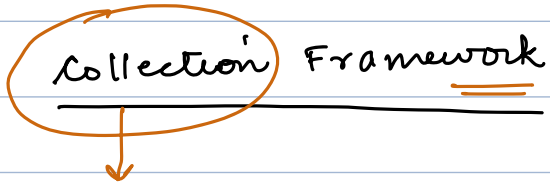
Recording ✗

→ { Notes ✓
Code ✓

Agenda :

Collection Framework
Java

comparable vs
comparators

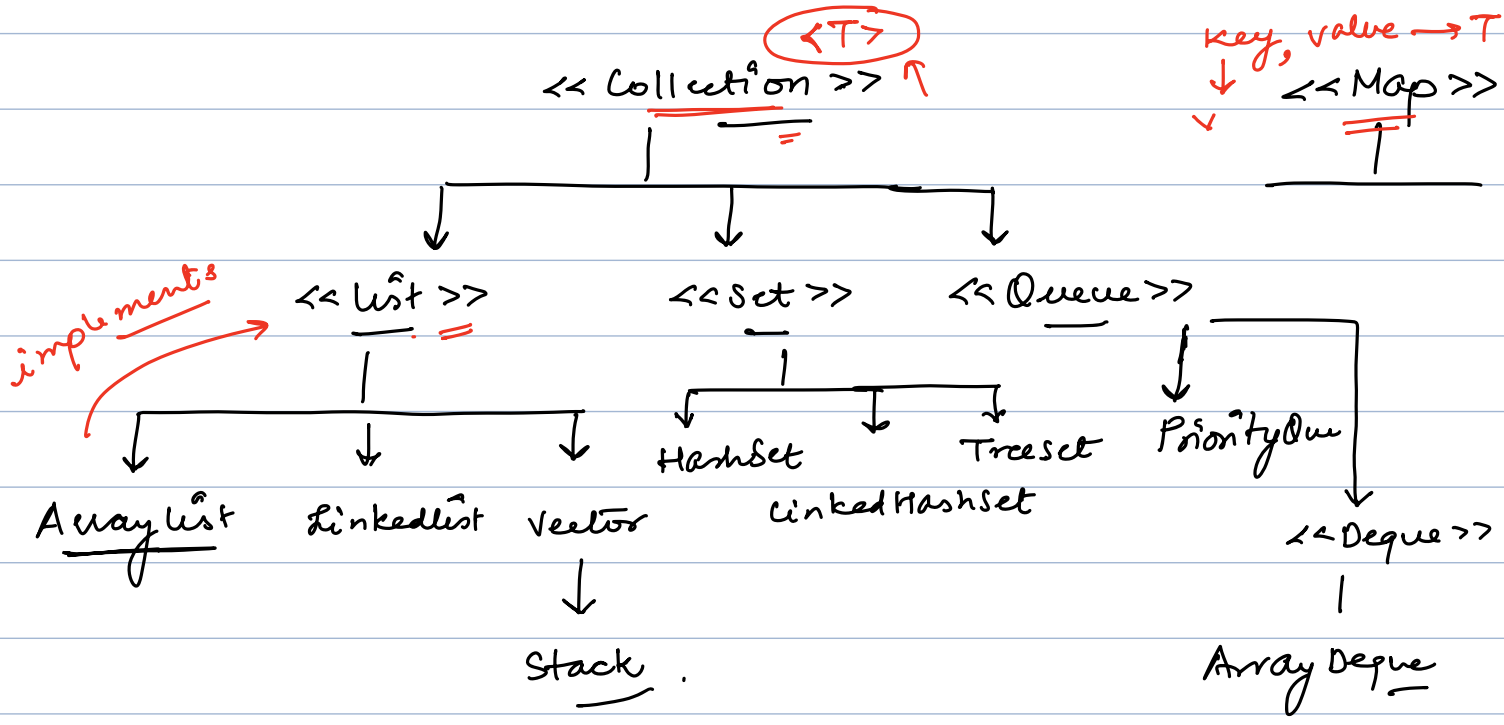


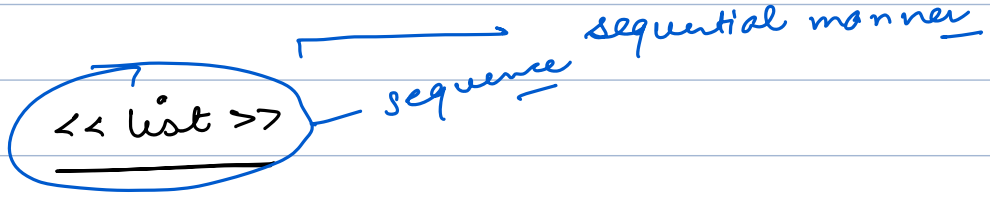
C++
↓
STL

container ← store data

1) Reduces programming effort ✓

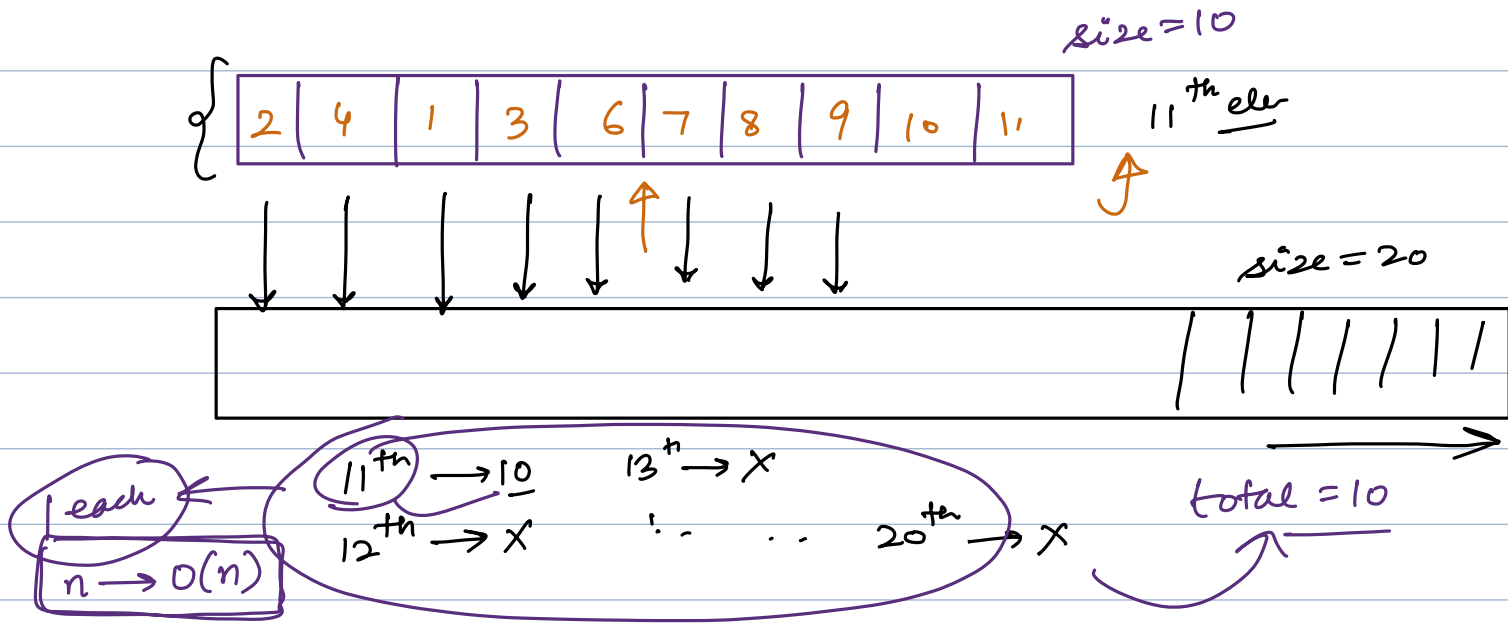
2) Increase program speed & quality





An ordered collection (also known as a *sequence*). The user of this interface has precise control over where in the list each element is inserted. The user can access elements by their integer index (position in the list), and search for elements in the list.

