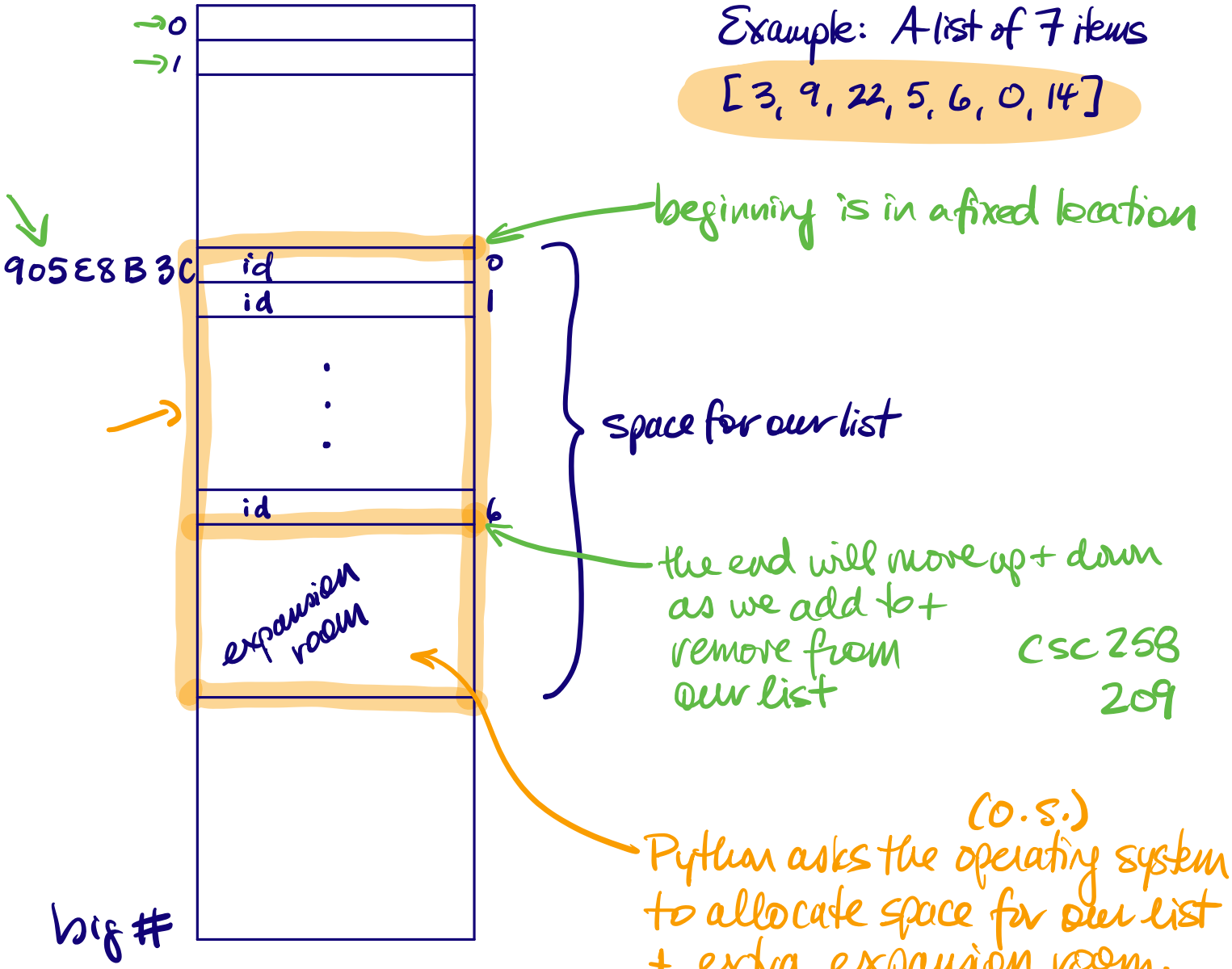
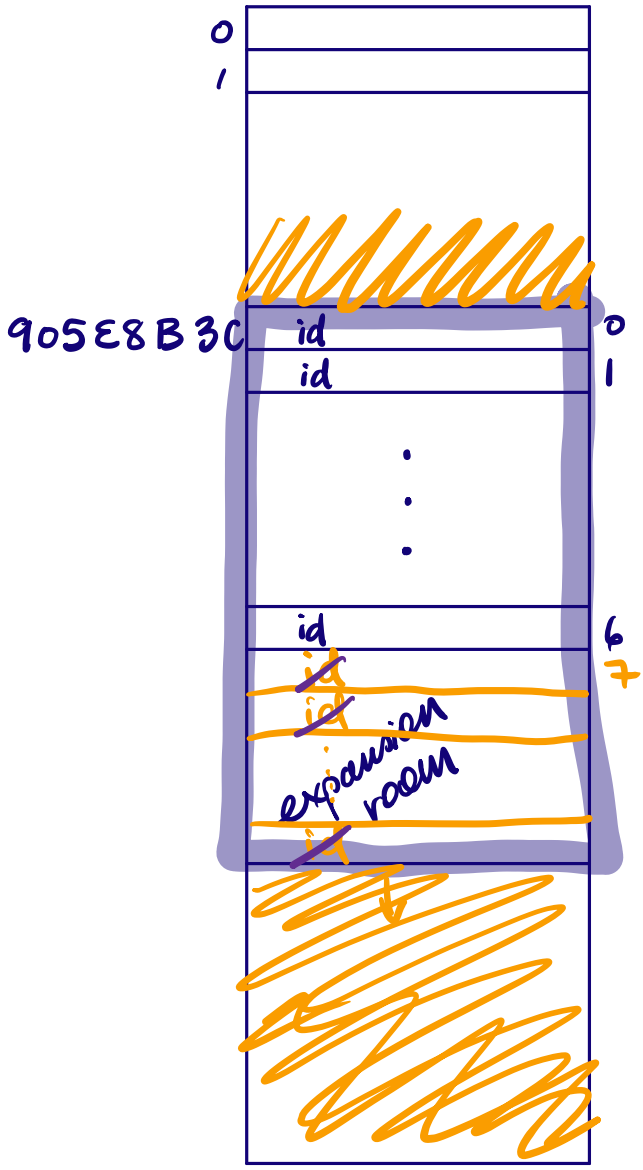
How a Python list is stored



