## Find the final result, and explain why

**Q1. 3 || 0**

**A1.**

The result is True because the logical OR (||) operator returns True when at least one of its operands is True. In the context of logical operations, numbers are implicitly converted to boolean values. Therefore, any number other than zero is evaluated as True. In this specific case, it can be thought of as the expression 'True || False,' which naturally evaluates to True.

## Evaluate the following expressions, and mention all the operators, operands, and sub-expressions they contain

**Q2. 2 + 3 * 4 (evaluate)**

**A2.**

**Operators:**

{+, *} -> Arithmetic

**Operands:**

{3, 4} -> *

{2, 3 * 4} -> +

**Sub-expressions:**

{2 + 3 * 4, 3 * 4} -> Constant and Arithmetic

{2, 3, 4} -> Constant

**Evaluation:**

2 + 3 * 4 = 14

**Q3. Y = 5 (evaluate)**

**A3.**

**Operators:**

{=} -> Assignment

**Operands:**

{Y, 5} -> =

**Sub-expressions:**

{Y = 5} -> Constant

{Y} -> Variable

{5} -> Constant

**Evaluation:**

The expression returns the final value of Y, which is 5

**Q4. X == Y**

**A4.**

**Operators:**

{==} -> Comparison

**Operands:**

{X, Y} -> ==

**Sub-expressions:**

{X == Y} -> Variable and Relational

{X, Y} -> Variable

**Evaluation:**

Returns True when the values of X and Y are the same, and False when they differ.

**Q5. !X**

**A5.**

**Operators:**

{!} -> Logical

**Operands:**

{X} -> !

**Sub-expressions:**

{!X} -> Variable and Logical

{X} -> Variable

**Evaluation:**

Returns True when the value of X is False, and False otherwise.

**Q6. !(-2) (evaluate)**

**A6.**

**Operators:**

{!} -> Logical

{-} -> Arithmetic

**Operands:**

{2} -> -

{-2} -> !

**Sub-expressions:**

{-2} -> Constant and Arithmetic

{!(-2)} -> Constant and Logical

{2} -> Constant

**Evaluation:**

Returns False.

**Q7. 2 + 3 * 4 / 2 (evaluate)**

**A7.**

**Operators:**

{+, *, /} -> Arithmetic

**Operands:**

{3, 4} -> *

{3 * 4, 2} -> /

{2, 3 * 4 / 2} -> +

**Sub-expressions:**

{3 * 4, 3 * 4 / 2, 2 + 3 * 4 / 2} -> Constant and Arithmetic

{2, 3, 4, 2} -> Constant

**Evaluation:**

2 + 3 * 4 / 2 = 8.

**Q8. X += Y * Z**

**A8.**

**Operators:**

{+=} -> Assignment

{*} -> Arithmetic

**Operands:**

{Y, Z} -> *

{X, Y * Z} -> +=

**Sub-expressions:**

{Y * Z} -> Variable and Arithmetic

{X += Y * Z} -> Variable and Assignment

{X, Y, Z} -> Variable

**Evaluation:**

Returns the value of X + Y * Z depending on the values of the three variables.

**Q9. (X < Y) && (Y < Z)**

**A9.**

**Operators:**

{<, <} -> Comparison

{&&} -> Logical

**Operands:**

{X, Y} -> <

{Y, Z} -> <

{(X < Y), (Y < Z)} -> &&

**Sub-expressions:**

{X < Y, Y < Z} -> Relational and Variable

{(X < Y) && (Y < Z)} -> Logical and Variable

{X, Y, Y, Z} -> Variable

**Evaluation:**

Returns True if Y has a value larger than X and less Than Z, otherwise it returns False.

**Q10. (3 > 1) && (3 > 2) && (1 >= 2) && (10 > 5)**

**A10.**

**Operators:**

{>, >, >=, >} -> Comparison

{&&, &&, &&} -> Logical

**Operands:**

{3, 1} -> >

{3, 2} -> >

{1, 2} -> >=

{10, 5} -> >

**Sub-expressions:**

{3 > 1} -> Relational and Constant

{(3 > 1) && (3 > 2),

(3 > 1) && (3 > 2) && (1 >= 2),

(3 > 1) && (3 > 2) && (1 >= 2) && (10 > 5)} -> Logical and Constant

{3, 1, 3, 2, 1, 2, 10, 5} -> Constant

**Evaluation:**

Returns False.

**Short-Circuit Evaluation:**

The checking stops at the (1 >= 2) expression, because the accumulated value will be False, and the logical AND (&&) returns False if at least of of its operands is False

**Q11. (3 < 1) || (3 == 3) && (12 / 3) (evaluate and give the expression stops the evaluation)**

**A11.**

**Operators:**

{<, ==} -> Comparison

{||, &&} -> Logical

{/} -> Arithmetic

**Operands:**

{3, 1} -> <

{3, 3} -> ==

{12, 3} -> /

{(3 < 1), (3 == 3)} -> ||

{(3 < 1) || (3 == 3), (12 / 3)} -> &&

**Sub-expressions:**

{(3 < 1)} -> Relational and Constant

{(3 == 3)} -> Relational and Constant

{(3 < 1) || (3 == 3)} -> Logical and Constant

{(12 / 3)} -> Arithmetic and Constant

{(3 < 1) || (3 == 3) && (12 / 3)} -> Logical and Constant

{3, 1, 3, 3, 12, 3} -> Constant

**Evaluation:**

Returns True.

**Short-Circuit Evaluation:**

The evaluation process pauses at the (12 / 3) expression. This occurs because the accumulated value for the || operator becomes False, requiring it to assess the next side before delivering a result. In this case, the other side evaluates to True. The accumulated value for the && operator, however, becomes True, necessitating

the examination of the subsequent operand before providing a final answer. This next operand is the expression (12 / 3), which equals 6, and therefore evaluates to True. Consequently, the && operator combines the two True operands, resulting in a final outcome of True.

**Q12. (3 < 1) || (3 == 3) || (12 / 3) (evaluate and give the expression …)**

**A10.**

**Operators:**

{<, ==} -> Comparison

{||, ||} -> Logical

{/} -> Arithmetic

**Operands:**

{3, 1} -> <

{3, 3} -> ==

{12, 3} -> /

{(3 < 1), (3 == 3)} -> ||

{(3 < 1) || (3 == 3), (12 / 3)} -> ||

**Sub-expressions:**

{(3 < 1)} -> Relational and Constant

{(3 == 3)} -> Relational and Constant

{(3 < 1) || (3 == 3)} -> Logical and Constant

{(12 / 3)} -> Arithmetic and Constant

{(3 < 1) || (3 == 3) || (12 / 3)} -> Logical and Constant

{3, 1, 3, 3, 12, 3} -> Constant

**Evaluation:**

Returns True.

**Short-Circuit Evaluation:**

The evaluation process comes to a halt when it reaches the (3 == 3) expression. This is because the accumulated value for the || operator turns to False, prompting it to assess the next operand, which, in this case, is found to be True. Consequently, the accumulated value for the || operator becomes True, obviating the need to check the following operand, and the final result is True.