**Core Python Concepts**

1. **Understanding data types: integers, floats, strings, lists, tuples, dictionaries, sets**
   1. numeric types

* int: Represents integers, e.g., 5, -3, 42.
* float: Represents floating-point numbers, e.g., 3.14, -0.001.
  1. sequence types
* str (String): Immutable sequence of characters, e.g., "hello".
* list: Mutable, ordered collection, e.g., [1, 2, 3].
* tuple: Immutable, ordered collection, e.g., (1, 2, 3).
  1. mapping Type
* dict (Dictionary): Unordered collection of key-value pairs, e.g., {"name": "Alice", "age": 30}.
  1. set types
* set: Mutable, unordered collection of unique elements, e.g., {1, 2, 3}.
* frozenset: Immutable version of set.
  1. boolean type

bool: Represents True or False values.

1. **Python variables and memory allocation.**

* Dynamic Typing: Variables in Python are dynamically typed, meaning their type is determined at runtime.
* Memory Management: Python uses automatic memory management, including garbage collection, to handle memory allocation and deallocation.
* Object References: Variables hold references to objects in memory. For example, when you assign x = 5, x references the integer object 5.
* Immutability: Immutable objects (like integers and strings) cannot be altered after their creation. Assigning a new value to a variable creates a new object.

1. **Python operators: arithmetic, comparison, logical, bitwise**

**1. Arithmetic Operators**

* +: Addition
* -: Subtraction
* \*: Multiplication
* /: True division (returns float)
* //: Floor division (returns integer quotient)
* %: Modulo (remainder)
* \*\*: Exponentiation

**2. Comparison Operators**

* ==: Equal to
* !=: Not equal to
* <: Less than
* >: Greater than
* <=: Less than or equal to
* >=: Greater than or equal to

**3. Logical Operators**

* and: Returns True if both operands are true
* or: Returns True if at least one operand is true
* not: Returns the negation

**4. Bitwise Operators**

* &: Bitwise AND
* |: Bitwise OR
* ^: Bitwise XOR
* ~: Bitwise NOT
* <<: Left shift
* >>: Right shift