1. Given the following code:

**x = 10**

**y = 5**

**z = (x - y) \* (x + y) / x**

Which of the following statements is **correct** about **z**?

A) **z** will always be an integer for any values of **x** and **y**.

B) **z** will be a float because of implicit type conversion in division.

C) **z** will be a float only if **x** or **y** is a float.

D) **z** will raise an error because of operator precedence.

**Answer: B**

1. What will be the output of the following code?

**a = 5**

**b = 2**

**a, result, b = b, (a % b) + (a // b) \* b, a**

**print(a, b, result)**

**A)** **2 5 5**

**B)** **2, 5, 5**

**C)** **2, 2, 5**

**D)** **2 2 5**

**Answer: A**

1. Which of the following expressions evaluates to **True** in Python? (assume **a = 5, b = 10, c = -5**)

A) **a + b + c == b** B) **(a \* c) % b == 0** C) **a - b \* c < 0** D) **b // a == c**

**Answer: B**

1. What is the output of the following code snippet?

**a = 16**

**b = 3**

**result = (a // b) \*\* (b // a + 1)**

**print(result)**

A) 4  
 B) 5  
 C) 16  
 D) Error

**Answer: B**

1. Consider the following code:

**x = 5**

**y = 2**

**z = (x \*\* y) // (y \*\* x)**

A) **z** will always be an integer for any **x > y > 0**.

B) **z** will always be a float due to division precedence.

C) **z** will raise a **ZeroDivisionError** for **y = 0**.

D) **z** will evaluate to a power of **x** if **y** is even.

**Answer: C**

1. In which of the following cases will the **associativity** of the operator **NOT** play any role in determining the result?

A) (4 \*\* 2 \*\* 3)  
 B) (10 // 3 // 2)  
 C) (8 / 4 / 2)  
 D) (5 + 10 \* 2 - 3)

**Answer: D**

1. What will the following code print?

**x = 10**

**y = 20**

**if x \* 2 // y + y % x == 2 and x \*\* 2 > y:**

**print("Condition Met")**

**else:**

**print("Condition Not Met")**

A) Condition Met  
 B) Condition Not Met  
 C) Error  
 D) None of the above

**Answer: B**

1. Consider the following code:

**x = 25**

**y = 5**

**if x // y == y or x % y > 0:**

**if x \*\* (y // y) > y \*\* 2:**

**print("Nested Condition Met")**

**else:**

**print("Nested Condition Not Met")**

**else:**

**print("Outer Condition Not Met")**

What will it print?

A) Nested Condition Met  
B) Nested Condition Not Met  
C) Outer Condition Not Met  
D) Error

**Answer: B**

1. What will be the output of the following code?

**x = 8**

**y = 2**

**result = 0**

**for i in range(1, x // y + 1):**

**if (x \*\* y) % i == 0:**

**result += i**

**print(result)**

A) 3

B) 7

C) 12

D) 4

**Answer: B**

1. What does the following code snippet do?

**x = 9**

**y = 3**

**while x >= y:**

**x //= y**

**y += 1**

**print(x, y)**

A) Outputs **x** and **y** after reducing **x** to less than **y** through floor division.

B) Outputs the first instance where **x** becomes less than or equal to **y** through exponential reduction.

C) Raises an error due to invalid operator precedence.

D) Outputs **0** for **x** in all cases.

**Answer: A**

1. What will the following code print?

**x = 29**

**y = 15**

**result = (x & y) | (x ^ y)**

**print(result)**

A) 15  
B) 29  
C) 30  
D) 31

**Answer: D**

1. Which of the following statements about bitwise operators is **True**?

A) **x | y** is equivalent to **(x & ~y) ^ (~x & y).**

B) **x & y** always results in a number less than or equal to **x** and **y**.

C) **x ^ y** is commutative but not associative.

D) **~x** is not equivalent to **-(x + 1)**.

**Answer: C**

1. Which of the following statements is **True** about left shift (**<<**) and right shift (**>>**) operators in Python?

A) **(x << y)** is equivalent to multiplying **x** by **2 \*\* y**, while **(x >> y)** is equivalent to dividing **x** by **2 \*\* y.**  
B) **(x << y)** discards higher-order bits and **(x >> y)** adds zeros to the lower-order bits.  
C) Negative numbers cannot be used with **<<** and **>>.**  
D) **x >> y** is the same as **(x // y)** only for positive integers.

**Answer: A**

1. What will the following code print? (assume Integer = 8 bits)

**x = -16**

**y = 2**

**result = x >> y**

**print(result)**

A) -8

B) -4

C) -2

D) Error

**Answer: B**

1. What will the following code print?

**x = [1, 2, 3, 4, 5]**

**y = 3**

**result = (y in x) and (y + 1 not in x)**

**print(result)**

A) True

B) False

C) Error

D) None

**Answer: B**

1. Which of the following is **True** about membership operators?

A) **in** can only be used with lists, sets, and tuples, but not dictionaries or strings.  
B) **not in** returns True if the element is not found in the iterable, including dictionaries' keys.  
C) Membership operators raise an error if used with mixed data types (e.g., integers in strings).  
D) **not in** is the negation of **in** but cannot be used with sets.

**Answer: B**

1. Given the following code, which statement is **True**?

**x = [10, 20, 30, 40]**

**y = (10, 20)**

**result = all(i in x for i in y)**

A) **result** is always True for any x and y.  
B) **result** checks if all elements of y are in x.  
C) **result** raises an error because tuples cannot use membership checks with lists.  
D) **result** will always be False.

