Introduction to HashMap

Sunday, 27 June 2021 2:48 PM

data Structure →

1. Arrays
2. Array list
3. Stack] → array with discipline] add remove
4. Queue]
5. Linked List
6. Trees → G.T. / → B-T / → B.ST.

find → 10] → O(n) Linear
└ sorted - O(log n)

Main function of D-S →
- → add
- → remove
- → get
- → set
- → size
- → display

get] → Index, on the basis of

value basis

get → O(n)

Usage of HashMap →

(String) Keys		(Integer) Values
India	→	~~150~~ 160
Pak	→	90
China	→	130
UK	→	100

- ✓ (India → 150)
- ✓ (pak → 90)
- ✓ (China → 130)
- ✓ (UK → 100)
- ✓ (India → 160)

HashMap.

Time.

O(1) (1) put (key, value)
- → Already present → Update Value.
- → Absent → Insertion of key, value pair

O(1) (2) remove (key)
- → Already present → Remove from Hashmap and return value;
- → Absent → return null;

O(1) (3) containsKey (key) (presence of key in Hashmap)
- → present → True
- → Absent → False.

O(1) (4) get (key)
- → key is present → value return.
- → key is Absent → null return.

O(n) (5) KeySet () → Return all the keys in Set
{ "India", "pak", "China", "UK", "Ugenda", "Nigeria" }
order of keys are random

Key (String)	Value (Integer)
"India"	~~150~~ 160
~~"pak"~~	100
"China"	120
"UK"	130
"Ugenda"	70
"Nigeria"	30

NOTE: keys are uniquely stored in the HashMap.

Highest Frequency Character

Sunday, 27 June 2021 2:49 PM

String → "$\overset{0}{a}$ $\overset{1}{a}$ b c d b c d e f e b d h" d d"

highest freq. character??

HashMap < Character, Integer> map.

