PYTHON

Nested if Statements:

Syntax:

```
if condition1:
   # code if condition1 is true
   if condition2:
      # code if condition2 is also true
   else:
      # code if condition2 is false
else:
   # code if condition1 is false
```

Example:

```
num = 10
if num > 0:
   if num % 2 == 0:
      print("The number is positive and even.")
   else:
      print("The number is positive but odd.")
else:
   print("The number is not positive.")
```

Left Shift and Right Shift:

- **Left Shift (<<)** shifts bits to the left, multiplying by 2n2*n*.
- **Right Shift (>>)** shifts bits to the right, dividing by 2n2*n*.

Example:

```
a = 5      # 0b0101
print(a << 1)  # 10 (0b1010)
print(a >> 1)  # 2 (0b0010)
```

Bitwise AND, OR:

- **Bitwise AND (&)**: Each bit of the result is 1 if both corresponding input bits are 1.
- **Bitwise OR (|)**: Each bit of the result is 1 if at least one corresponding input bit is 1.

Example:

```
a = 6   # 0b0110
b = 3   # 0b0011
print(a & b)  # 2 (0b0010)
print(a | b)  # 7 (0b0111)
```

Bit-wise Operators:

- & : Bitwise AND
- | : Bitwise OR
- ^ : Bitwise XOR (exclusive OR)
- ~ : Bitwise NOT (ones' complement)
- <<: Bitwise left shift
- >>: Bitwise right shift

Formula: n**2

This formula is frequently used with bit shifting:

- Left shifting by n*n* places multiplies a value by 2n2*n*.
- Right shifting by n*n* places divides by 2n2*n* (floor division for integers).

Example:

```
n = 3
print(2 ** n)   # 8
x = 2
print(x << n)   # 16, same as 2 * (2 ** 3)
```

All these operations are core parts of Python and useful for low-level data processing and logical control flow.