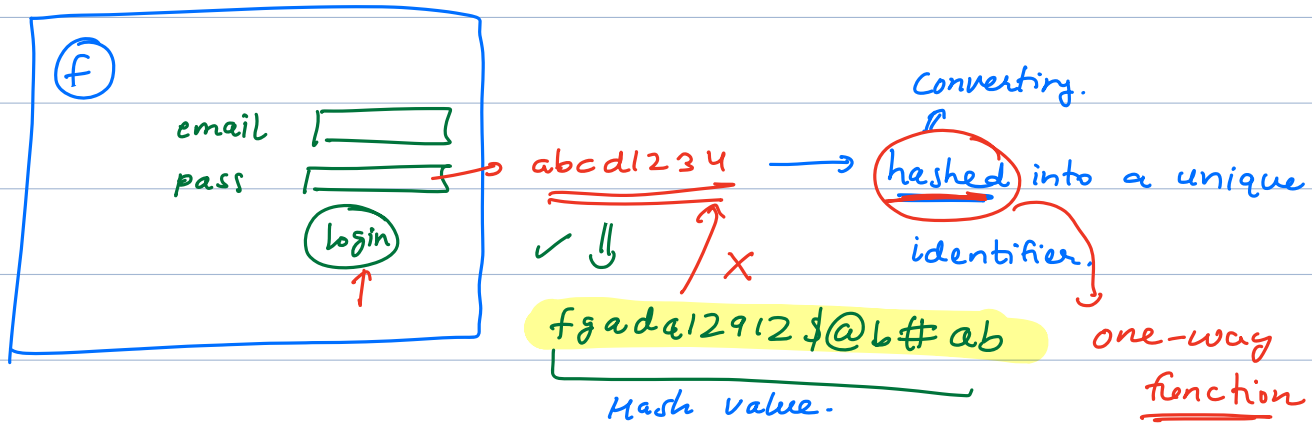
Hashing – 1

Problem Solving (optional) tomorrow 9 pm

– Sachin Kaushik

(Recursion + Combinatorics least solved Assignment problems)



Encryption

'abc' → +2 → cde → -2 → 'abc'

msg — encrypted. — decrypted msg.

ASCII

a: 97 +2 ⇒ 99 ⇒ c

b: 98 +2 ⇒ 100 ⇒ d

c: 99 +2 ⇒ 101 ⇒ e

secret

Example 2

Students example

→ Code 1 → O(N) T.C.

→ Code 2 → O(1) T.C

Dict => How?

Quiz - ?

```
d = {
      5 : 'Python',
      5.0 : 'CPP'
    }

d[5] => 'CPP'
```

} Hashing.

Hash Function

Let's say you own a hotel

↳ Receptionist

Room No: 'B12'

Room No: 'C14'

Room No: 'D101'

Register

C14	→	✓
D101	→	✗
A22	→	✗
B16	→	✓
B12	→	✓

TC: O(N)

A27 → ✓

B12 → X

B16 → X

C14 → ✓

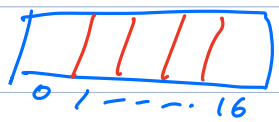
D101 → ✓

TC: $O(\log_2 N)$

'B12' => 66 + 1 + 2 = 69 %17 => 1 => X

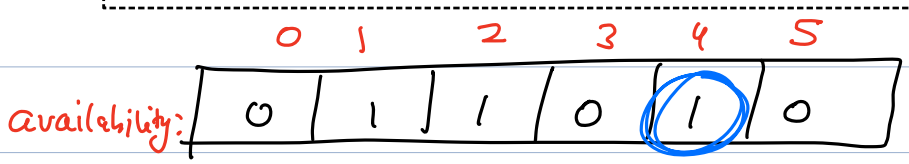
'C14' => 67 + 1 + 4 = 72 %17 => 4 => ✓

'D101' => 68 + 1 + 0 + 1 = 70 %17 => 2 => X



17 rooms.

```
def hash_f(room_no):
    # add ASCII value of char
    # add num as it is
    return (sum % 17)
```



availability[4] => O(1)