array → map

26 → array – mapping

a → ~~1~~ 2

b → ~~1~~ ~~2~~ 3

c → ~~1~~ 2

d → ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5

e → ~~1~~ 2

f → 1

h → 1

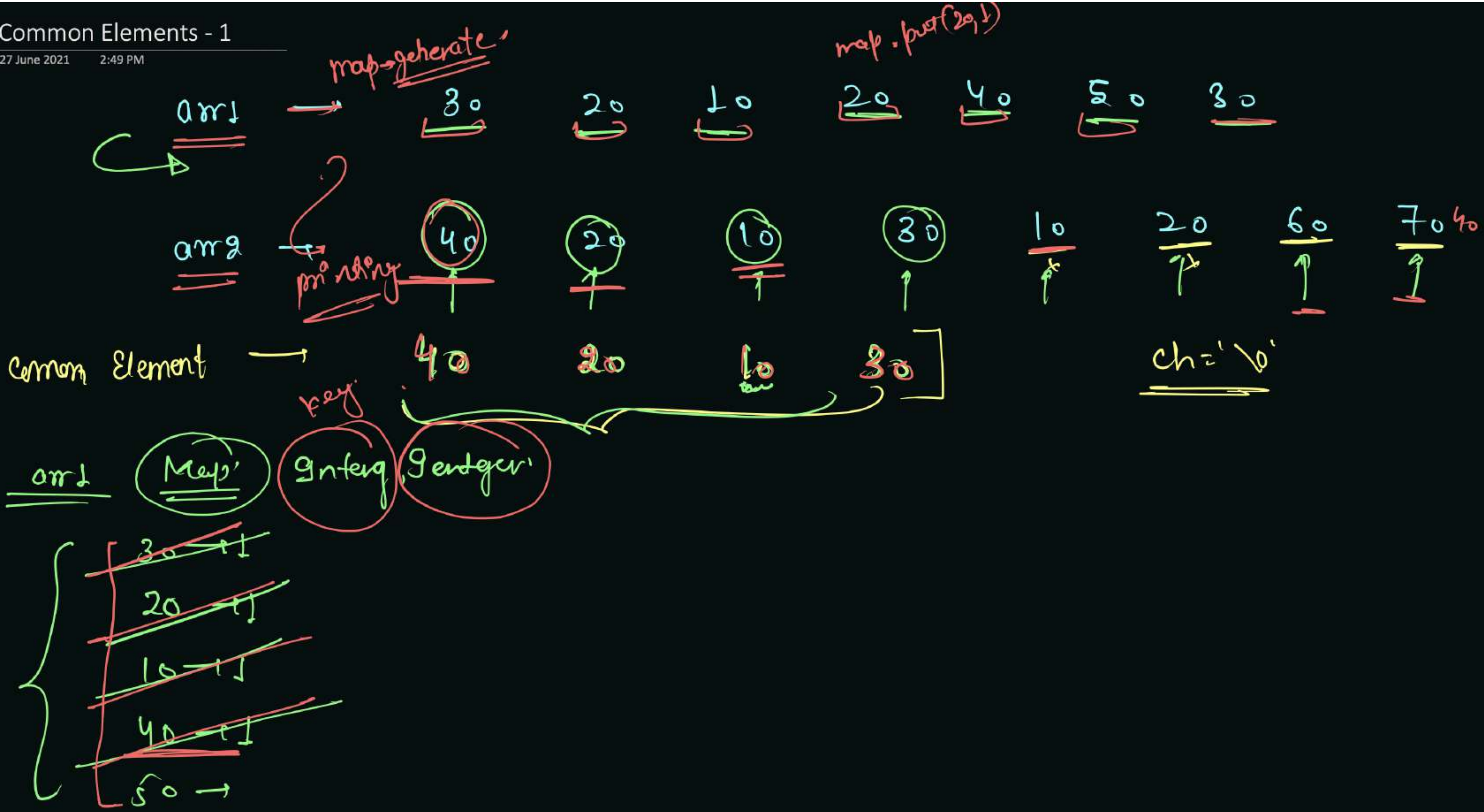
KeySet (a, b, c, d, e, f, h)

freq. ← map.get( )

maximise

Get Common Elements - 1

Sunday, 27 June 2021 2:49 PM


map→generate'
map.put(20,1)
arr1 → 30 20 10 20 40 50 30
arr2 → 40 20 10 30 10 20 60 70 40
printing
Common Element → 40 20 10 30
ch='\b'
key
arr1 Map' Integer, Integer'
30 → 1
20 → 1
10 → 1
40 → 1
50 →

Get Common Elements - 2

Sunday, 27 June 2021 2:50 PM

freq. map

arr1 → 20 30 40 20 10 30 50 40 70 10

arr2 → 40 70 10 30 10 8 12 20 50 10

Inter section

Integer, Integer

20 → ~~1~~ ~~2~~ 1

30 → ~~1~~ ~~2~~ 1

40 → ~~1~~ ~~2~~ 1

10 → ~~1~~ ~~2~~ ~~1~~ 0

50 → ~~1~~ 0

70 → ~~1~~ 0

40 70 10 30 10 20 50

40 70 10 30 10 20 50

Longest Consecutive Sequence Of Elements

Sunday, 27 June 2021 2:50 PM

arr → 22 23 1 12 2 10 3 11 16 13 21 15 14

Sequence , which is consecutive →

→ C++ python Integer vs Boo

22 →

Ex → 1 . 2 → 3

→ 10 - 11 - 12 - 13 - 14 - 15 - 16

← 21 - 22 - 23 - ~~24 - 25 - 26 - 27~~

10
11
12
13
14
15
16

←

Longest consecutive sequence → 10 - 11 - 12 - 13 - 14 - 15 - 16

22 23 1 12 2 10 3 11 16 13 21 15 14

HashMap <Integer, Boolean> , Key - True.

→ 22 → ~~True~~ - False

→ 23 — ~~T~~ False.

