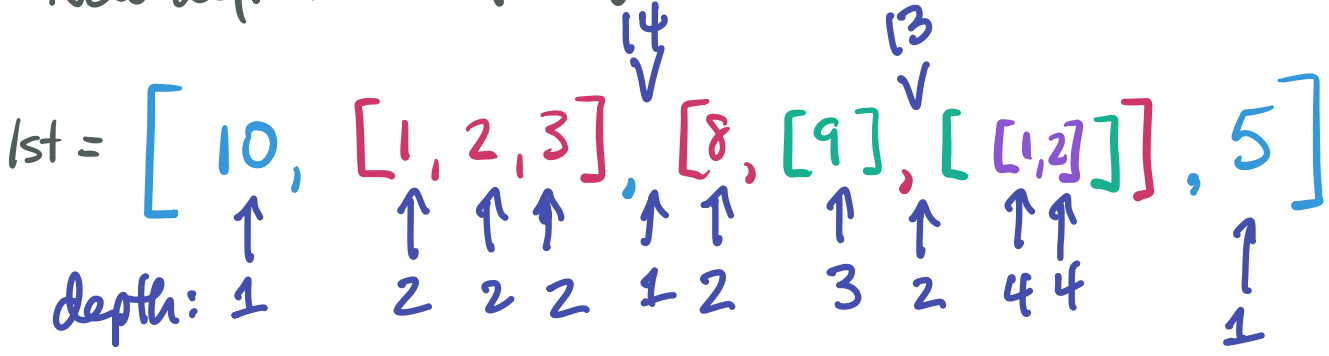
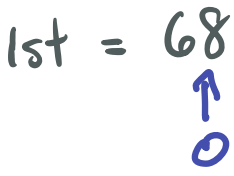
First_at_depth

New definition: depth of an object within a nested list.






```
def first_at_depth(obj: Union[int, List], d: int) -> Optional[int]:
    """Return the first (leftmost) item in <obj> at depth <d>.

    Return None if there is no item at depth <d>.

    Precondition: d >= 0.
```

1. Write doctests
2. Trace one big-ish example.
   first_at_depth([10, [[20, 21]], [30, 40]], 2)

move sublists here would not matter!

first_at_depth([10,[[20,21]],[30,40]], 2)

sublist	call	do/return correct result
→ 10	first_at_depth(10, 1)	None
→ [[20,21]]	first_at_depth([[20,21]], 1)	None
→ [30,40]	first_at_depth([30,40] 1)	30 stop!

Even if there were additional sublists in obj, there would be no reason to look at them. We've already found something at depth d + anything that comes later cannot therefore be first!