ArrayList access is O(1) at Given index

=> l[idx]

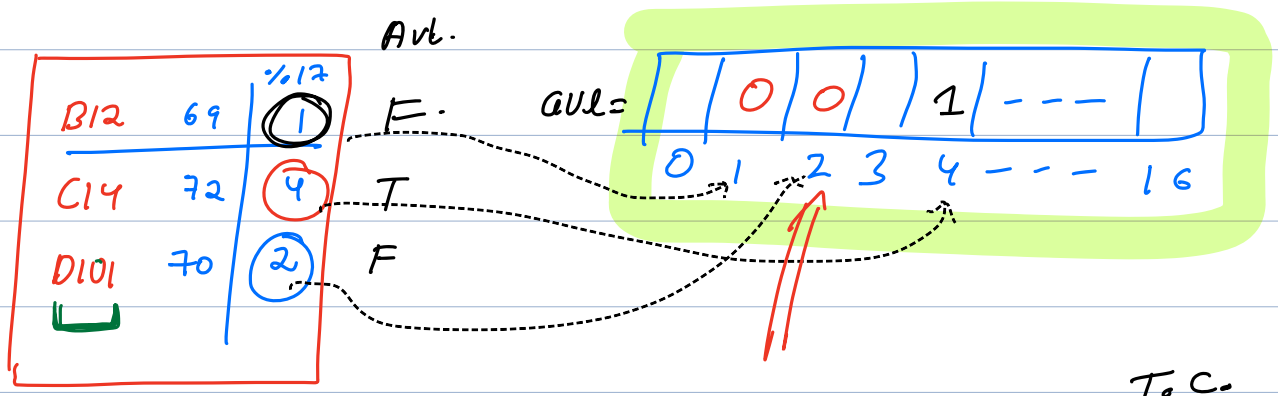
→ Index

Chapter_name	Pg No
History: 1	5
Civics: 1	15
Geo: 1	27

=> mapping a chapter name to a page num.

=> mapping the room num to an index

Hash Table M = 17



D101 avl or not

‖

To C. => O(1)

To C.

① Get the hash value => O(1)

hv = hash_f("D101") # 2

② Access the 2nd index in avl => O(1)

avl[hv]

d = { 'B12': 0, 'C14': 1, 'D101': 0 }

'abc123' → a$#99bcd@

'abc12' → a$#99bcd@

hash value

l = [1, 2, 3]

l.append(4)

l = [1, 2, 3, 4]

hash value

pqr2a978#@

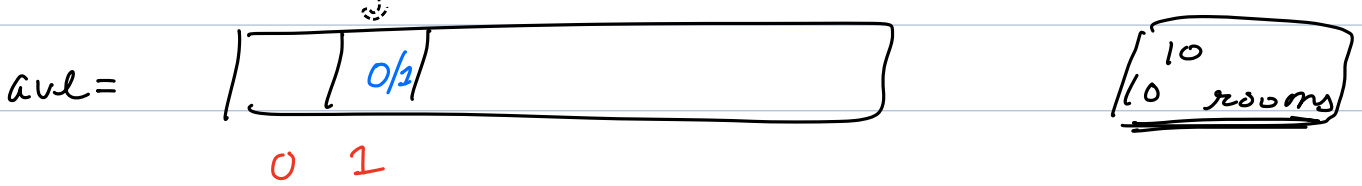
Mutable data structures are not possible to be used as keys in dict/set => not hashable

Because their content is not fixed.