→ 1 → True }

→ 12 → ~~T~~ False

→ ✓2 → ~~T~~ False

→ 10 → True }

→ ✓3 → ~~T~~ False

→ ✓11 → ~~T~~ False.

→ ✓16 → ~~T~~ False

→ ✓13 → ~~T~~ False.

→ ✓21 → True }

→ ✓15 → ~~T~~ False.

→ ✓14 → ~~T~~ False.

n ① Fill the Hashmap with de Vs - True.

+

n ② Trace starting pt. of sequence in visible manner.

+

2n ③ Starting from starting point of seq. until consecutive is present and maximise the seq.

Total - 4n

≅ O(n)

10

st = ~~0~~ ~~1~~   leng = ~~0~~ ~~3~~ 7

22 → ~~True~~ - False

~~28~~ → ~~T~~ False.

1 → True}

12 → ~~T~~ False

2 → ~~T~~ False

10 → True}

3 → ~~T~~ False

11 → ~~T~~ False.

16 → ~~T~~ False

13 → ~~T~~ False.

21 → True}

15 → ~~T~~ False

14 → ~~T~~ False.

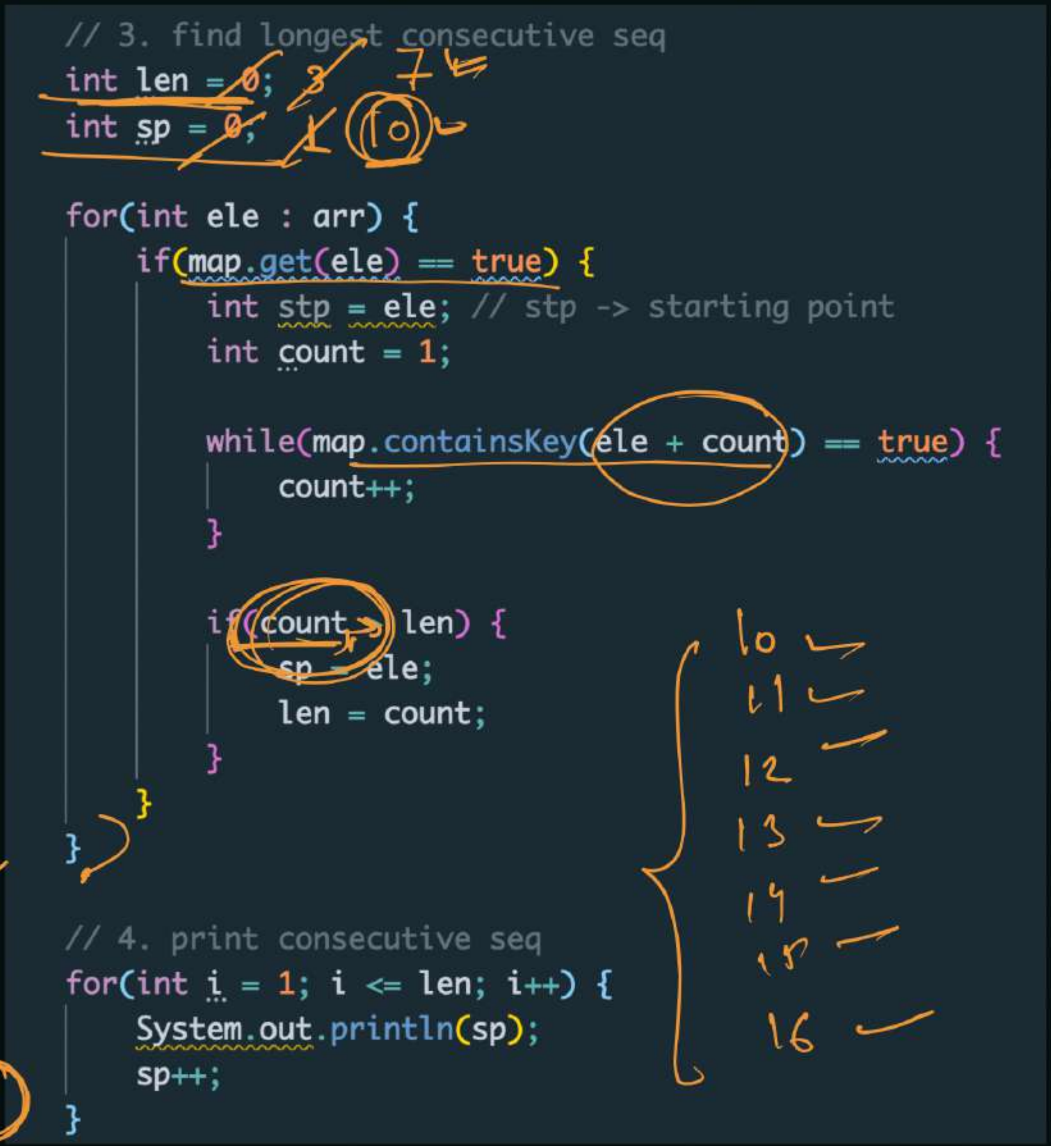
elem = ~~10~~ 21

Cocnt = ~~1~~ ~~2~~ 3

~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~

7

~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ 7



```
// 3. find longest consecutive seq
int len = 0;
int sp = 0;

for(int ele : arr) {
