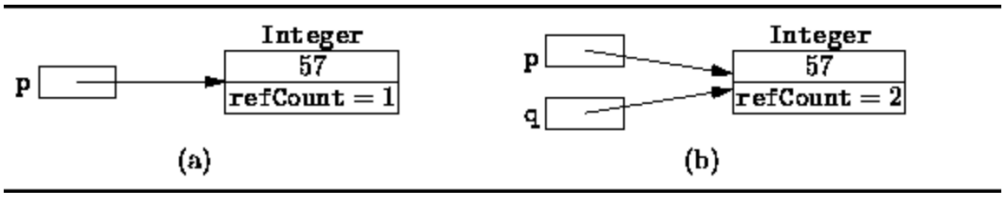
Memory Management (II)

Garbage Collection

- Garbage is any block of heap memory that cannot be accessed by the program.
- It includes the memory that **was allocated but not free after use**, and the pieces of **memory that not large enough to meet the allocating requests**.
- Garbage collection needs to **identify garbage** and then **make garbage memory available**.
- A separate process running in the background to reclaim unused heap areas for later use by the program.
- How does a garbage collection system know when an object is no longer in use?
- When should garbage collection occur?
- There are many different strategies in use.
- Here are three general strategies: **reference counting**, **mark-sweep**, and **copy collection**.
- Specific garbage collection systems may be hybrids.
- All these strategies leverage linked list to manage memory. All **unused nodes are in a singly linked list**.

Reference Counting

- **Each node has a field to count** the number of references to that node.
- **Assignments involving references must increment the counter** of the node specified by the r-value.
  - E.g. Integer p = new Integer(57); Integer q = p;



- **Sometimes**, when assigning, it also **needs to decrement the counter of another node**.
  - E.g. Integer p = new Integer(57); Integer q = new Integer(99); p = q;

