Lecture 10, 17 September 2024

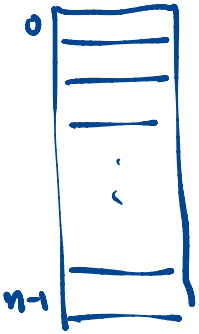
Madhavan Mukund

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

Programming and Data Structures with Python

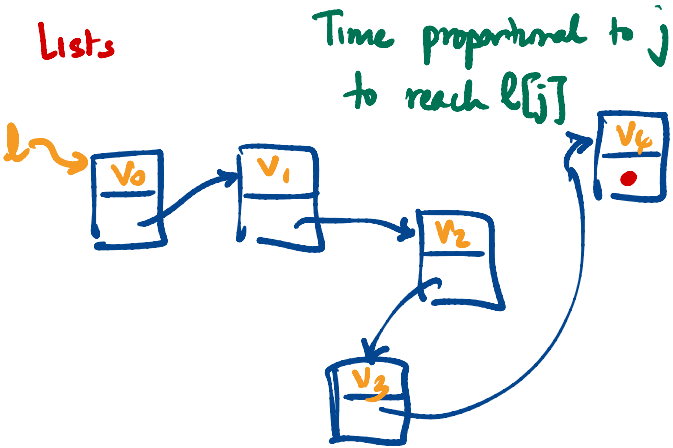
Arrays, lists, dictionaries

Array - Random access



Lists

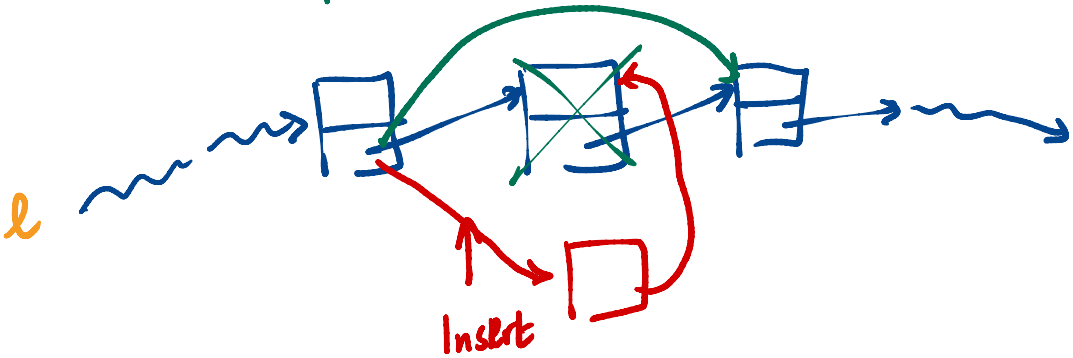
Time proportional to j to reach l[j]



Lists

"Flexible"

Size is not fixed

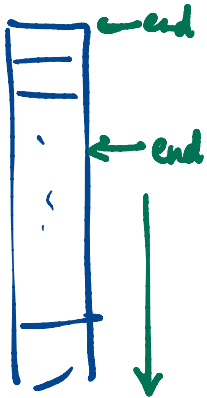


Python "lists"

Flexible array

Allocate

l=[]



del(l[j])

"removes" l[j]

Shift l[j+1] -- back by 1

Run out of space - allocate a new array of double size, copy

l.insert(0,v)

list 0 .. n-1

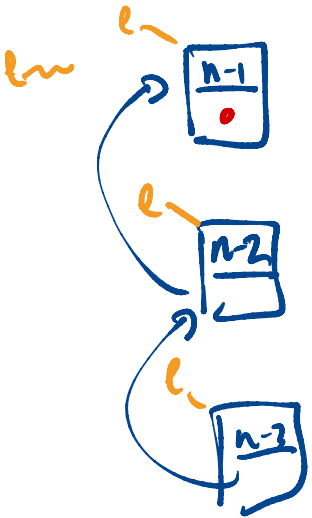
l.insert(0,n-1) – 1

l.insert(0,n-2) – 2

l.insert(0,n-k) – k

l.insert(0,0) – n

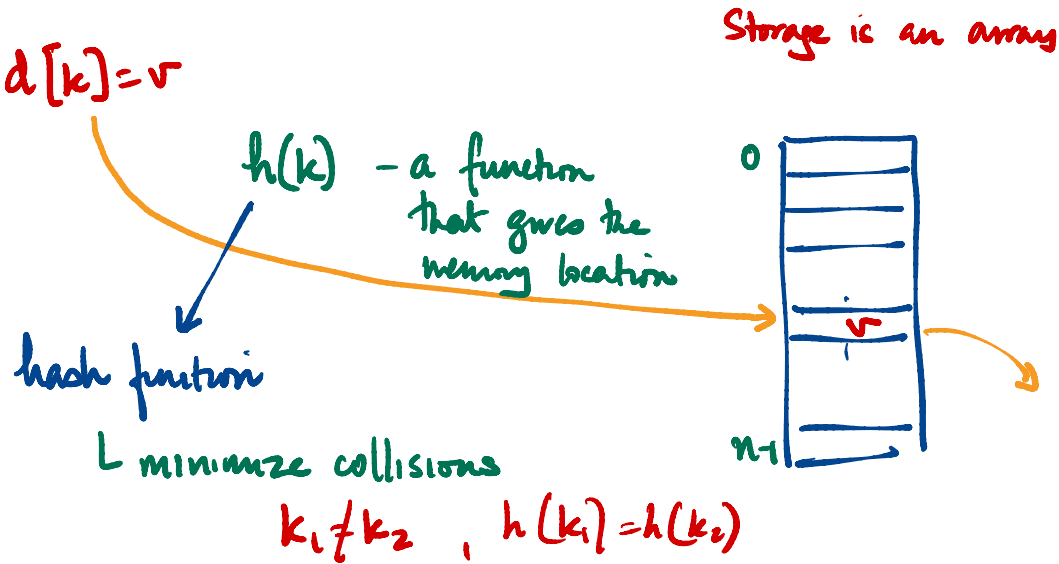
$$1 + 2 + \cdots + n = \frac{n(n+1)}{2} \propto n^2$$