Indexing into our list

just some arithmetic + go there → vv fast

Updates at the end of our list



Insertion is usually vv fast: Just go into the expansion room.

Occassionally, run of expansion room.

When that happens, we can't expand beyond the space that the operating system has allocated for this. It is likely already in use for something else.

Instead Python ① asks the O.S. for new space, somewhere else in memory, that is even larger.

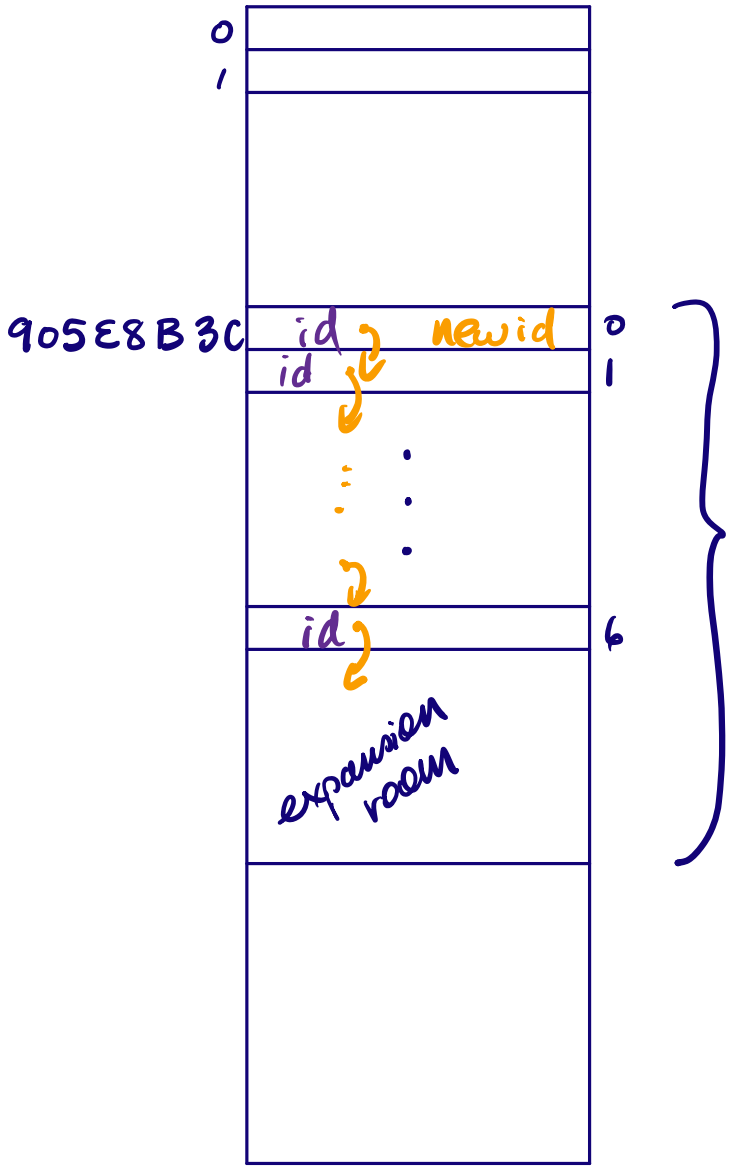
② Then copies each list item into the new space, one at a time. This is vv slow.

③ Finally, Python gives the original space back to the O.S.

The O.S. keeps track of what memory space is in use and what chunks (of what size) are available.

It uses a structure called a "heap" for this. You'll learn heaps in CSC263.

Updates at the front of our list



Insertion is VV slow.
Must shift every other element down!

Deletion is slow too: must shift all the other elements up by one.

(We can just consider item 0 to be one slot further down. We need to have the beginning of the list in a fixed position in order to have very fast list indexing.)