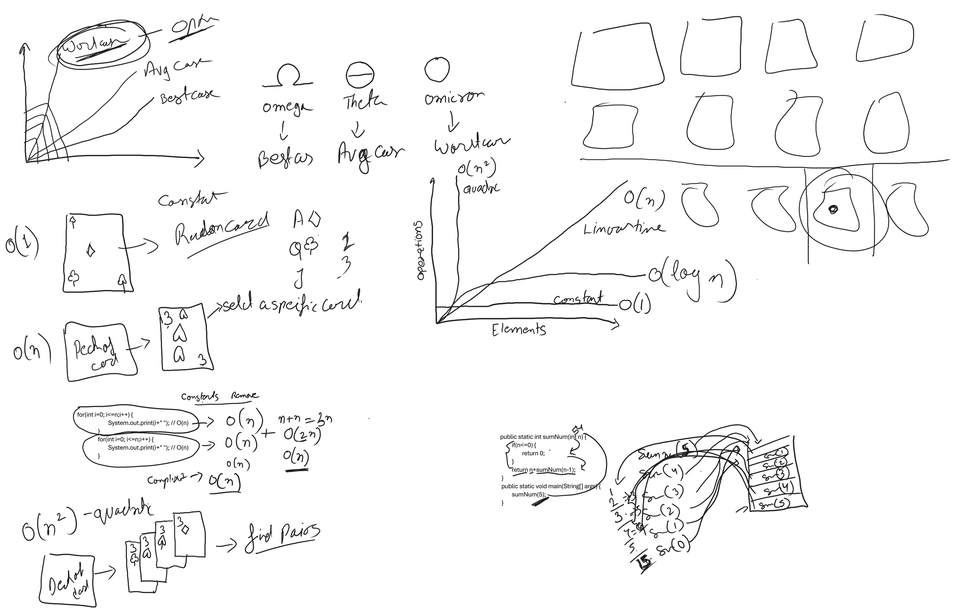
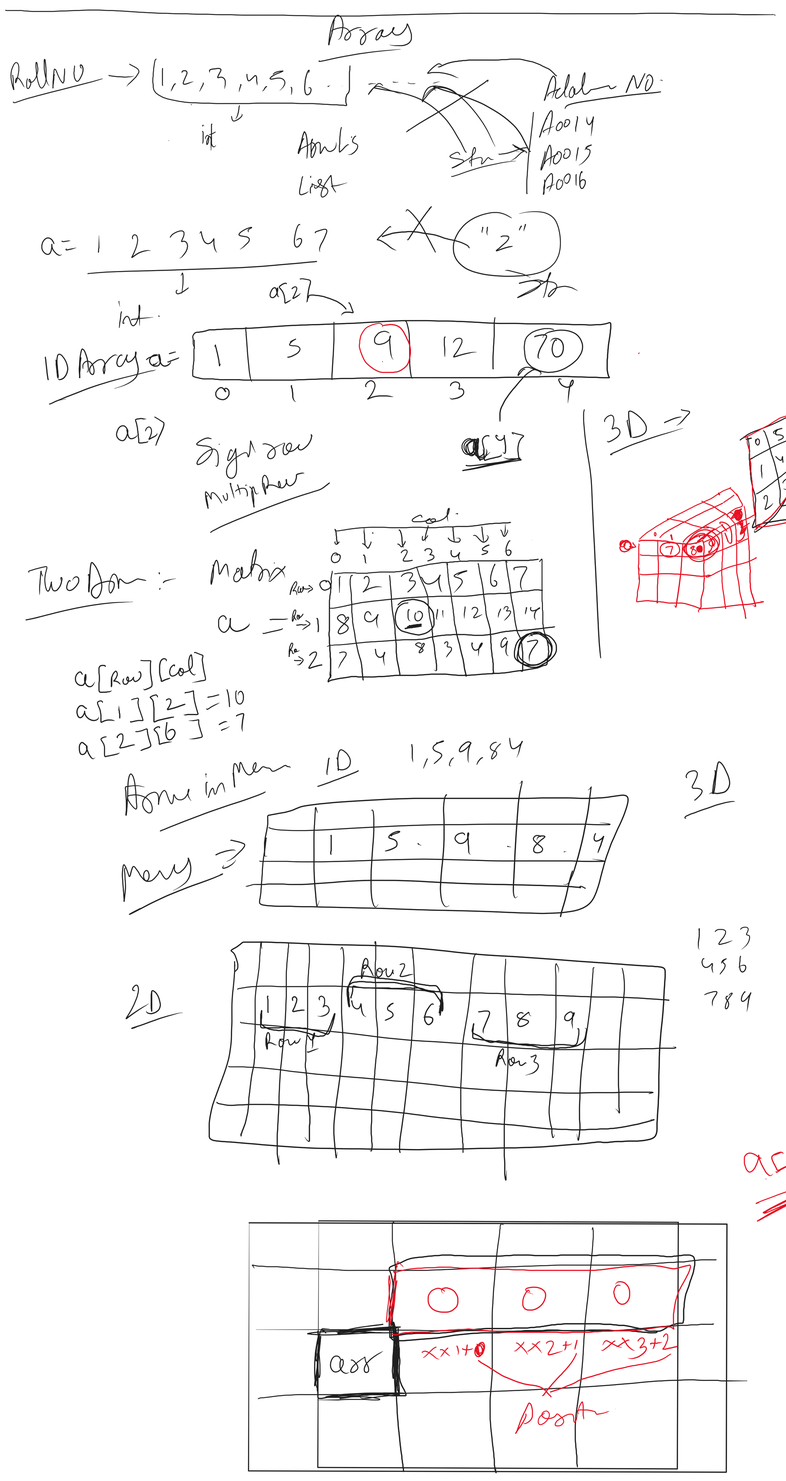
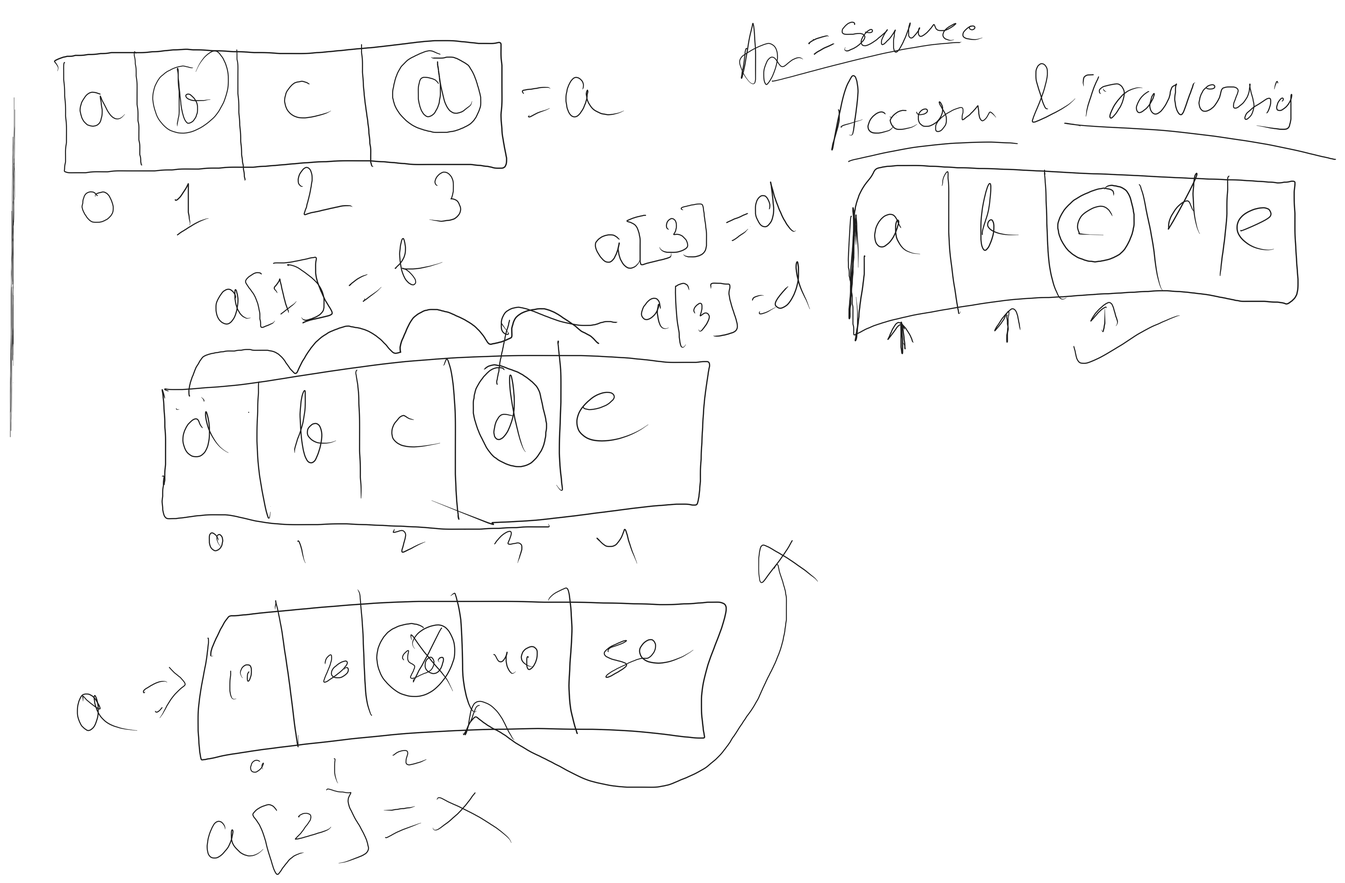

Worst case
Avg case
Best case
omega
Theta
omicron
Best case
Avg case
Worst case
O(n²)
quadratic
O(n)
Linear time
O(log n)
constant
O(1)
operations
Elements
Constant
O(1)
Random card
A ♢
Q ♤
J
1
3
O(n)
Perfect card
3 ♢
A
Q
3
select a specific card
Constants Remove
for(int i=0; i<n;i++){
System.out.print(i+" "); // O(n)
}
for(int i=0; i<n;i++){
System.out.print(i+" "); // O(n)
}
O(n)
O(n)
O(n)
n+n = 2n
O(2n)
O(n)
