=> Accessspecifiers :- { public, protected, default, private

=> Package :-

⇓

Folder

⇓

files

Projects :- 100 files, 200 files

500 files

=> Banking Project => Register login

≡ ≡ ≡

=> weblayer service DAO

JavaProject

↳ weblayer ↳ folder, → files ↳ rol → files

↳ service

↳ DAO

{ lang util sql net

⇓

package

java

↳ javalang → ---- string,

↳ util → Scanner, Array -----

↳ sql

↳ net

≡

import java.util.Scanner;

→ com.telusko.login

↳ ≡ 2 files

→ com.telusko.register

=> ≡ 3 files

→ com.telusko.Service

| =>

main

import com.telusko.register.*;

≡

main

≡

public ✓
protected ✓
default ✓
private ✓
} ✓

package ✓

---

variable => Data to be stored

{ int age=18; }

Array :- large Data => similar) Homogeneus

↳ Limitations {
=> only Homogneus
=> fixed size (cannot grow/shrink)
=> contigous location.
=> Arrays -> utility

Legacy.
=> Vector
Dictionary
Stack }

open Source

0-9 -> Alpha

Java 1.0

Java 1.2

Java 2

=> Joshua =>

collection
⇓
interfaces & 7 classes
⇑

Collection -> Java 2

{ ↳ set of interfaces & classes & each class with specific Data structure & algorithm to handle & manage large volume of Data both homogenus & heterognis

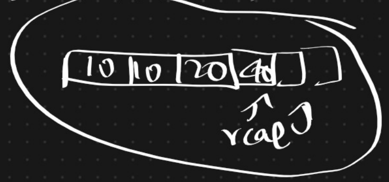
Arraylist => DS -> Dynamic Array DS

↳ size can grow or shrink Dynamically

↳ can store Both homognis & heterognus type Data

↳ Data is stored as object.

list.add(40);



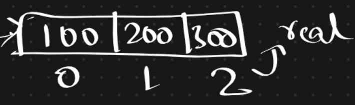
-> implement List interface.

-> indexed Based access is allowed.

al.add(100);

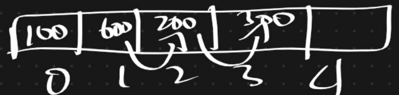
al.add(200);

al.add(300);