$l = [0, \ldots n-1]$

n times $del(l[0])$

Dictionaries



Ideally

$$k_1 \neq k_2 \implies h(k_1) \neq h(k_2)$$

$$h(k_1) = h(k_2) \implies k_1 = k_2$$

Error correction / Tamper detection

Inserting into a dictionary is random access

$d[k] = v \longrightarrow j = h(k)$

$l[j] = v$

Measuring running time in Python

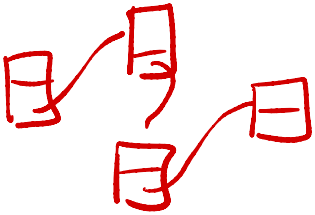
time library

t1 = time.perf_counter() — returns a number

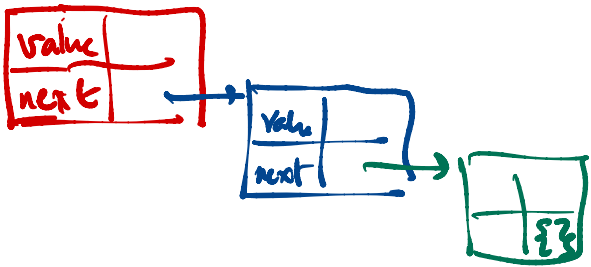
t2 = time.perf_counter()

t2 - t1 is running time in seconds

Implementing a "true" list



Using a dictionary



Append

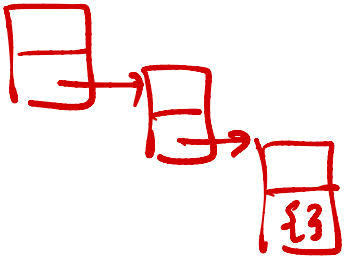
Start with d

While d[next] is not {}

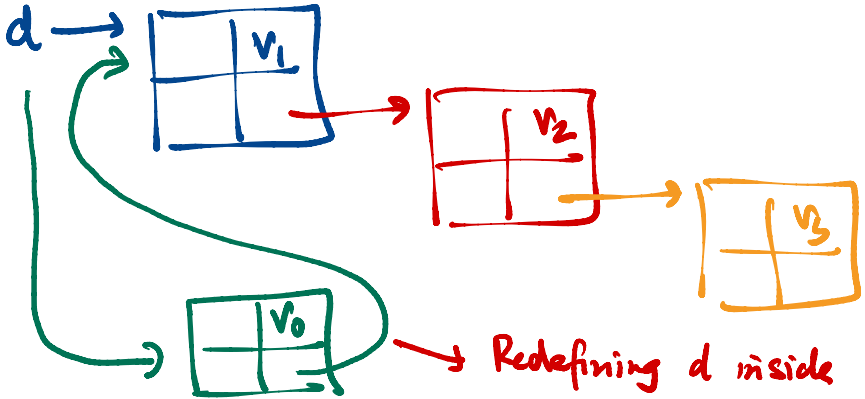
d = d.next

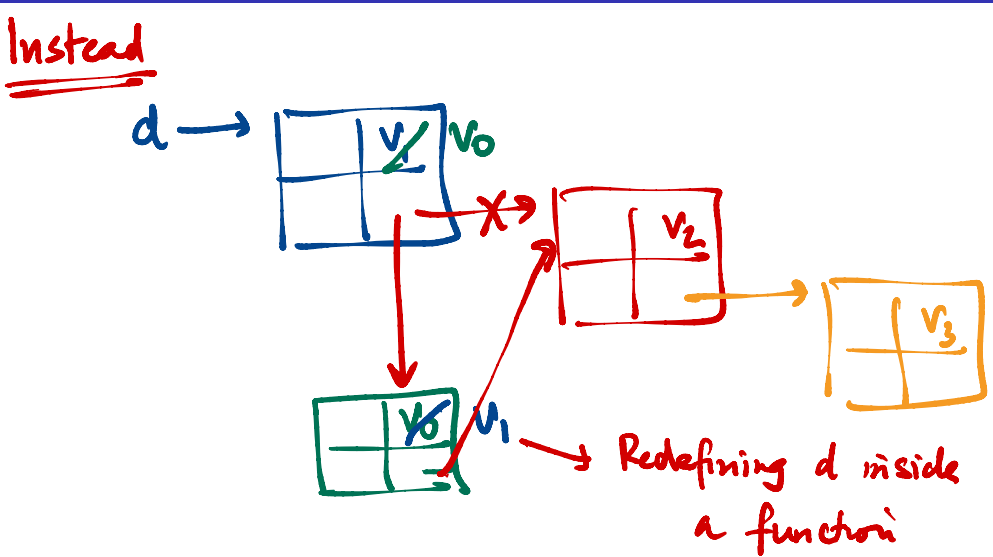
d[value] = v

d[next] = {}



Insert at 0




Instead
d
v1
v0
X
v2
v3
v0
v1
Redefining d inside
a function