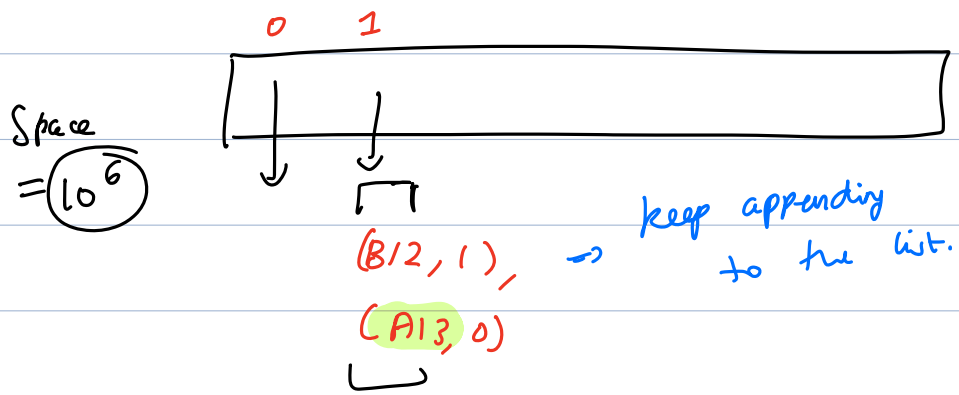
---

1 ← B12 ⇒ 66+1+2 = 69 %17 = 1

0 ← A13 ⇒ 65+1+3 = 69 %17 = 1

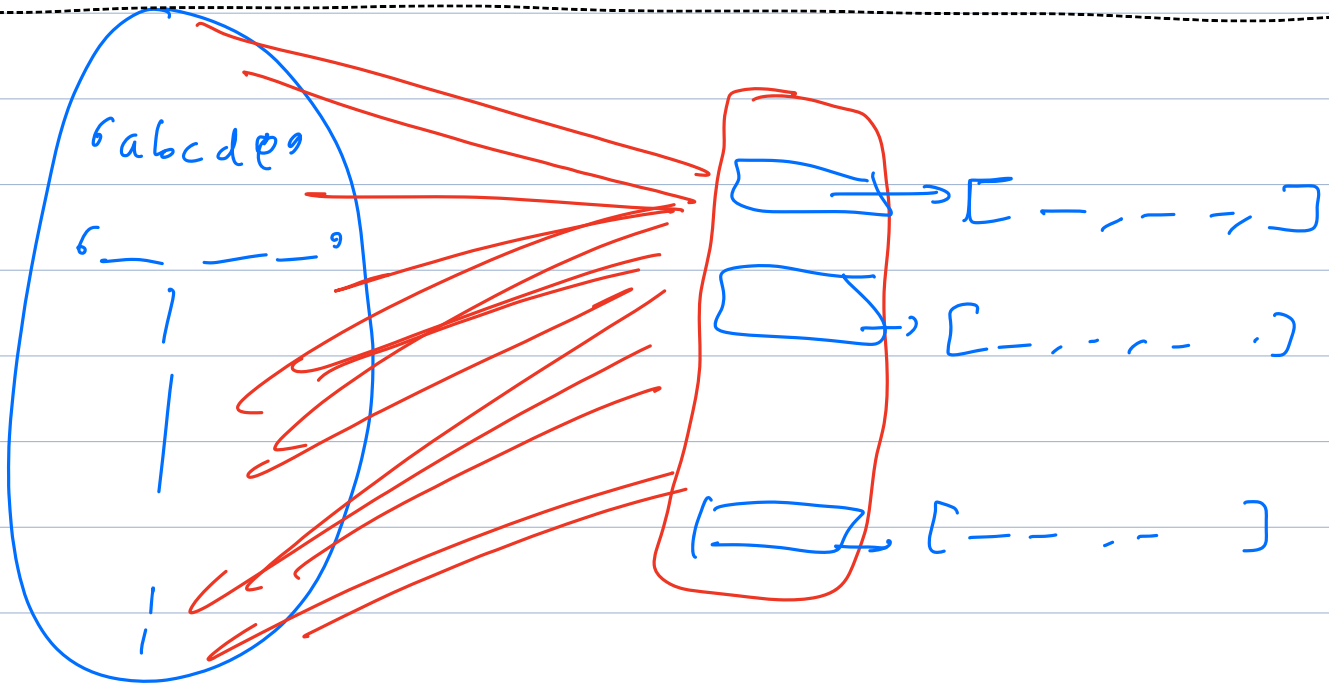


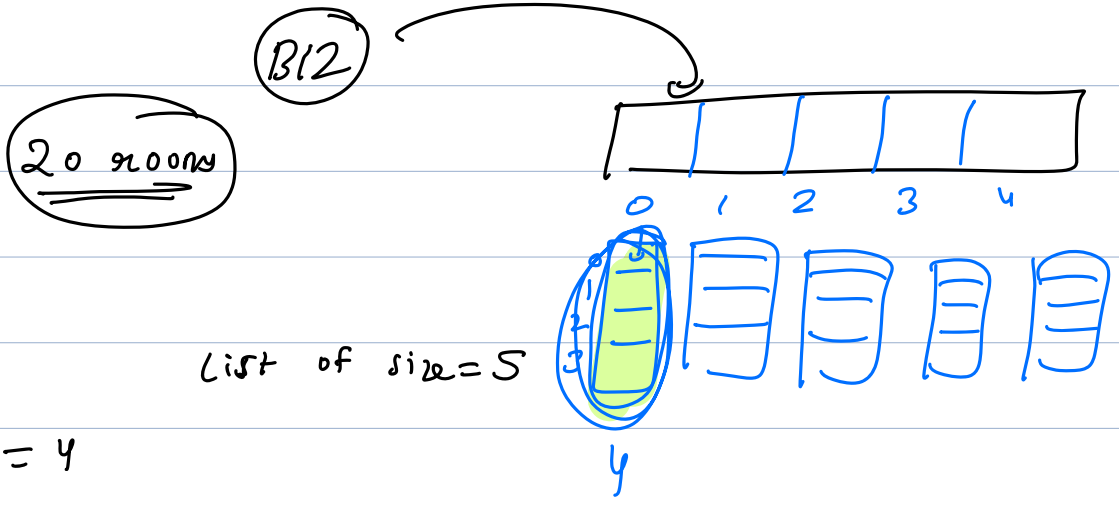
