1.

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

    return inner_function() + 5

# Example usage
result = outer_function(3, 7)
print(result)  # Output: 15
```

2.

```
def max_of_two(a, b):
    """Helper function to return the larger of two numbers."""
    return a if a > b else b

def max_of_three(a, b, c):
    """Main function to return the largest of three numbers using the helper function."""
    return max_of_two(a, max_of_two(b, c))

# Example usage
num1, num2, num3 = 10, 25, 15
print("Largest number:", max_of_three(num1, num2, num3))
```

3.

```
def sum_of_numbers(*args):
    return sum(args)

def product_of_numbers(*args):
    product = 1
    for num in args:
        product *= num
    return product

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

4.

```
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 usage:
n_terms = 10  # Change this value as needed
print(fibonacci(n_terms))
```

5.

```
def factorial(n):
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

def fibonacci(n):
    fib_series = []
    a, b = 0, 1
    for _ in range(n):
        fib_series.append(a)
        a, b = b, a + b
    return fib_series

def main():
    while True:
        print("\nChoose an option:")
        print("1. Compute Factorial")
        print("2. Print Fibonacci Series")
        print("3. Exit")
```

```
        choice = input("Enter your choice (1/2/3): ")

        if choice == '1':
            num = int(input("Enter a number: "))
            if num < 0:
                print("Factorial is not defined for negative numbers.")
            else:
                print(f"Factorial of {num} is {factorial(num)}")

        elif choice == '2':
            num = int(input("Enter the number of terms: "))
            if num < 0:
                print("Number of terms cannot be negative.")
            else:
                print(f"Fibonacci series: {fibonacci(num)}")

        elif choice == '3':
            print("Exiting the program.")
            break

        else:
            print("Invalid choice! Please enter 1, 2, or 3.")

if __name__ == "__main__":
    main()
```

6.

```
def is_even_or_odd(number):
    return "Even" if number % 2 == 0 else "Odd"

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

def main():
    while True:
        print("\nMenu:")
        print("1. Check if a number is Even or Odd")
        print("2. Check if a number is Prime")
        print("3. Exit")

        choice = input("Enter your choice (1-3): ")

        if choice == "1":
            num = int(input("Enter a number: "))
            print(f"The number {num} is {is_even_or_odd(num)}.")
        elif choice == "2":
            num = int(input("Enter a number: "))
```

```
            print(f"The number {num} is {'Prime' if is_prime(num) else 'Not Prime'}.")
        elif choice == "3":
            print("Exiting program. Goodbye!")
            break
        else:
            print("Invalid choice. Please enter a valid option.")

if __name__ == "__main__":
    main()
```

7.

```
def reverse_number(n):
    reversed_num = 0
    original = n  # Store the original number for palindrome check
    while n > 0:
        digit = n % 10  # Get last digit
        reversed_num = reversed_num * 10 + digit  # Append digit to reversed number
        n //= 10  # Remove last digit
    return reversed_num

def is_palindrome_number(n):
    return n == reverse_number(n)

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

```
def main():
    choice = input("Enter 'number' to reverse a number or 'string' to reverse a string: ").strip().lower()
    if choice == 'number':
        num = int(input("Enter a number: "))
        reversed_num = reverse_number(num)
        print(f"Reversed Number: {reversed_num}")
        if is_palindrome_number(num):
            print("The number is a palindrome.")
        else:
            print("The number is not a palindrome.")
    elif choice == 'string':
        text = input("Enter a string: ")
        print(f"Reversed String: {reverse_string(text)}")
    else:
        print("Invalid choice!")

if __name__ == "__main__":
    main()
```

8.

```
def pattern1(rows=4):
    for i in range(rows, 0, -1):
        print('* ' * i)

def pattern2(rows=4):
    for i in range(1, rows + 1):
        print('* ' * i)
```

```
def main():
    while True:
        print("\nMenu:")
        print("1. Pattern 1")
        print("2. Pattern 2")
        print("3. Exit")
        choice = input("Enter your choice (1-3): ")

        if choice == '1':
            pattern1()
        elif choice == '2':
            pattern2()
        elif choice == '3':
            print("Exiting...")
            break
        else:
            print("Invalid choice! Please enter a valid option.")

if __name__ == "__main__":
    main()
```

9.

```
# Program to store roll numbers and marks of students in a dictionary

# Read the number of students
n = int(input("Enter the number of students: "))

# Initialize an empty dictionary
student_dict = {}

# Read roll number and marks for each student
for i in range(n):
    roll_number = input(f"Enter roll number for student {i+1}: ")
    marks = float(input(f"Enter marks for student {i+1}: "))
    student_dict[roll_number] = marks

# Display the dictionary
print("\nStudent Dictionary:")
print(student_dict)
```

10.

```
def count_characters(s):
    uppercase = sum(1 for char in s if char.isupper())
    lowercase = sum(1 for char in s if char.islower())
    digits = sum(1 for char in s if char.isdigit())
    special = sum(1 for char in s if not char.isalnum())
```

```
    return uppercase, lowercase, digits, special

# Take user input
input_string = input("Enter a string: ")
uppercase, lowercase, digits, special = count_characters(input_string)

# Display results
print(f"Uppercase Letters: {uppercase}")
print(f"Lowercase Letters: {lowercase}")
print(f"Digits: {digits}")
print(f"Special Characters: {special}")
```

11.

```
# Create an empty list
my_list = []

# Append elements to the list
n = int(input("How many elements do you want to add? "))
for _ in range(n):
    element = input("Enter an element: ")
    my_list.append(element)

print("List after appending elements:", my_list)

# Insert an element at a specific position
pos = int(input("Enter the position to insert an element: "))
```

```
element = input("Enter the element to insert: ")
my_list.insert(pos, element)
print("List after insertion:", my_list)

# Remove a specific element from the list
rem_element = input("Enter the element to remove: ")
if rem_element in my_list:
    my_list.remove(rem_element)
    print("List after removal:", my_list)
else:
    print("Element not found in the list.")

# Sort the list in ascending order
my_list.sort()
print("List after sorting:", my_list)

# Display the index of an element
search_element = input("Enter an element to find its index: ")
if search_element in my_list:
    print(f"Index of {search_element}: {my_list.index(search_element)}")
else:
    print("Element not found in the list.")
```

12.

```
import numpy as np

# i. Create and check the shape of an array
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
print("Original Array:", arr)
print("Shape of array:", arr.shape)

# ii. Convert a 1D array of 12 elements into a 3x4 matrix
matrix = arr.reshape(3, 4)
print("\n3x4 Matrix:\n", matrix)

# iii. Convert a 2D array into a 1D array
flattened = matrix.flatten()
print("\nFlattened Array:", flattened)

# iv. Extract a subarray using slicing (e.g., first two rows and first three columns)
subarray = matrix[:2, :3]
print("\nExtracted Subarray:\n", subarray)

# v. Extract every alternate element from the original array
alternate_elements = arr[::2]
print("\nEvery alternate element:", alternate_elements)
```