





→ Sunday (Java Memory Management)

- > Lot of Theory Concepts

~~8, 2, 5~~  
→ ~~Subset~~

- ↳ Stack
- ↳ Heap
- ↳ GC
- ↳ Pass By value

## Doubts

Q

$$A = \begin{bmatrix} \underline{2} & 5 & 8 & \underline{10} & \underline{13} & 11 \end{bmatrix}$$

Pair of elements

$$x + y = 15$$

exactly

2 elements

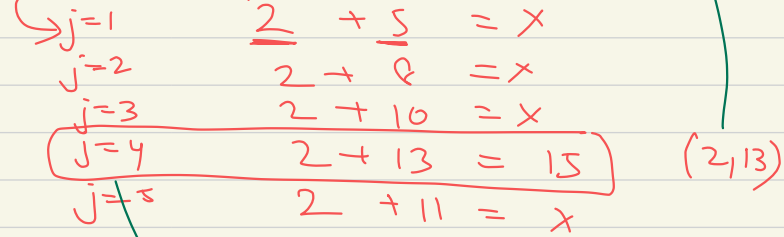
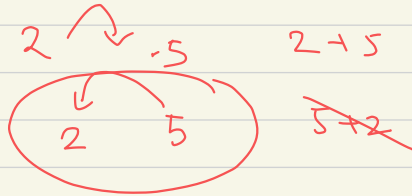
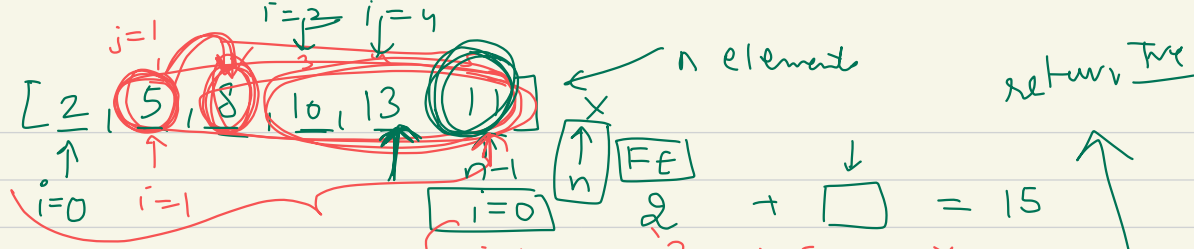
✓  
Should  
be in  
the array

$$2, 13 = 15$$
$$5, 10 = 15$$

> Yes

Print all  
pairs  
s.t

$$a[i] + a[j] = 15$$



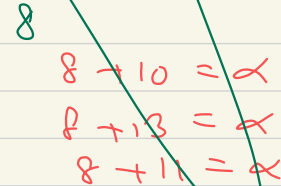
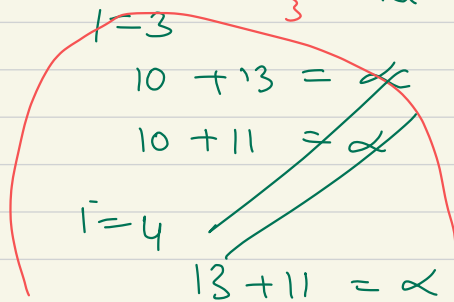
for ( $i=0$ ;  $i \leq n-2$ ;  $i++$ ) {

for ( $j=i+1$ ;  $j \leq n-1$ ;  $j++$ ) {

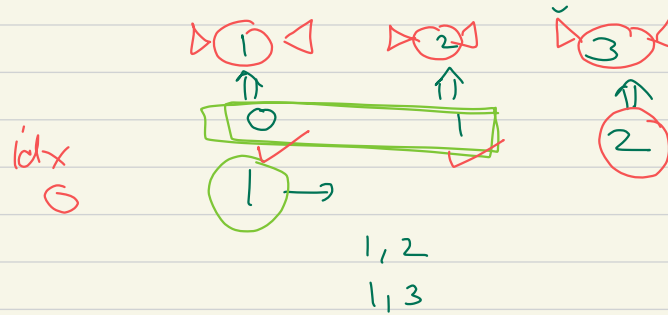
if ( $a[i] + a[j] == \text{sum}$ ) {

return true;