    if(map.get(ele) == true) {
        int stp = ele; // stp -> starting point
        int count = 1;

        while(map.containsKey(ele + count) == true) {
            count++;
        }

        if(count > len) {
            sp = ele;
            len = count;
        }
    }
}

// 4. print consecutive seq
for(int i = 1; i <= len; i++) {
    System.out.println(sp);
    sp++;
}
```

Introduction to Heap

Sunday, 27 June 2021 5:41 PM

(Heap sort) → nlogn

working → x

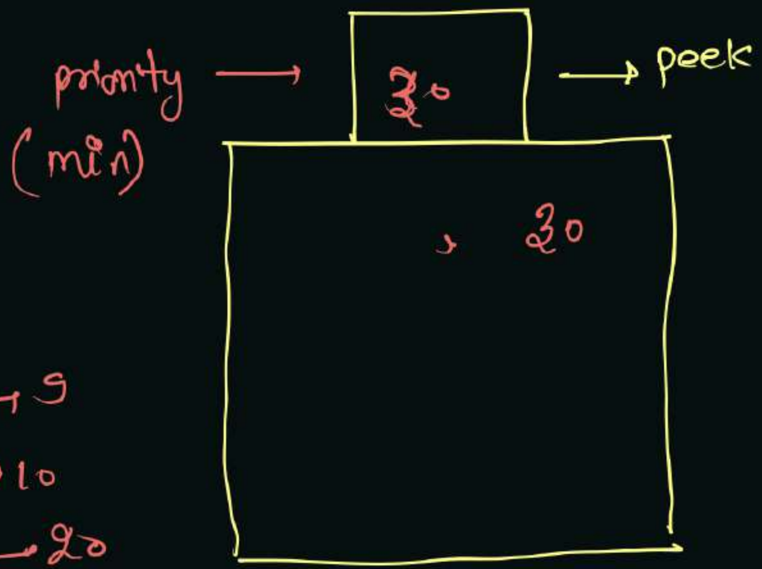
usage] ✓

As a user, our concern is for peek

priority → Min

Max

(values/ data)



P.Q. Container.

Java default priority Queue → (Min Priority)

functions in P.Q.

1. add → O(logn)
2. remove → peek)x O(log n)
3. peek → O(1)
4. size → O(1)
5. Syso( )

Random.

add(10) ✓]
add(20) ✓] k
add(30) ✓]
add(9) ✓
add(7) ✓
remove( ) ✓ → 7
add(3) ✓
peek() ✓ → 3
remove →
peek() → 9

✓ remove() → 9
✓ remove() → 10
✓ remove() → 20
✓ peek() → 30
✓ add(10) →
✓ add(10)
→ peek() → 10
→ remove() → 10
remove() → 10