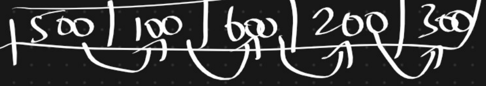
al ->


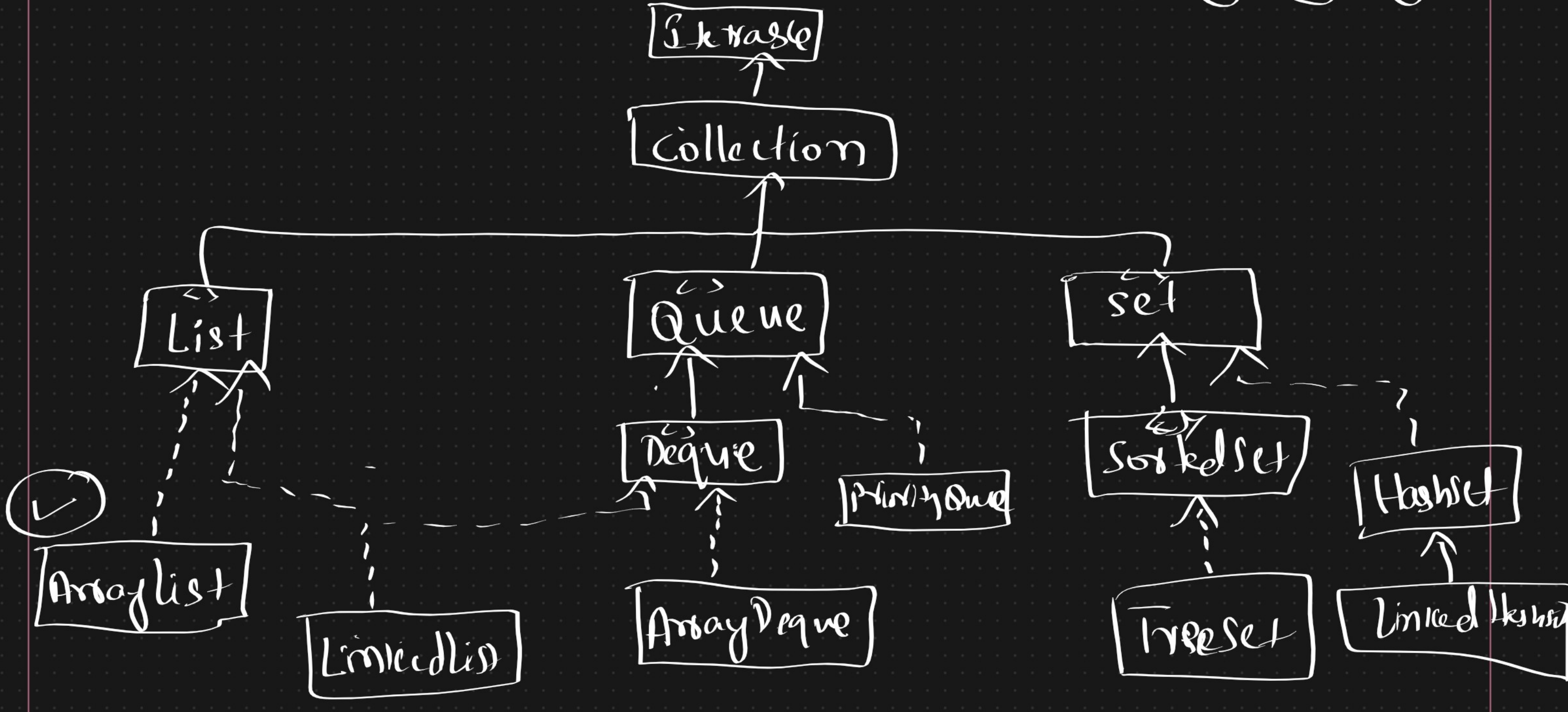
al.add(1,600);

al.add(0,500);



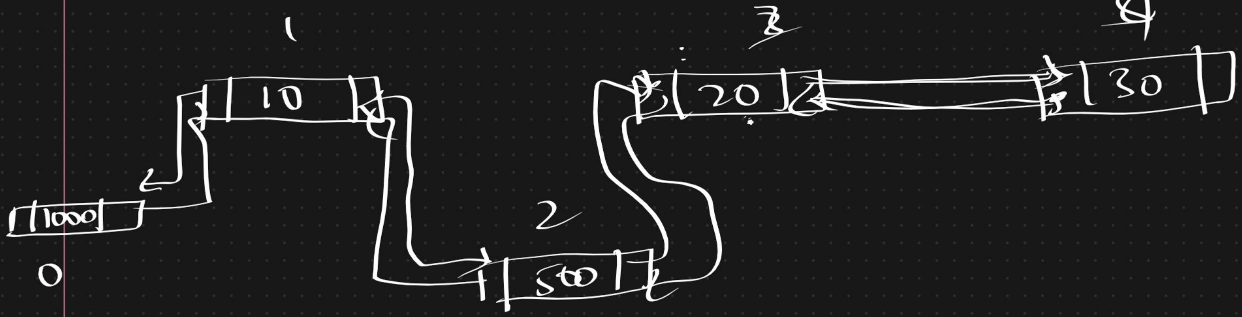
0.1 Ms



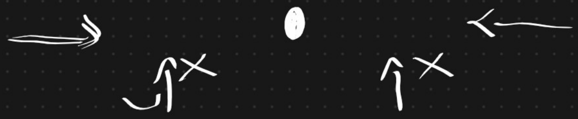


DoublyLinkedlist :-

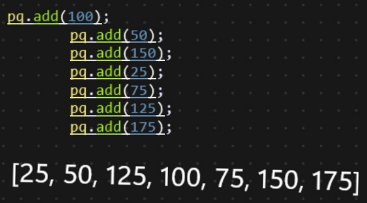
ll.add(2, 500);



