1. Write a Python program to calculate the area of a rectangle given its length and width

```
def calculate_rectangle_area(length, width):
    area = length * width
    return area

# Input
length = float(input("Enter the length of the rectangle: "))
width = float(input("Enter the width of the rectangle: "))

# Calculate area
rectangle_area = calculate_rectangle_area(length, width)

# Output
print(f"The area of the rectangle with length {length} and width {width} is: {rectangle_area}")
```

2. Write a program to convert miles to kilometers

```
def miles_to_kilometers(miles):
    kilometers = miles * 1.60934
    return kilometers

# Input
miles = float(input("Enter the distance in miles: "))

# Convert to kilometers
kilometers = miles_to_kilometers(miles)

# Output
print(f"{miles} miles is equal to {kilometers} kilometers.")
```

3. Write a function to check if a given string is a palindrome.

```
def is_palindrome(s):
    # Remove spaces and convert to lowercase for case-insensitive comparison
    s = s.replace(" ", "").lower()
```

```
    # Check if the reversed string is equal to the original string
    return s == s[::-1]

# Input
user_input = input("Enter a string to check if it's a palindrome: ")

# Check if the input is a palindrome
if is_palindrome(user_input):
    print(f"{user_input} is a palindrome.")
else:
    print(f"{user_input} is not a palindrome.")
```

4. Write a Python program to find the second largest element in a list.

```
def second_largest_element(nums):
    if len(nums) < 2:
        return "List should have at least two elements."

    # Find the maximum element in the list
    max_element = max(nums)

    # Remove the maximum element from the list
    nums.remove(max_element)

    # Find the second maximum element in the updated list
    second_largest = max(nums)

    return second_largest

# Input
try:
    num_list = [int(x) for x in input("Enter a list of numbers separated by spaces: ").split()]
    result = second_largest_element(num_list)
```

```
    print(f"The second largest element in the list is: {result}")
except ValueError:
    print("Invalid input. Please enter valid numbers.")
```

5. Explain what indentation means in Python

```
if x > 0:
    print("x is positive")
    square = x * x
    print("The square of x is:", square)
```

6. Write a program to perform set difference operation

```
def set_difference_operation(set1, set2):
    # Using the - operator
    difference_result_operator = set1 - set2

    # Using the difference() method
    difference_result_method = set1.difference(set2)

    return difference_result_operator, difference_result_method

# Input
set1 = set(input("Enter elements for the first set separated by spaces: ").split())
set2 = set(input("Enter elements for the second set separated by spaces: ").split())

# Perform set difference operation
result_operator, result_method = set_difference_operation(set1, set2)

# Output
print(f"Set difference using - operator: {result_operator}")
print(f"Set difference using difference() method: {result_method}")
```

7. Write a Python program to print numbers from 1 to 10 using a while loop.

```
# Initialize a variable
number = 1
```

```
# Use a while loop to print numbers from 1 to 10
while number <= 10:
    print(number)
    number += 1
```

8. Write a program to calculate the factorial of a number using a while loop

```
# Function to calculate factorial
def calculate_factorial(number):
    factorial = 1

    # Use a while loop to calculate factorial
    while number > 0:
        factorial *= number
        number -= 1

    return factorial

# Input
try:
    num = int(input("Enter a number to calculate its factorial: "))

    # Check if the input is non-negative
    if num < 0:
        print("Factorial is not defined for negative numbers.")
    else:
        # Calculate and print the factorial
        result = calculate_factorial(num)
        print(f"The factorial of {num} is: {result}")

except ValueError:
    print("Invalid input. Please enter a valid integer.")
```

9. Write a Python program to check if a number is positive, negative, or zero using if-elif-else statements

```
# Input
```

```
try:
    number = float(input("Enter a number: "))

    # Check if the number is positive, negative, or zero
    if number > 0:
        print("The entered number is positive.")
    elif number < 0:
        print("The entered number is negative.")
    else:
        print("The entered number is zero.")

except ValueError:
    print("Invalid input. Please enter a valid number.")
```

