Lecture 12, 16 September 2025

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Programming and Data Structures with Python

Arrays, lists, Dictionaries

Sequence of values, indexed by position

Array

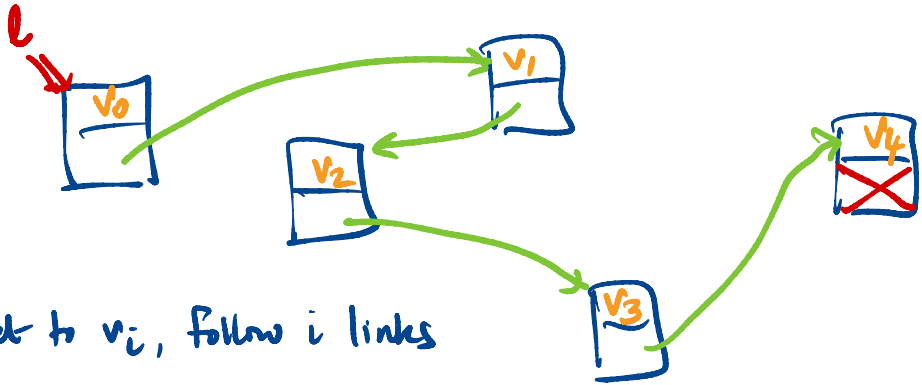
Contiguous [No gaps]

l[0] + i * width

Random access

Accessing l[i] has uniform cost, indep. of i

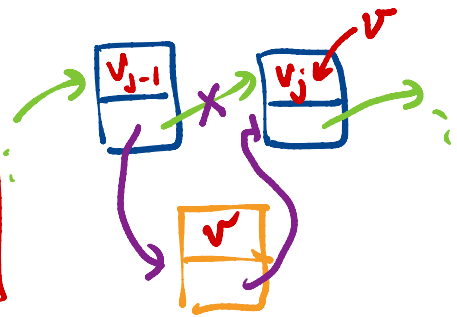
"True" list



To get to v_i, follow i links

"Flexible"

Operations

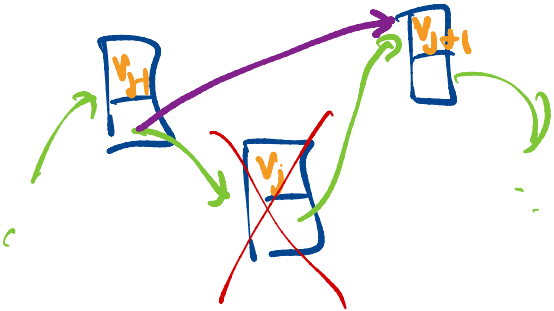
	Array	List
l[i]	Uniform, indep of i	Proportional to i
insert(j,v)	Make space at l[j] for v Shift current l[j:] right Time prop to n-j Where is the space?	Assume we are at l[j] 

del(L[j])

Array

Shift up n-j

List



Python lists are "flexible" arrays

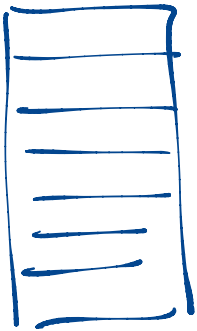
- Space requirement? A large array to start with + End of list index

Initial space runs out? Double & copy

One time cost

"Amortise" across initial inserts

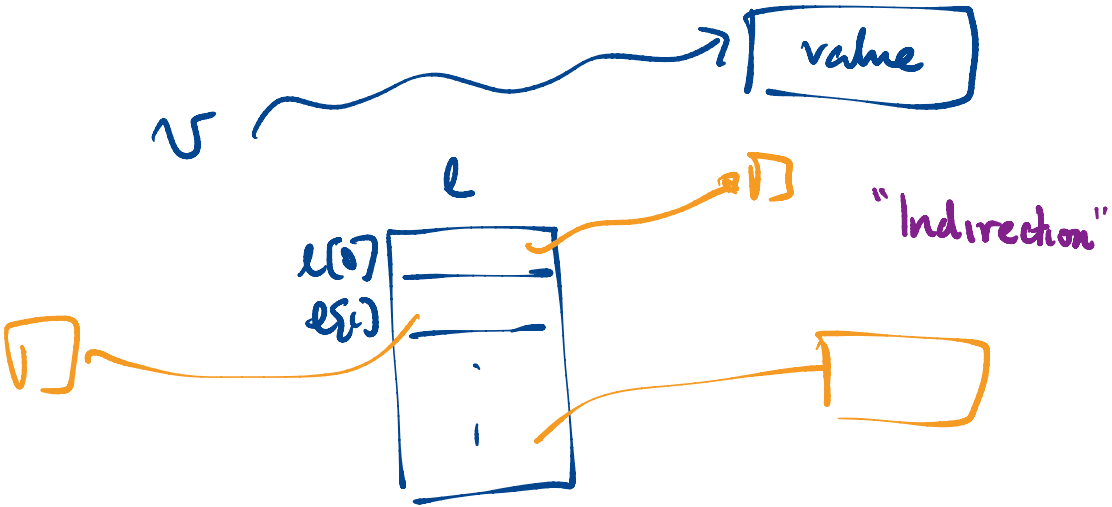
Array



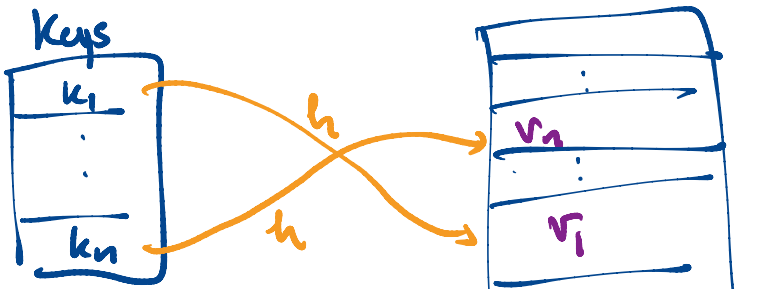
All values take equal space

Python lists can hold arbitrary (mixed) values

Variables & values



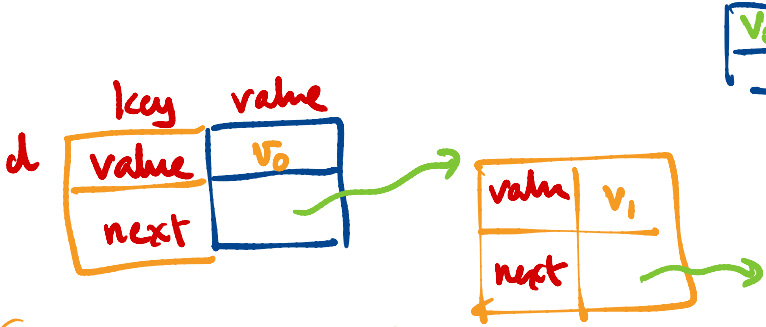
Dictionaries

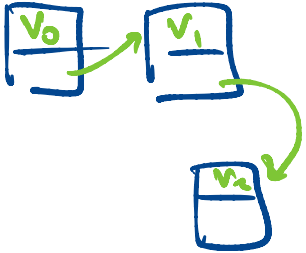


d[k] = v

- create a new pair k:v
- update current k to k:v

Implementing a "real" list





{value : v0, next : {value : v1, next : {value : v2, next : None}}}

Append a value 878

newd = {3

newd.value = 878

newd.next = None

Insert (0, 878)

d' = { value: 878,

next: d { value : v_0,

next : { value : v_1,

next

:

{ value : v_{n-1}

next : None } _ }

Call by reference.

insert (d, 0, v)

Insert at head of list