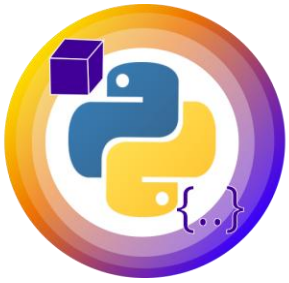
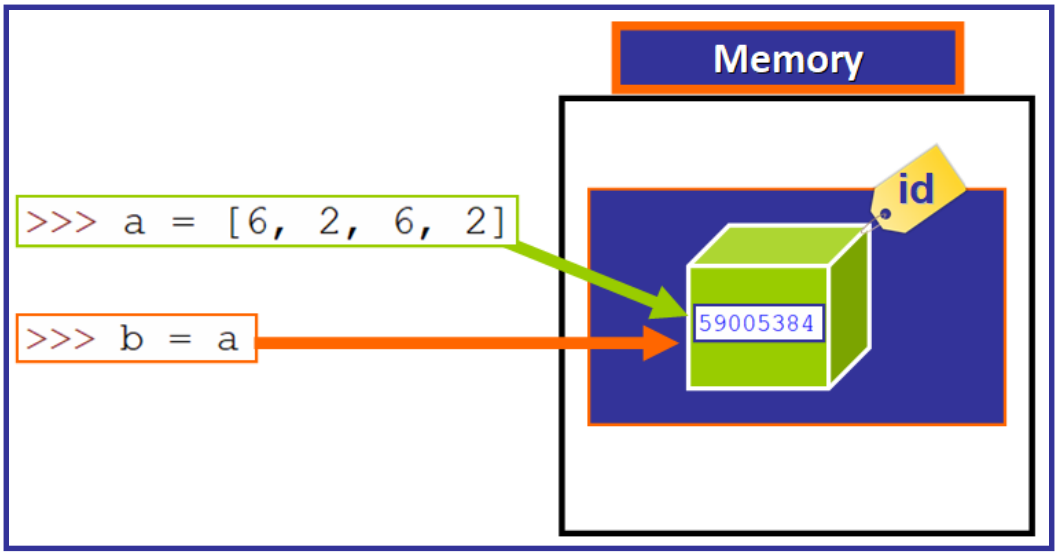


Python OOP:
Aliasing, Mutation, & Cloning

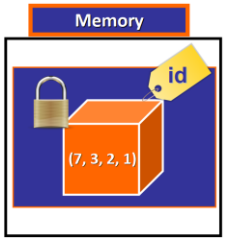
Key Takeaways

- **Aliasing:**
  - Occurs when **two or more variables point to the same object** in memory.
  - These variables that point to the same object are called “aliases.”
  - Aliases have the **same id** value when you call the id() function.
  - **Warning:** Aliases can be a source of bugs because you can modify an object using one the aliases and forget that other names point to the same object, so they will be modified too.

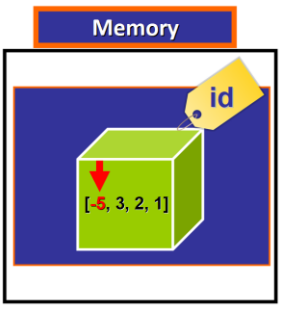


Key Takeaways

- **Mutation:**
  - There are two types of objects:
    - **Mutable**: **can** change after they've been defined.
    - **Immutable**: **can't** change after they've been defined.
  - Built-in Mutable Objects:
    - Lists, Sets, Dictionaries, more…



  - Built-in Immutable Objects:
    - Tuples, Strings, Floats, Integers, Booleans, more…
  - **Advantages:**
    - Mutable: can be modified easily without additional memory usage.
    - Immutable: safer from bugs and easier to understand.
  - **Disadvantages:**
    - Mutable: more prone to bugs due to aliasing.
    - Immutable: additional memory usage because a new object has to be created when you need a modified copy of the original.



Key Takeaways

- **Cloning:**
  - It is making a copy of the object in memory.
  - This copy has the same value of the original object, but it is a **different object in memory** (not an alias).
  - This way, you can modify the clone without affecting the original object.