10. Write a program to determine the largest among three numbers using conditional statements

```
# Input
try:
    num1 = float(input("Enter the first number: "))
    num2 = float(input("Enter the second number: "))
    num3 = float(input("Enter the third number: "))

    # Determine the largest among three numbers
    if num1 >= num2 and num1 >= num3:
        largest = num1
    elif num2 >= num1 and num2 >= num3:
        largest = num2
    else:
        largest = num3

    # Output
    print(f"The largest number among {num1}, {num2}, and {num3} is: {largest}")

except ValueError:
```

```
    print("Invalid input. Please enter valid numbers.")
```

11. Write a Python program to create a numpy array filled with ones of given shape.

```
import numpy as np

def create_ones_array(shape):
    ones_array = np.ones(shape)
    return ones_array

# Input
try:
    rows = int(input("Enter the number of rows: "))
    cols = int(input("Enter the number of columns: "))

    # Create a numpy array filled with ones
    ones_array = create_ones_array((rows, cols))

    # Output
    print(f"Numpy array filled with ones of shape {ones_array.shape}:\n{ones_array}")

except ValueError:
    print("Invalid input. Please enter valid integers for the number of rows and columns.")
```

12. Write a program to create a 2D numpy array initialized with random integers.

```
import numpy as np

def create_random_array(rows, cols, low, high):
    random_array = np.random.randint(low, high, size=(rows, cols))
    return random_array

# Input
try:
    rows = int(input("Enter the number of rows: "))
    cols = int(input("Enter the number of columns: "))
```

```
    low_limit = int(input("Enter the lower limit for random integers: "))
    high_limit = int(input("Enter the upper limit for random integers: "))

    # Create a 2D numpy array initialized with random integers
    random_2d_array = create_random_array(rows, cols, low_limit, high_limit)

    # Output
    print(f"2D numpy array initialized with random integers:\n{random_2d_array}")

except ValueError:
    print("Invalid input. Please enter valid integers for rows, columns, lower limit, and upper limit.")
```

13. Write a Python program to generate an array of evenly spaced numbers over a specified range using linspace

```
import numpy as np

def generate_linspace_array(start, stop, num_points):
    linspace_array = np.linspace(start, stop, num_points)
    return linspace_array

# Input
try:
    start_value = float(input("Enter the start value: "))
    stop_value = float(input("Enter the stop value: "))
    num_points = int(input("Enter the number of points: "))

    # Generate an array of evenly spaced numbers using linspace
    linspace_result = generate_linspace_array(start_value, stop_value, num_points)

    # Output
    print(f"Array of {num_points} evenly spaced numbers from {start_value} to {stop_value}:\n{linspace_result}")

except ValueError:
    print("Invalid input. Please enter valid numerical values for start value, stop value, and number of points.")
```

14. Write a program to generate an array of 10 equally spaced values between 1 and 100 using linspace

```
import numpy as np

# Generate an array of 10 equally spaced values between 1 and 100 using linspace
result_array = np.linspace(1, 100, 10)

# Output
print(f"Array of 10 equally spaced values between 1 and 100:\n{result_array}")
```

15. Write a Python program to create an array containing even numbers from 2 to 20 using arange

```
import numpy as np

# Create an array containing even numbers from 2 to 20 using arange
result_array = np.arange(2, 21, 2)

# Output
print(f"Array containing even numbers from 2 to 20:\n{result_array}")
```

16. Write a program to create an array containing numbers from 1 to 10 with a step size of 0.5 using arange

```
import numpy as np

# Create an array containing numbers from 1 to 10 with a step size of 0.5 using arange
result_array = np.arange(1, 10.5, 0.5)

# Output
print(f"Array containing numbers from 1 to 10 with a step size of 0.5:\n{result_array}")
```