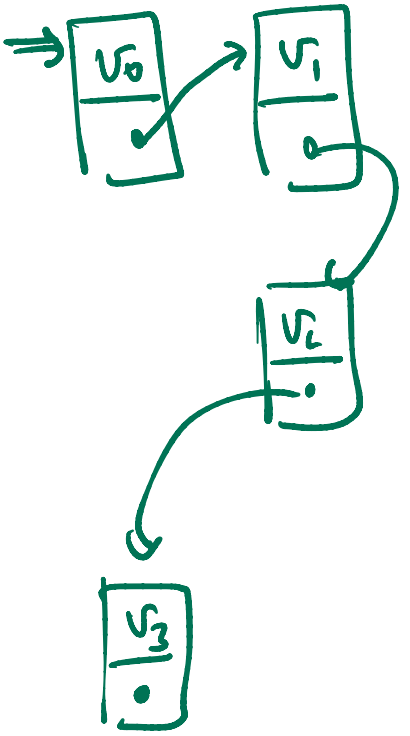
Array

Position of a[k] from a[0] by arithmetic

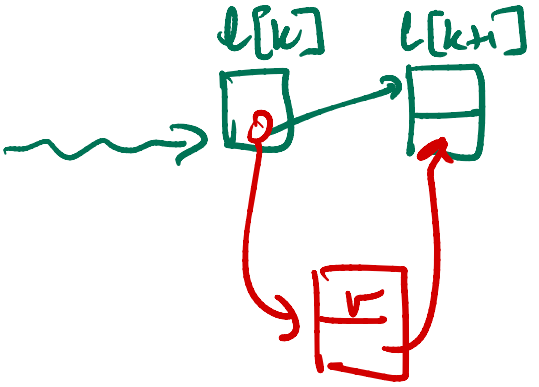
Random Access

Lists



Navigating to a[k] takes time proportional to k

List



Insertion & deletion are "easy"

Array

Insert/delete require shifting values to make a space or compress a space

Dictionary

Array for storage

Hash function — maps keys to positions in array

- Collision $h(k_1) = h(k_2)$, $k_1 \neq k_2$

Random access modulo collisions

Python lists?

Seem to be flexible, like "real" lists

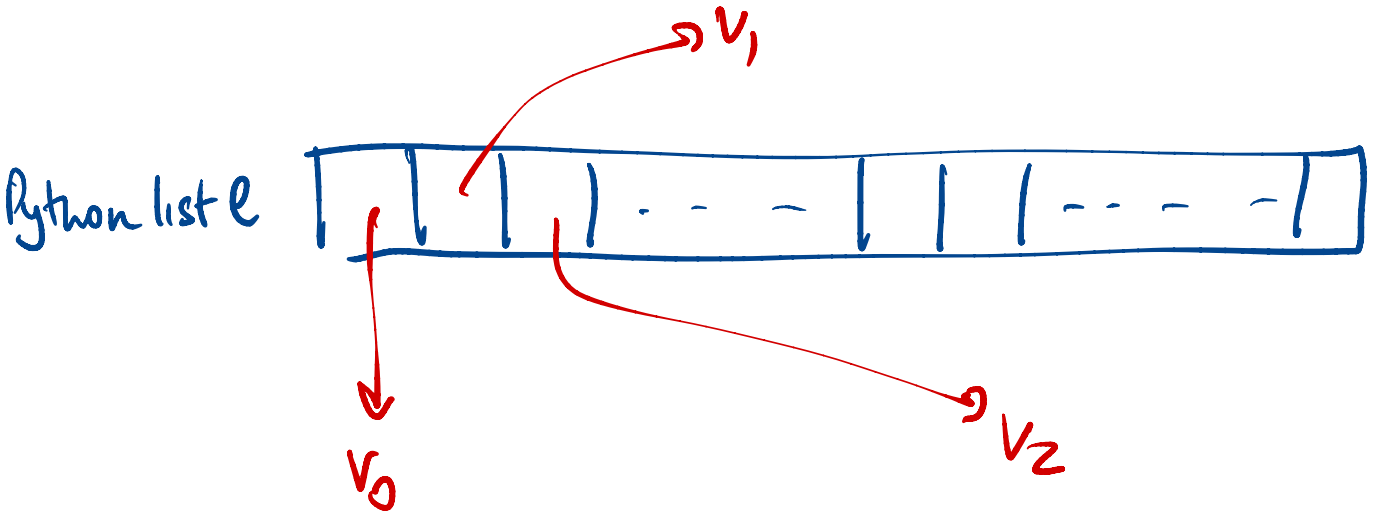
Python lists are actually arrays

l=[] → allocate an array (of some size) for l

What if l grows beyond size of array?

Allocates an array of double the size

But ... Python lists are not of uniform type



Implement a linked ("real") list using dictionary

$d = \{$ value : v_0,

next : { value : v_1,

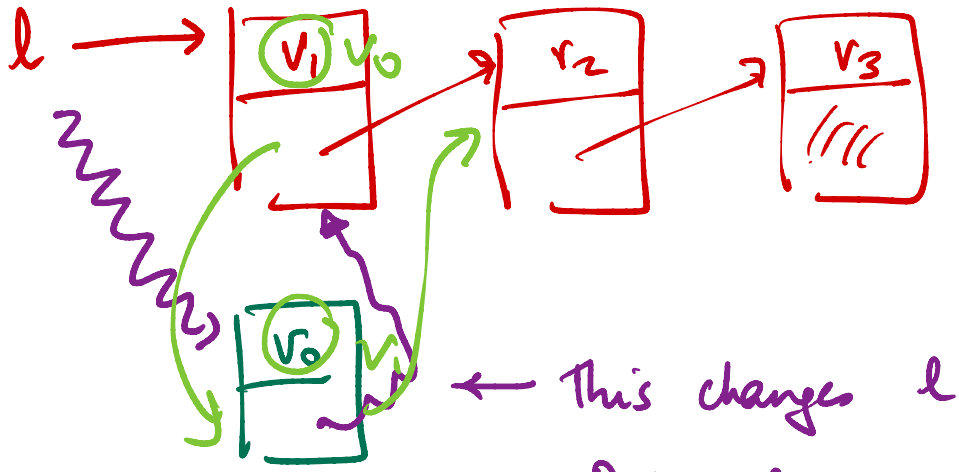
next : { value : v_2,

next : ___

value : v_{n-1}

next : {}

Inserting at the beginning of a list.



Insert v_0 at beginning

← This changes l inside the function,
Outer l is unchanged

Solution? Swap v_0 & v_1

```
def myappend(l, v):
```

`l = l + [v]` vs `l.append(v)`

✗ ✓