nested list ??



Separate Chaining.

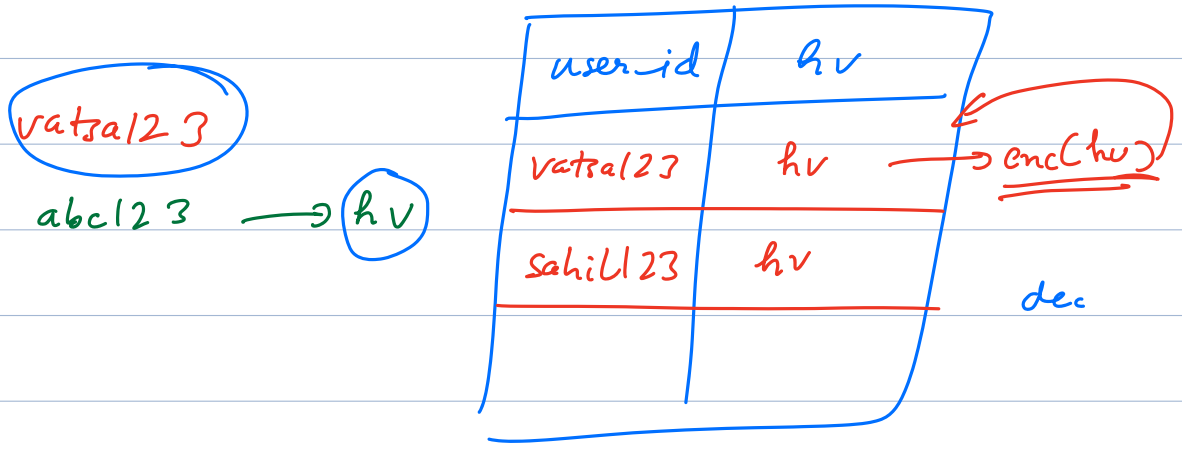
$$\frac{10^{10}}{10^6} = 10^4$$





$\frac{20}{5} = 4$

Doubts



enc(hash(pass))