~~sync~~ ArrayList : dynamic array



n = 4

| 4 | 1 | 2 | 3 |

↓ ↓ ↓ ↓

1 1 1 1 = 4 iteration

5^{th} element

n=8

| 4 | 1 | 2 | 3 | 6 | 9 | 8 | 7 |

4 extra to copy

insert 4 new
4 copy
1 elem – 1 copy

N → O(N)

Ro 6000 → 30 days   200 – Bisleri

$\frac{6000}{30} = 200$

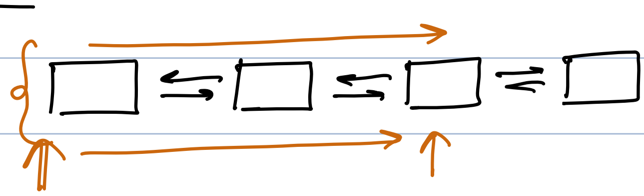
array vs linkedlist

arr[5] → ~~O(1)~~

2-3

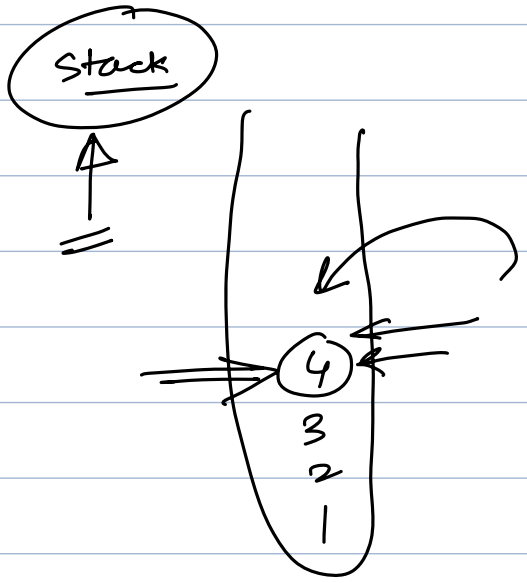
add^n & deletions

~~sync~~ linkedlist



LRU cache

Vector : synchronized list