}



2 Sweets



N=3

for ( $i=0$ ;  $i < \underline{N-2}$ ;

1      2 →      2, 3      ( $j = i+1, \text{ --- } ; N-1$ )

Code

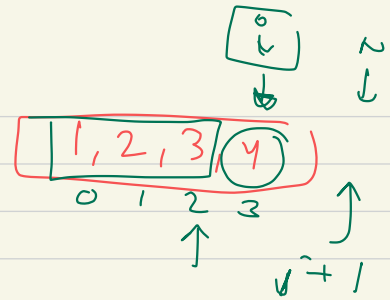
$i = n - 1$   $\nearrow$  arr.size  
`for (i = 0;  $i < n$ ; i++) {`

`for ( $j = i + 1$ ,  $j < n$ ; j++) {`

X Run for  $i = n - 1$

if ( $a[i]$  +  $a[j]$  == sum) {  
    return True

~~$a.get(i) +$~~  }  
}



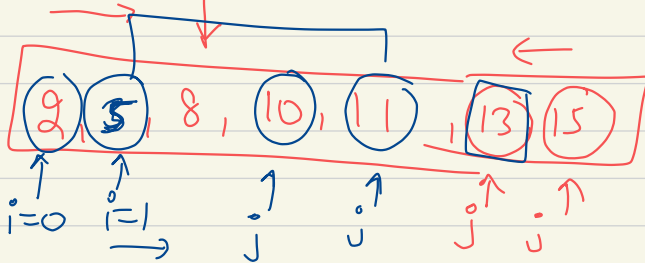
~~Sorting / hashmap~~

return True,

⇒ not-repeating  
Unique

2, 8, 5, 10, 13, 11, 15

SORT  
N log N



13  $\downarrow$  2    13  $\downarrow$  2

Two Pointer Trick

2, 13    (16) > 15  
          (15)  
[13] → [15]

while ( $i < j$ ) {

→ if ( $a[i] + a[j] > \text{sum}$ )  $j--$ ;

else if ( $a[i] + a[j] == \text{sum}$ )

print ( $a[i]$ ,  $a[j]$ );  $i++$ ,  $j--$

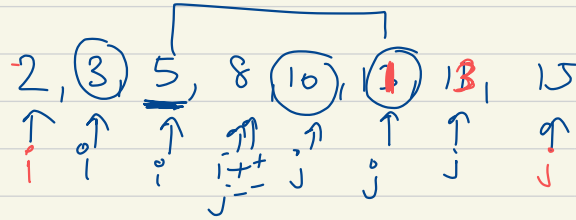
else  
     $i++$ ;  
}

[15] Redup

(17) (?:)

2 + 13 = 15 (?:)

2 + 13 = 15 (?:)



while ( $i < j$ ) {

}

$$2 + 5 = 17 \uparrow$$

$j--$

$$2 + 13 = 15$$

$$3 + 11 = 14 \downarrow$$

$$5 + 11 = 16 \uparrow$$

$$5 + 10 = 15 \begin{matrix} i++ \\ j-- \end{matrix}$$

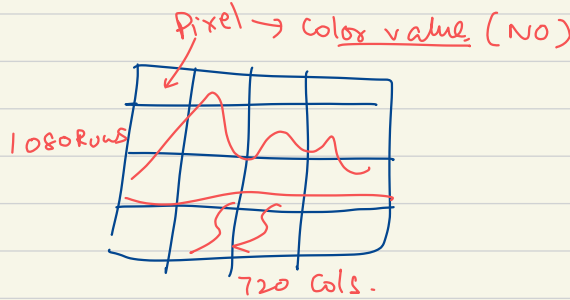
# " 2D ARRAYS "