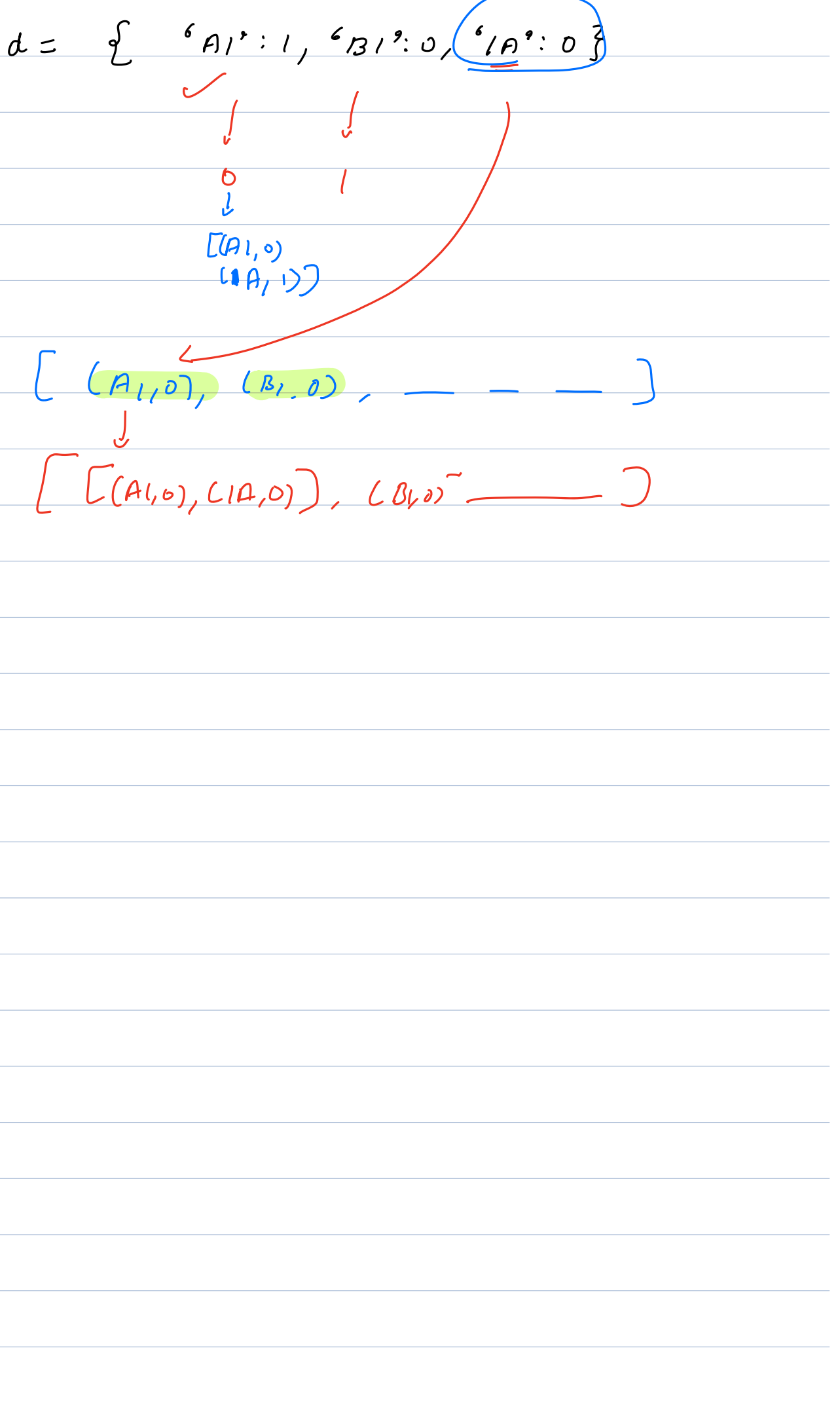
X % 17

X = 51 → 0

X = 68 → 0

X = 34 → 0

TC: O(1)


d = { 'A1': 1, 'B1': 0, '1A': 0 }
0
1
[(A1,0)
(A,1)]
[(A1,0), (B1,0), — — —]
[[(A1,0), (1A,0)], (B1,0) — — —]