**Answer: B**

1. What does the following code snippet output?

**x = [1, 2, 3]**

**y = [1, 2, 3]**

**print(x is y, x == y)**

A) True True

B) True False

C) False True

D) False False

**Answer: C**

1. What will the following code output?

**x = "Logic"**

**y = "Logic"**

**z = "LogicWhile!"[:5]**

**print(x is y, x is z)**

A) True True

B) True False

C) False True

D) False False

**Answer: B**

1. Which of the following is **True** about identity operators in Python?

A) **is** checks value equality, while **==** checks reference equality.  
B) **is** compares object identity, while **==** compares value equality.  
C) Immutable objects like integers and strings always result in **False** when compared with **is**.  
D) **is** behaves identically to **==** in all scenarios.

**Answer: B**

1. What will be the result of the following code?

**x = "LW" \* 1000**

**y = "LW" \* 1000**

**a = 'Logic' + 'While'**

**b = 'Logic' + 'While'**

**print(a is b, x is y)**

A) True True

B) False False

C) False True

D) True False

**Answer: A**

1. What will be the result of this comparison?

**x = [1, 2, 3]**

**y = x.copy()**

**print(x == y, x is y)**

A) True True

B) True False

C) False True

D) False False

**Answer: B**

1. Consider the following statement:

**a = 7.0 // 3**

Which of the following is correct about **a**?

A) The type of **a** is **int**, and its value is **2**.

B) The type of **a** is **float**, and its value is **2.0**.

C) The type of **a** is **float**, and its value is **2**.

D) The type of **a** is **int**, and its value is **2.0**.

**Answer: B**

1. Which of the following expressions will raise a **ZeroDivisionError**?

A) **5 / 0.0**

B) **5.0 // 0**

C) **5.0 // 0 / 0**

D) All of the above

**Answer: D**

1. Which of the following is **not true** about augmented assignment in Python?

A) It is equivalent to performing the operation and then assigning the result to the same variable.

B) It creates a new object in memory every time it is used.

C) It is more efficient than writing the equivalent long-form operation.

D) For mutable objects, the operation can modify the object in place.

**Answer: B**

1. What will be the output of the following code?

**x = [1, 2, 3]**

**y = x**

**x += [4, 5]**

**z = x**

**x = x + [6, 7]**

**print(y, z, x)**

A) [1, 2, 3], [1, 2, 3, 4, 5], [1, 2, 3, 4, 5, 6, 7]

B) [1, 2, 3, 4, 5], [1, 2, 3, 4, 5], [1, 2, 3, 4, 5, 6, 7]

C) [1, 2, 3], [1, 2, 3], [1, 2, 3, 4, 5, 6, 7]

D) [1, 2, 3, 4, 5], [1, 2, 3, 4, 5], [1, 2, 3, 4, 5]

**Answer: B**

1. Which of the following is **not true** about short-circuit evaluation in Python?

A) In an **and** operation, if the first operand is **False**, the second operand is not evaluated.

B) In an or operation, if the first operand is **True**, the second operand is not evaluated.

C) Short-circuiting always ensures both operands are evaluated before returning the result.

D) Short-circuiting can be used to avoid errors by evaluating conditions in the correct sequence.

**Answer: C**

1. What will be the output of the following code?

**x = 5**

**y = 0**

**print(y and (10 / y) or x)**

A) 5

B) 10

C) **ZeroDivisionError**

D) 0

**Answer: A**

1. Which of the following expressions correctly checks if a number **n** is a power of 2 using bitwise operators?

A) **n & (n - 1) == 0 and n > 0**

B) **n | (n - 1) == n**

C) **n ^ (n - 1) == 0**

D) **n & (n + 1) == 0**

**Answer: A**

1. What will the following function **fun** return?

**def fun(n):**

**return n > 0 and (n & -n) == n**

**print(fun(16))**

A) True

B) False

C) None

D) Error

**Answer: A**

1. What will be the output of the following code?

**a = "LogicWhile"**

**b = 2**

**c = "3"**

**result = a + b \* c**

**print(result)**

A) LogicWhile6

B) LogicWhile33

C) LogicWhileLogicWhile3

D) Error

**Answer: B**

1. Which of the following code snippets will raise an error in Python?

A) **w = ("Hi" \* 3) + ("5" \* 3)**

B) **y = "Hello" + str(5)**

C) **z = 5 \* "Hello" + 2 \* "!"**

D) **x = "4" + 4**

**Answer: D**

1. What does the following code output?

**x, y, z = 5, 3, 8**

**x += y > z and z \* 2 or y**

**y \*= z < x + y or z - x**

**z -= x and y or x % y**

**print(x, y, z)**

A) 8 3 5

B) 6 12 0

C) 8 3 6

D) 6 12 6

**Answer: A**

1. In Python, which of the following statements is **True** about the behavior of **&** (bitwise AND) and **and** (logical AND)?

A) **&** is a bitwise operator and evaluates both operands, whereas **and** is a logical operator and stops evaluation if the first operand is false.  
B) **&** and **and** have the same behavior and precedence in Python.  
C) **and** performs bitwise AND, and **&** performs logical AND in Python.  
D) **and** is a bitwise operator and evaluates both operands, whereas **&** is a logical operator and stops evaluation if the first operand is false.

**Answer: A**

1. Which of the following statements about the **+** and **\*** operators in python is **True**?

A) The **\*** operator can only be used for multiplication, whereas **+** is used exclusively for addition.

B) The **+** operator is used for both addition and concatenation, while **\*** is used for both multiplication and unpacking.

C) The **\*** operator is used only for multiplication, **+** is only used for concatenation.

D) The **+** operator is used only for addition, **\*** is used only for unpacking.

**Answer: B**

**##### Sprint\_1 on Operators - END #####**