1080 x 720

1920 x 1080 → Full HD

Data in  
grid

→ Rows &  
Columns



Image

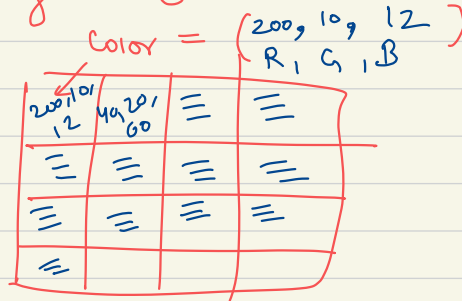
More info

↳ More Quality

↳ More memory

Data

Each image is just a matrix (2D Array) of numbers.



[RGB]

↑ ↑ ↑

0-255 0-255 0-255



0 ↓  
255 Full

- Images
- googlesheets
- matrices
- Chess, tic-tac-toe, cal



Byte  
 0, 0, 0 — Black  
 100-100-100 — Dark grey  
 200-200-200 — light grey  
 255, 255, 255 — white

## 2D Arrays in Java

int →   
 int[] → 

int[]

obj reference  
(stack)

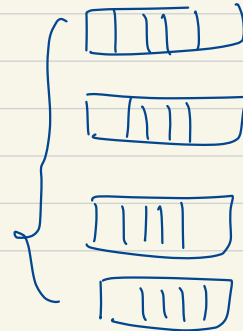
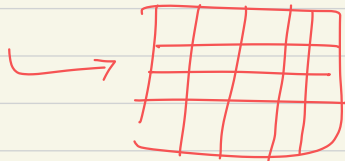
arr = new int[5];

actual array (heap)



int[][] arr = new int[4][5];

visualise



(optional)

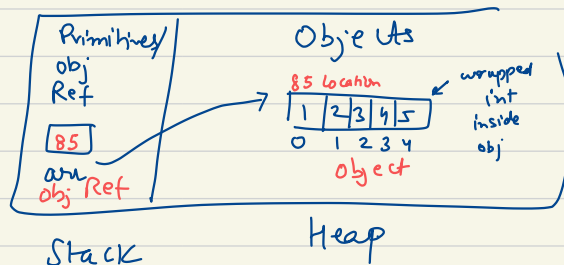
memory →  
↓

we can create only 1D Arrays

Object

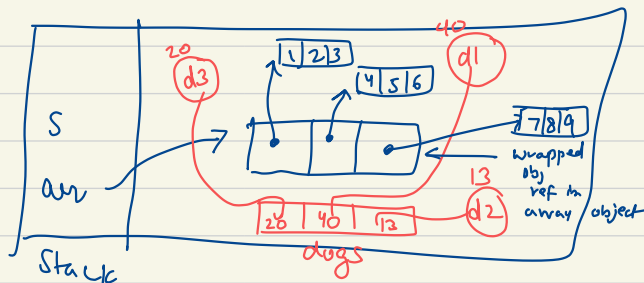
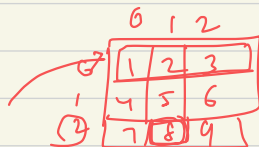
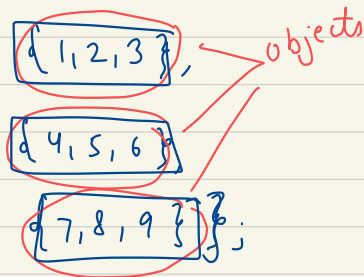
arr[] = { 1, 2, 3, 4, 5 }  
1D Array of ints