ArrayDeque :- (Double ended Queue)



=> rear end & frontend |

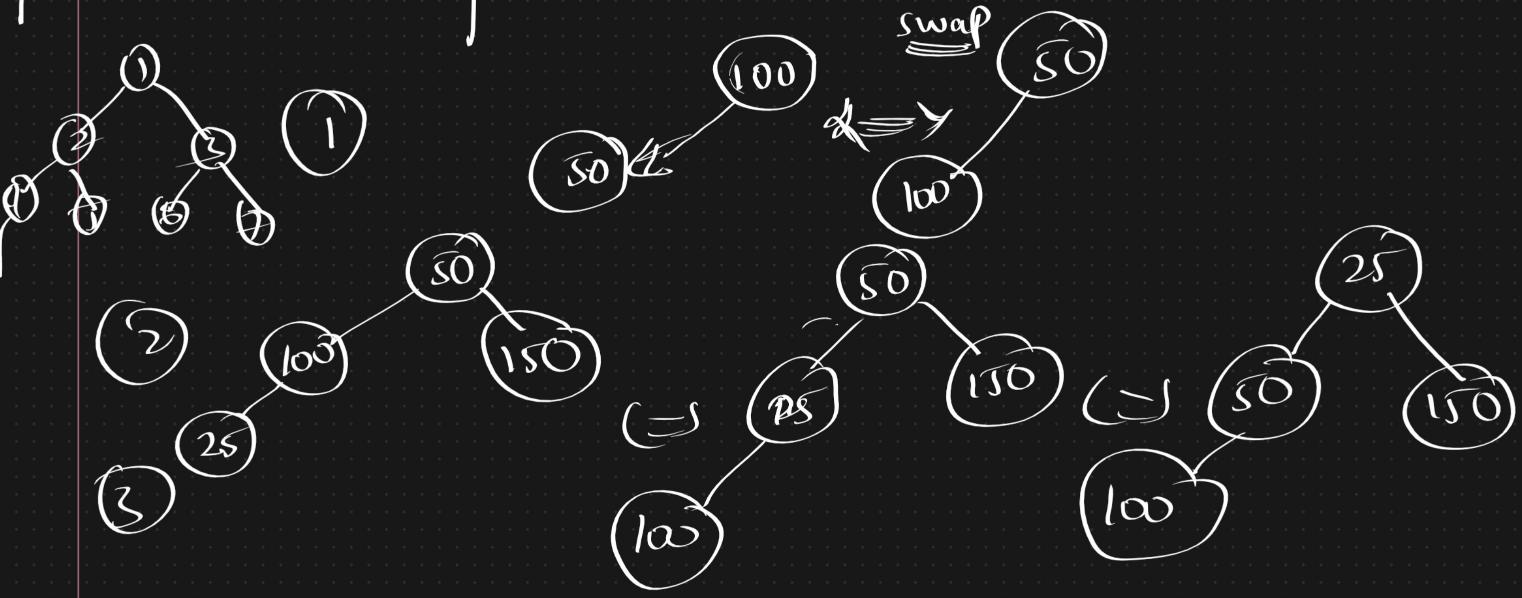


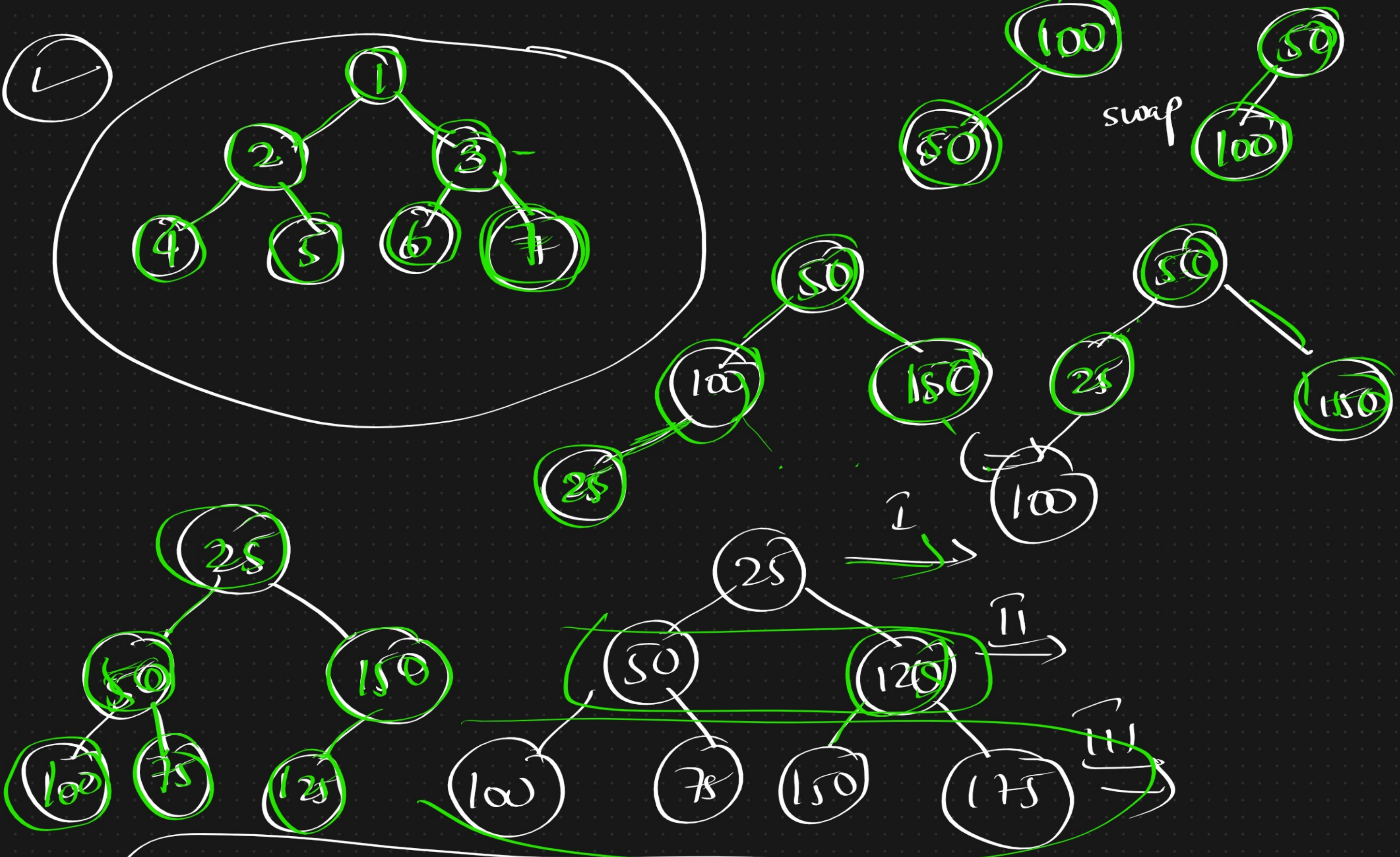
```
pq.add(100);
        pq.add(50);
        pq.add(150);
        pq.add(25);
        pq.add(75);
        pq.add(125);
        pq.add(175);
```

```
[25, 50, 125, 100, 75, 150, 175]
```

100 50 150 ~~25~~ 75 125 175

min-Heap :-





25 50 120 100 75 150 125