

Chapter 3 Assignment : Writing Statements

Target for this chapter

1. Recognize what an expression is and how Python evaluates it.
2. Use different operators to perform calculations and comparisons.
3. Explain what a statement is in Python.
4. Write different types of statements to make your programs do useful work.

Instructions for Students

1. Open src/assignment.py and implement the functions marked with **TODO**.
2. Do not modify the test file.
3. Run pytest in your terminal to check if your functions pass all tests.
4. You need to make all tests pass.

1. Arithmetic Operations

Source Codes

```
def calculate_sum(a, b):
    """
    Returns the sum of two numbers.
    Example:
        calculate_sum(2, 3) -> 5
    """
    # TODO: Implement the function
    return 0

def calculate_expression(a, b, c):
    """
    Returns the result of the expression a + b * c
    Example:
        calculate_expression(2, 3, 4) -> 14
    """
    # TODO: Implement the function
    return 0
```

Test Cases

```
def test_calculate_sum():
    assert calculate_sum(2, 3) == 5
    assert calculate_sum(-1, 5) == 4

def test_calculate_expression():
    assert calculate_expression(2, 3, 4) == 14
    assert calculate_expression(1, 2, 3) == 7
```

2. Comparison Operations

Source Codes

```
def is_greater(a, b):  
    """  
    Returns True if a > b else False  
    Example:  
        is_greater(5, 3) -> True  
    """  
    # TODO: Implement the function  
    return False  
  
def is_equal(a, b):  
    """  
    Returns True if a == b else False  
    """  
    # TODO: Implement the function  
    return False
```

Test Cases

```
def is_greater(a, b):  
    """  
    Returns True if a > b else False  
    Example:  
        is_greater(5, 3) -> True  
    """  
    # TODO: Implement the function  
    return False  
  
def is_equal(a, b):  
    """  
    Returns True if a == b else False  
    """  
    # TODO: Implement the function  
    return False
```

3. Conditional Statements

Source Codes

```
def max_of_two(a, b):  
    """  
    Returns the greater of a and b using if-else statement  
    """
```

```
# TODO: Implement the function
return a

def categorize_number(n):
    """
    Returns:
        'positive' if n > 0
        'negative' if n < 0
        'zero' if n == 0
    """
    # TODO: Implement the function
    return ""
```

Test Cases

```
def max_of_two(a, b):
    """
    Returns the greater of a and b using if-else statement
    """
    # TODO: Implement the function
    return a

def categorize_number(n):
    """
    Returns:
        'positive' if n > 0
        'negative' if n < 0
        'zero' if n == 0
    """
    # TODO: Implement the function
    return ""
```

4. Loop Statements

Source Codes

```
def sum_n_numbers(n):
    """
    Returns sum of numbers from 1 to n using a loop
    """
    # TODO: Implement the function
    return 0

def count_even_numbers(numbers):
    """
    Counts how many even numbers are in the given list
    """
    # TODO: Implement the function
    return 0
```

Test Cases

```
def sum_n_numbers(n):
    """
    Returns sum of numbers from 1 to n using a loop
    """
    # TODO: Implement the function
    return 0

def count_even_numbers(numbers):
    """
    Counts how many even numbers are in the given list
    """
    # TODO: Implement the function
    return 0
```

5. Expression Statement

Source Codes

```
def multiply_and_print(a, b):
    """
    Multiplies a and b and prints the result.
    """
    # TODO: Implement the function
    pass
```

Test Cases

```
def test_multiply_and_print(capfd):
    multiply_and_print(2, 3)
    out, err = capfd.readouterr()
    assert out.strip() == '6'

    multiply_and_print(5, 5)
    out, err = capfd.readouterr()
    assert out.strip() == '25'
```

✅ Features of this assignment:

- Covers all chapter objectives: expressions, operators, statements, loops, conditionals.
- Students start with failing tests.
- Encourages step-by-step coding and testing using pytest.
- Includes expression statements (print) and different statement types.