Great question! You're asking whether **numbers and strings** are cleaned up by the **garbage collector** *later* (or handled differently) than **lists, dictionaries**, or other data types.

✅ Short Answer:

Yes, **numbers and strings** (especially small ones) are treated **differently** from lists/dictionaries because of something called **interning** and **object reuse**.
So they often **don't get cleaned up immediately** — or **at all**, in many cases.

🔍 Full Explanation:

1. **Numbers and Strings (Immutable types)**

- Python **internally caches** small integers (`-5` to `256`) and short strings (like `"hello"`).
- These are reused **again and again** throughout the program.
- So even if you do:

```python
x = 10 del x
```

Python **does not delete** the `10` — it's kept in memory because it's **shared** and used **often**.

➕ **These objects:**

- Are **immutable** (cannot be changed).
- Are **reused** to improve **speed and memory usage**.
- Often **don't need garbage collection** because they are never really removed.

---

2. **Lists, Dictionaries, Sets (Mutable types)**

- These are **not reused** by Python automatically.
- When reference count goes to **zero**, they are **immediately deleted** by Python.
- If they are part of **circular references**, Python's **garbage collector (GC)** will eventually clean them up.

Example:

```python
a = [1, 2, 3] b = a del a del b # Now the list is removed from memory
```

🔄 Comparison Table:

Type	Reused / Interned	Needs GC?	Deleted Quickly?
`int`, `str`	✅ Yes	❌ Usually not	❌ No
`list`, `dict`	❌ No	✅ Sometimes	✅ Yes

🧠 Summary:

- **Numbers/strings** are reused (cached) → not deleted right away.
- **Lists/dicts** are custom objects → deleted when not used.
- **Garbage collector** is mostly for cleaning **cycles**, not simple variables.