**1. Classes and Objects**

* **Objects as Instance Variables**:
  + Can use built-in types or custom classes (e.g., self.minions = [], self.lead = People("Gru")).
* **Objects as Function Parameters**:
  + Pass objects to functions to modify their state (e.g., minion.eat\_banana(5) updates minion\_kevin's weight).
* **Objects in Collections**:
  + Store objects in lists/dictionaries (e.g., minion\_army = [minion\_kevin, minion\_stuart]).
* **The**self**Variable**:
  + Refers to the instance; required in methods (e.g., def steal\_moon(self):).
* **Why Use Classes?**
  + Modularity, code reuse, and consistency (e.g., avoid mismatched lists like minion\_names = ["Kevin"], minion\_height = [159]).

**2. Complexity**

* **Big O Notation**:
  + Measures worst-case time complexity:
    - List search: **O(n)** (linear time).
    - Dictionary key lookup: **O(1)** (constant time).
* **Algorithm Efficiency**:
  + Insertion Sort: **O(n²)**.
  + Merge Sort: **O(n log n)**.
* **Data Structures Matter**:
  + Choose based on operations needed (e.g., dictionaries for fast lookups, lists for ordered data).