Complint → O(n)
O(n²) - quadratic
Deck of card
find Pairs
public static int sumNum(int n){
if(n==0) {
return 0;
}
return n+sumNum(n-1);
}
public static void main(String[] args){
sumNum(5);
}
Stack
sum(5)
sum(4)
sum(3)
sum(2)
sum(1)
sum(0)


Array
RollNO → 1,2,3,4,5,6
int
Array List
Actual NO.
A0014
A0015
A0016
a = 1 2 3 4 5 6 7
"2"
int
a[2]
1D Array a =
1 5 9 12 70
0 1 2 3 4
a[2]
a[4]
Single row Multiplier
3D →
a=[Row][col][Depth]
a=[0][0][1]
a=[2][0][2]
Two Dim :- Matrix
col
0 1 2 3 4 5 6
Row 0 1 2 3 4 5 6 7
1 8 9 10 11 12 13 14
2 7 4 8 3 4 9 7
a[Row][col]
a[1][2] = 10
a[2][6] = 7
Array in Mem
1D
1,5,9,8,4
Mem →
1 5 9 8 4
2D
Row1
Row2
Row3
1 2 3 4 5 6 7 8 9
1 2 3
4 5 6
7 8 9
3D
0 1 2
3 4 5
6 7 8
9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
arr
xx1+0
xx2+1
xx3+2
Depth
a[0]=e
a = a b c d
0 1 2 3 4
a[2] = "c"
a[3] = "d"
a[5]
e a b c d
0 1 2 3 4
a[1] = f
if Arr full
Create New Array
e a b c d
e f a b c d


a b c d = a
0 1 2 3
Arr = Sequence
Access & Traversing
a[1] = f
a[3] = d
a[3] = d
a b c d e
0 1 2 3 4
a b c d e
0 1 2 3 4
a →
10 20 30 40 50
0 1 2
a[2] = x