In [7]:
```
#Simple Hello World Program
print("Hello World")
```

```
Hello World
```

In [8]:
```
# Write programs to read input from the user and display output using input() and print().

num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
result = num1 + num2
print(f"The sum of {num1} and {num2} is {result}.")
```

```
The sum of 3.0 and 7.0 is 10.0.
```

In [9]:
```
import math

radius = float(input("Enter the radius of the circle: "))

area = math.pi * (radius ** 2)

print(f"The area of the circle with radius {radius} is {area:.2f}.")
```

```
The area of the circle with radius 4.35 is 59.45.
```

In [10]:
```
user_string = input("Enter a string to reverse: ")

reversed_string = user_string[::-1]

print(f"The reversed string is: {reversed_string}")
```

```
The reversed string is: pihsnretnI
```

In [11]:
```
#Implement tasks that use different data types (int, float,string, list, tuple, dict, set)

int1 = 10
int2 = 5

float1 = 10.5
float2 = 2.5

sum_result = int1 + float1
diff_result = int1 - float2
prod_result = int2 * float1
quot_result = float1 / int2

# Display the results
print("Integer addition:", sum_result)
print("Integer and float subtraction:", diff_result)
print("Integer and float multiplication:", prod_result)
print("Float division:", quot_result)
```

```
Integer addition: 20.5
Integer and float subtraction: 7.5
Integer and float multiplication: 52.5
Float division: 2.1
```

In [12]:
```
string1 = "Hello"
string2 = "World"

# Concatenation
concat_result = string1 + " " + string2

# String length
length = len(concat_result)

# Upper and lower case conversion
upper_case = concat_result.upper()
lower_case = concat_result.lower()

# Display the results
print("Concatenated string:", concat_result)
print("Length of the string:", length)
print("Uppercase string:", upper_case)
print("Lowercase string:", lower_case)
```

```
Concatenated string: Hello World
Length of the string: 11
Uppercase string: HELLO WORLD
Lowercase string: hello world
```

In [13]:
```
int_list = [1, 2, 3, 4, 5]

# Adding an element to the list
int_list.append(6)

# Removing an element from the list
int_list.remove(3)

# List slicing
sub_list = int_list[1:4]

# Display the results
print("Original list:", int_list)
print("Sub list:", sub_list)
```

```
Original list: [1, 2, 4, 5, 6]
Sub list: [2, 4, 5]
```

In [14]:
```
string_tuple = ("apple", "banana", "cherry")

# Accessing elements
first_fruit = string_tuple[0]
last_fruit = string_tuple[-1]

# Length of the tuple
tuple_length = len(string_tuple)

# Display the results
print("First fruit:", first_fruit)
print("Last fruit:", last_fruit)
print("Length of tuple:", tuple_length)
```

```
First fruit: apple
Last fruit: cherry
Length of tuple: 3
```

In [15]:
```
fruit_prices = {
    "apple": 3,
    "banana": 1,
    "cherry": 2
}

# Adding a new key-value pair
fruit_prices["date"] = 4

# Updating a value
fruit_prices["apple"] = 5

# Removing a key-value pair
del fruit_prices["banana"]

# Display the results
print("Fruit prices:", fruit_prices)
```

```
Fruit prices: {'apple': 5, 'cherry': 2, 'date': 4}
```

In [16]:
```
# Set of integers
int_set = {1, 2, 3, 4, 5}

# Adding an element to the set
int_set.add(6)

# Removing an element from the set
int_set.remove(3)

# Union and intersection with another set
another_set = {4, 5, 6, 7, 8}
union_set = int_set.union(another_set)
intersection_set = int_set.intersection(another_set)

# Display the results
print("Original set:", int_set)
print("Union of sets:", union_set)
print("Intersection of sets:", intersection_set)
```

```
Original set: {1, 2, 4, 5, 6}
Union of sets: {1, 2, 4, 5, 6, 7, 8}
Intersection of sets: {4, 5, 6}
```

In [17]:
```
# Mixed data types
mixed_list = [1, 2.5, "hello", [3, 4], (5, 6), {"key": "value"}, {7, 8, 9}]

# Access and display each element with its type
for element in mixed_list:
    print(f"Element: {element}, Type: {type(element)}")
```

```
Element: 1, Type: <class 'int'>
Element: 2.5, Type: <class 'float'>
Element: hello, Type: <class 'str'>
Element: [3, 4], Type: <class 'list'>
Element: (5, 6), Type: <class 'tuple'>
Element: {'key': 'value'}, Type: <class 'dict'>
Element: {8, 9, 7}, Type: <class 'set'>
```

In [18]:
```
#Create programs using if-else statements, loops (for, while), and try-except for error handling


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

if number > 0:
    print(f"The number {number} is positive.")
elif number < 0:
    print(f"The number {number} is negative.")
else:
    print(f"The number is zero.")
```

```
The number 5.0 is positive.
```

In [19]:
```
number = int(input("Enter a number to calculate its factorial: "))

factorial = 1

for i in range(1, number + 1):
    factorial *= i

# Display the result
print(f"The factorial of {number} is {factorial}.")
```

```
The factorial of 9 is 362880.
```

In [20]:
```
a, b = 0, 1
count = 0

# Define the number of Fibonacci numbers to generate
n = 10

# Use a while loop to generate Fibonacci numbers
print("The first 10 Fibonacci numbers are:")
while count < n:
    print(a)
    a, b = b, a + b
    count += 1
```

```
The first 10 Fibonacci numbers are:
0
1
1
2
3
5
8
13
21
34
```

In [21]:
```
try:
    numerator = float(input("Enter the numerator: "))
    denominator = float(input("Enter the denominator: "))

    # Perform the division
    result = numerator / denominator

    # Display the result
    print(f"The result of {numerator} divided by {denominator} is {result}.")

except ZeroDivisionError:
    # Handle division by zero error
    print("Error: Cannot divide by zero!")

except ValueError:
    # Handle invalid input error
    print("Error: Invalid input. Please enter numeric values.")
```

```
Error: Cannot divide by zero!
```

In [22]:
```
numbers = []

try:
    n = int(input("Enter the number of elements: "))

    for i in range(n):
        try:
            number = float(input(f"Enter number {i + 1}: "))
            numbers.append(number)
        except ValueError:
            print("Error: Invalid input. Please enter a numeric value.")
            break

    else:
        # Calculate the sum of the numbers
        total = sum(numbers)

        # Use if-else to display appropriate messages
        if total > 0:
            print(f"The sum of the numbers is {total}, which is positive.")
        elif total < 0:
            print(f"The sum of the numbers is {total}, which is negative.")
        else:
            print(f"The sum of the numbers is zero.")
```

```
except ValueError:
    print("Error: Invalid input. Please enter an integer value.")
```

```
The sum of the numbers is 35.0, which is positive.
```