & Palindrome Check

```
def reverse_number(n):
    rev = 0
    temp = n
    while temp > 0:
        rev = rev * 10 + temp % 10
        temp //= 10
    return rev

def reverse_string(s):
    return s[::-1]
```

```
num = int(input("Enter a number: "))
print("Reversed Number:", reverse_number(num))
print("Palindrome:", num == reverse_number(num))

string = input("Enter a string: ")
print("Reversed String:", reverse_string(string))
```

---

8. Patterns

```
def pattern_1():
    for i in range(4, 0, -1):
        print("* " * i)

def pattern_2():
    for i in range(1, 5):
        print(" ".join("*" * i))

pattern_1()
print()
pattern_2()
```

---

9. Dictionary with Roll Numbers and Marks

```
n = int(input("Enter number of students: "))
students = {}
```

```
for _ in range(n):
    roll = input("Enter roll number: ")
    marks = int(input("Enter marks: "))
    students[roll] = marks

print("Student Dictionary:", students)
```

---

10. Count Uppercase, Lowercase, Digits, and Special Characters

```
def count_characters(s):
    upper = lower = digits = special = 0
    for char in s:
        if char.isupper():
            upper += 1
        elif char.islower():
            lower += 1
        elif char.isdigit():
            digits += 1
        else:
            special += 1
    return upper, lower, digits, special

text = input("Enter a string: ")
upper, lower, digits, special = count_characters(text)
```

```
print(f"Uppercase: {upper}, Lowercase: {lower}, Digits: {digits}, Special Characters: {special}")
```

---

11. List Methods

```
lst = []
lst.append(10)
lst.append(20)
lst.insert(1, 15)
lst.remove(20)
lst.sort()
print("Sorted List:", lst)
print("Index of 15:", lst.index(15))
```

---

12. NumPy Array Operations

```
import numpy as np
arr = np.arange(12)
print("Shape of Array:", arr.shape)

matrix = arr.reshape(3, 4)
print("3x4 Matrix:\n", matrix)

flattened = matrix.flatten()
print("Flattened Array:", flattened)
```

```
sliced = arr[::2]
print("Every alternate element:", sliced)
```

---

13. Regular Expressions for Email & Date

```
import re
email_pattern = r'^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$'
date_pattern = r'^(0[1-9]|[12][0-9]|3[01])/(0[1-9]|1[0-2])/\d{4}$|^(0[1-9]|1[0-2])-(0[1-9]|[12][0-9]|3[01])-\d{4}$'

email = input("Enter email: ")
print("Valid Email" if re.match(email_pattern, email) else "Invalid Email")

date = input("Enter date (DD/MM/YYYY or MM-DD-YYYY): ")
print("Valid Date" if re.match(date_pattern, date) else "Invalid Date")
```

---

14. Regular Expressions for URL & Phone Number

```
url_pattern = r'^(http://|https://)[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}(/.*)?$'
phone_pattern = r'^\d{3}-?\d{3}-?\d{4}$'

url = input("Enter URL: ")
print("Valid URL" if re.match(url_pattern, url) else "Invalid URL")
phone = input("Enter phone number: ")
print("Valid Phone Number" if re.match(phone_pattern, phone) else "Invalid Phone Number")
```