RAM (virtual) by JVM



int[][]  
arr = {

1D Array  
of  
Object (ARRAY)



arr[0][0] → 1  
arr[2][1] → 8

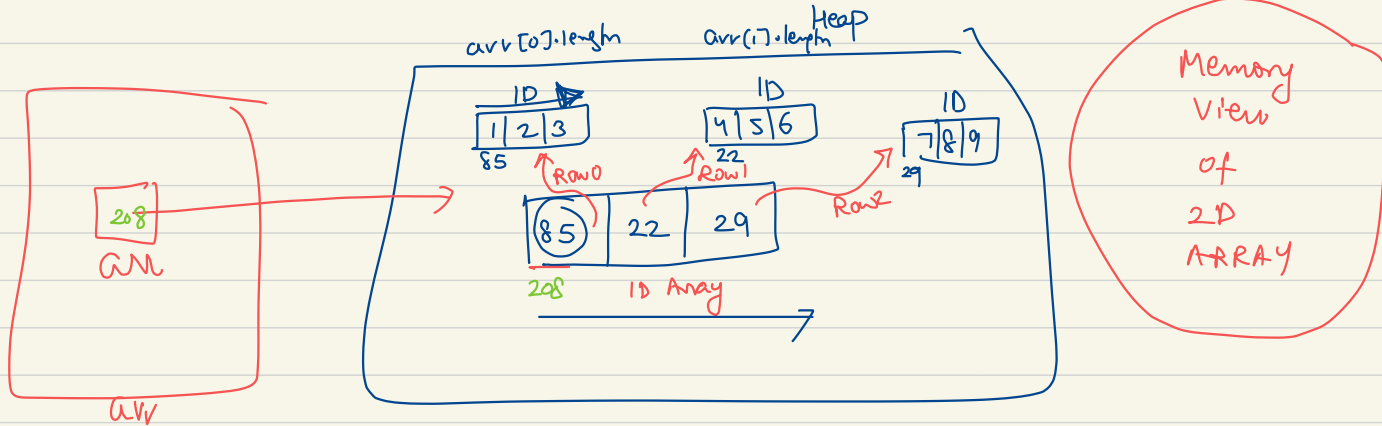
dogs[] = { d1, d2, d3 }

obj  
`int[][] arr =`  
 Stack

Each obj can be different  
 ↓  
 array → its own length

{ obj1, obj2, obj3 }

Row0: 1, 2, 3, 4, 5  
 Row1: 4, 5, 6  
 Row2: 7, 8, 9, 10, 11, 12

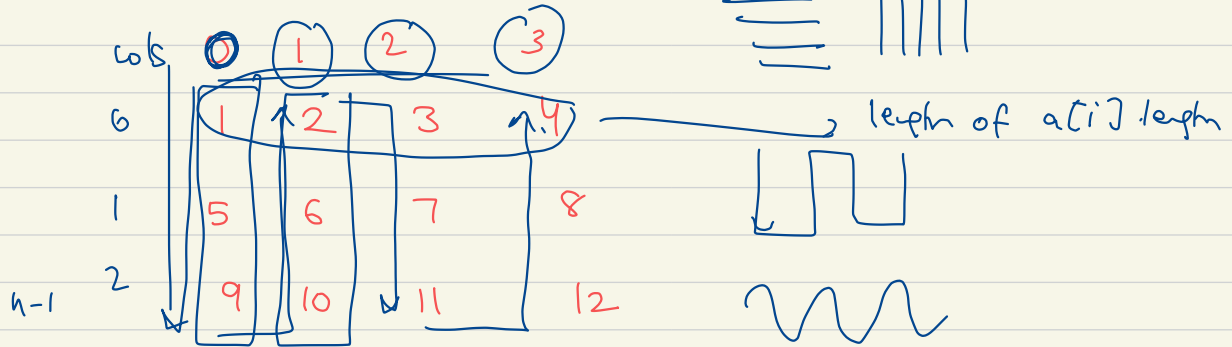


arr - 208  
 arr[0] - 85  
 arr[1] - 22 arr[2] - 29

Think ...

$N \times M$   
↑  
fixed cols

$N=3$   
 $M=4$



Print  
wave  
order

$\Rightarrow 1, 5, 9, 10, 6, 2, 3, 7, 11, 12, 8, 4$

(1) Col by Col.

for(

)

(2)

even col

odd col

Row  $\rightarrow$

Row  $n-1$

Row  $n-1$   $\rightarrow$  Row 0