LIFO

push

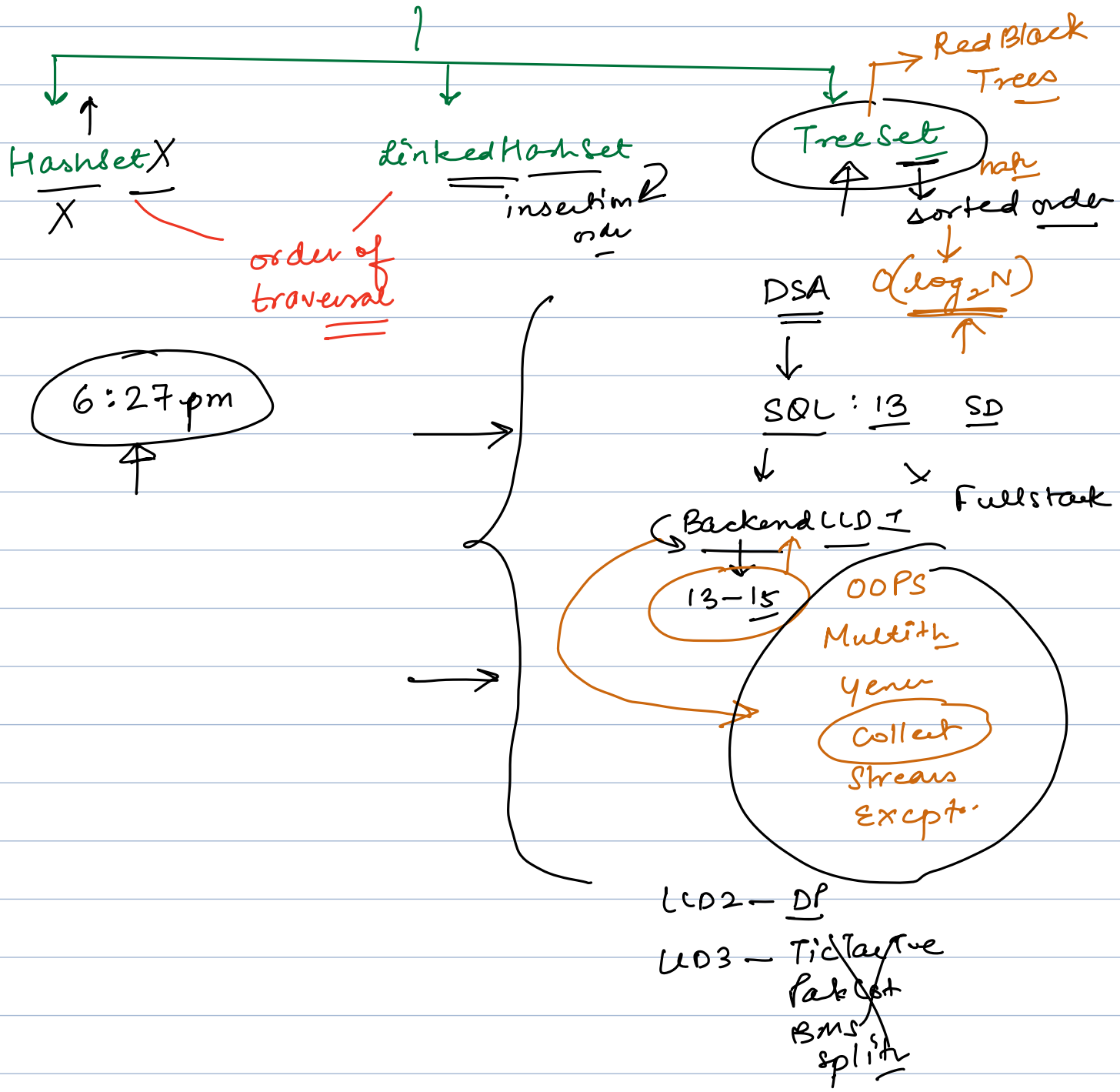
pop

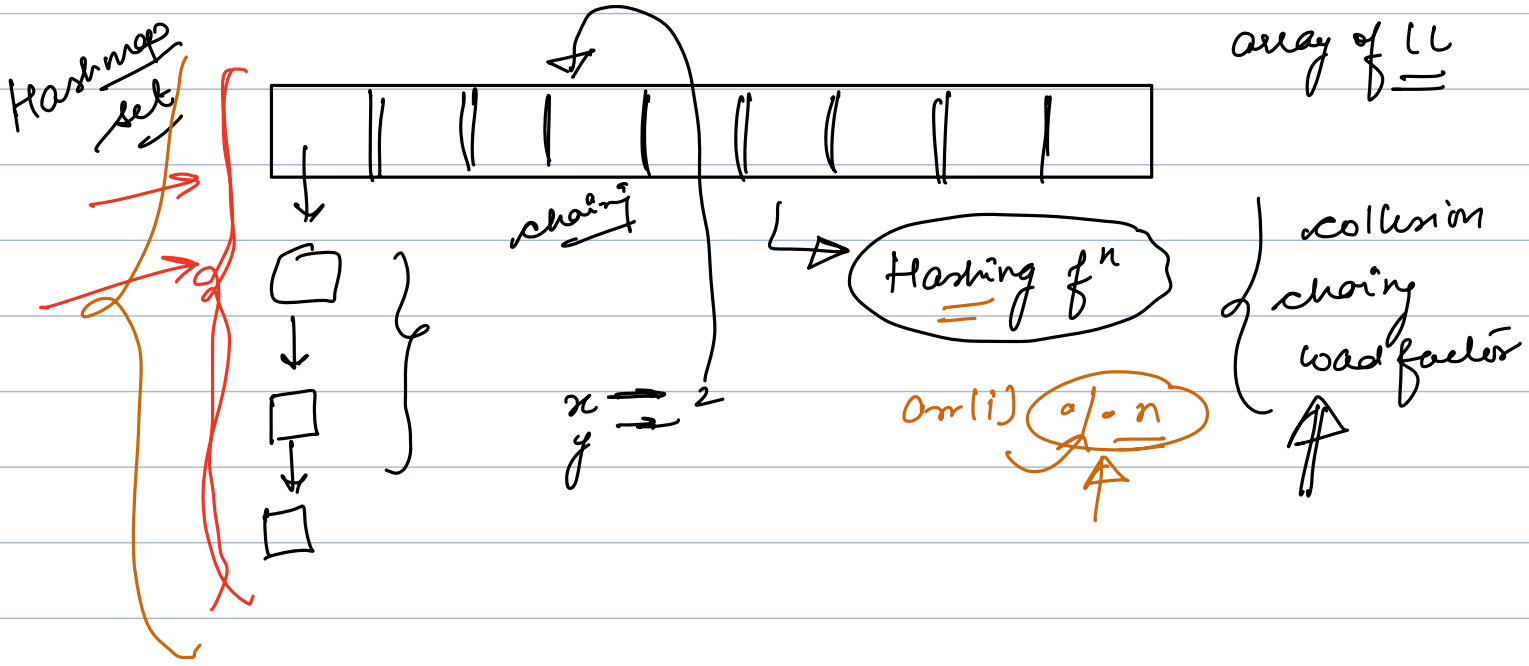
peek

Collection · synchronizedList

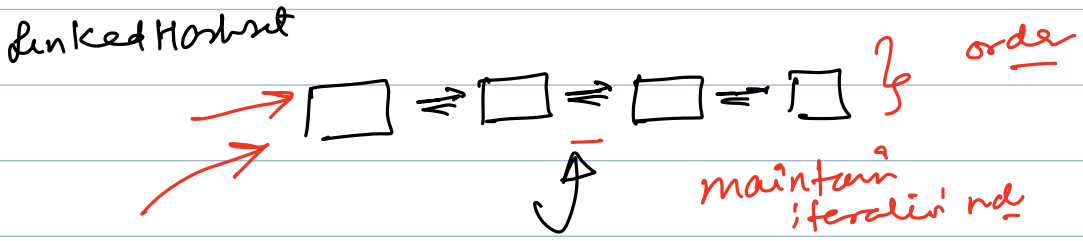
<< Set >> — search — O(1) avg
— ordering

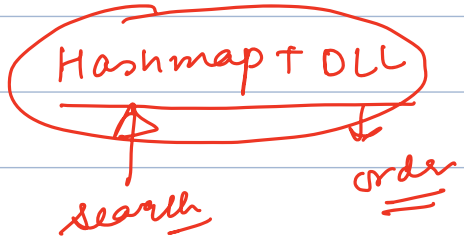
A collection that contains no duplicate elements. More formally, sets contain no pair of elements e1 and e2 such that `e1.equals(e2)`, and at most one null element. As implied by its name, this interface models the mathematical *set* abstraction.




Hashmap set
array of LL
chaining
Hashing fn
collision
chaining
load factor
x = y = 2
O(1)
a/n

+


LinkedHashset
order
maintain insertion order


Hashmap + DLL
search
order

2	1	3	4	7	8	6	5	15

8 exists?

search - iterate - O(N)

0	1	2	3	4	5	6	7
0	9	74	83	12	5	166	15

collision

6 %8 = 6

166 — Hash fn

Hash fn

x %8 → 6 index

12 %8 = 4

15 %8 = 7

9 %8 = 1

8 %8 =

74 %8 =

166

83 %

5 %8 = 5

102 %8 = 6th

83 %8 = 3 → O(1)

O(1)

10 n

Ha → 3 class
