In [ ]:
```
Introduction to User-Defined Functions and Python Classes
User-Defined Functions
User-defined functions are blocks of reusable code in Python that perform a specific task. They allow you to break down a large program into smaller, manageable pieces, improving readability, reusability, and maintainability.
```

In [ ]:
```
features of User-Defined Functions:
Abstraction: They hide the implementation details, allowing you to focus on the functionality.
Modularity: Functions promote code modularization, making it easier to debug and maintain.
Reusability: Once defined, functions can be reused multiple times throughout your program.
Parameterization: Functions can accept parameters, enabling customization and flexibility.
Return Values: They can return data back to the caller, facilitating communication between different parts of the program.
```

In [ ]:
```
Python is an object-oriented programming (OOP) language, and classes are a fundamental concept in OOP. A class is a blueprint for creating objects (instances) that share similar attributes and behaviors.

Key Concepts of Python Classes:
Attributes: Data stored within a class instance.
Methods: Functions defined within a class.
Inheritance: Ability to create a new class based on an existing class.
Encapsulation: Binding data and functions that operate on the data into a single unit (class).
Polymorphism: Ability to use a single interface for different data types or objects.
```

In [2]:
```python
import math

def calculate_area_circle(radius):
    area = math.pi * radius**2
    return area
radius = 5
area = calculate_area_circle(radius)
print("Area of the circle:", area)
```
```
Area of the circle: 78.53981633974483
```

In [3]:
```python
def is_prime(number):

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

num = 17
if is_prime(num):
    print(num, "is prime.")
else:
    print(num, "is not prime.")
```
```
17 is prime.
```

In [4]:
```python
def reverse_string(string):
    return string[::-1]


original_str = "hello"
reversed_str = reverse_string(original_str)
print("Reversed string:", reversed_str)
```
```
Reversed string: olleh
```

In [5]:
```python
def factorial(n):
    """Calculate the factorial of a number."""
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

# Example usage:
num = 5
fact = factorial(num)
print("Factorial of", num, "is", fact)
```
```
Factorial of 5 is 120
```

In [6]:
```python
def fibonacci(n):

    sequence = [0, 1]
    while len(sequence) < n:
        next_num = sequence[-1] + sequence[-2]
        sequence.append(next_num)
    return sequence

# Example usage:
terms = 10
fib_sequence = fibonacci(terms)
print("Fibonacci sequence:", fib_sequence)
```
```
Fibonacci sequence: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

In [9]:
```python
def add(x, y):

    return x + y

result = add(3, 5)
print(result)
```
```
8
```

In [10]:
```python
def subtract(x, y):

    return x - y

# Example usage:
result = subtract(10, 4)  # Result:
print(result)
```
```
6
```

In [11]:
```python
def absolute_value(x):
    if x >= 0:
        return x
    else:
        return -x

result = absolute_value(-9)  # Result: 9
print(result)
```
```
9
```

In [12]:
```python
def power(x, n):
    """Raise x to the power of n."""
    return x ** n


result = power(2, 3)
print(result)
```
```
8
```

In [13]:
```python
def area_rectangle(length, width):
    return length * width


result = area_rectangle(5, 4)  # Result: 20
print(result)
```
```
20
```

In [14]:
```python
def perimeter_square(side):
    """Calculate the perimeter of a square."""
    return 4 * side
result = perimeter_square(5)
print(result)
```
```
20
```

In [16]:
```python
def hypotenuse(a, b):
    result = hypotenuse(3, 4)
print(result)
```
```
20
```

In [17]:
```python
def celsius_to_fahrenheit(celsius):

    return (celsius * 9/5) + 32
result = celsius_to_fahrenheit(25)  # Result: 77.0
print(result)
```
```
77.0
```

In [18]:
```python
def fahrenheit_to_celsius(fahrenheit):

    return (fahrenheit - 32) * 5/9

result = fahrenheit_to_celsius(77)  # Result: 25.0
print(result)
```
```
25.0
```

In [19]:
```python
def minimum(x, y):

    return x if x < y else y
```

```python
result = minimum(7, 12)  # Result: 7
print(result)
```

```
7
```

In [ ]: