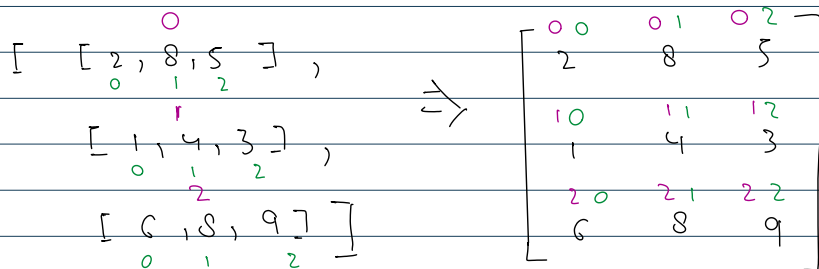
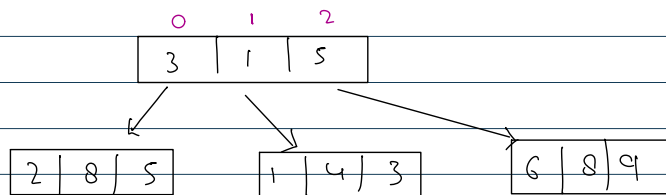


Array

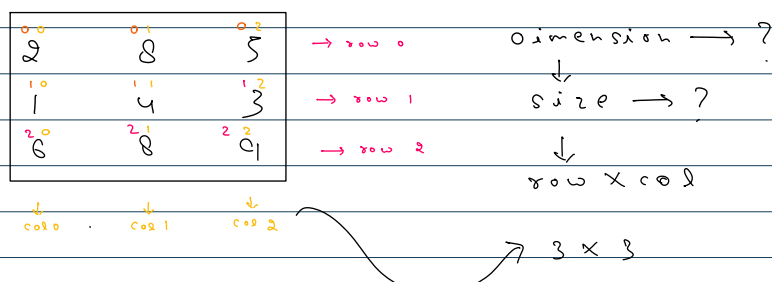
An array is a collection of elements of the same type.

arr \rightarrow { 3, 1, 5 } \rightarrow 1D



arr[2] \Rightarrow [6, 8, 9]

arr[2][0] \Rightarrow 6



$\left\{ \begin{array}{l} 1D \rightarrow \text{int[]} \text{ arr} = \text{new int}[5]; \\ 2D \rightarrow \text{int}[][] \text{ arr} = \text{new int}[3][3]; \end{array} \right.$

size \rightarrow (points to 5 in the first line)
 name of array \rightarrow (points to arr in the second line)
 row \rightarrow (points to 3 in the second line)
 col \rightarrow (points to 3 in the second line)

1D \rightarrow int[] arr = { 3, 1, 4 };

2D \rightarrow int[][] arr = {
 { 1, 2, 3 },
 { 4, 5, 6 },
 { 7, 8, 9 }
 };

2D array contiguous memory allocation ?

```
public class Main {
    public static void main(String[] args) {
```

```
int[][] arr = new int[3][4];
```

```
System.out.println(arr);
```

```
System.out.println(arr[0]);
```

```
System.out.println(arr[0][1]);
```

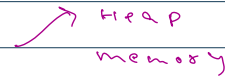
```
int[][] other = arr;
```

```
System.out.println(other);
```

```
int row = arr.length;
```

```
System.out.println(row);
```

```
int col = arr[0].length;  
System.out.println(col);
```



$n+1$ $\frac{1}{2}$, Total no. of ID array.

$n : 2$
$m : 3$

0 0	0 1	0 2
1 0	1 1	1 2

$00 \quad 01 \quad 02 \Rightarrow i = 0$ \rightarrow $0, 1, 2$ $\hookrightarrow \text{loop}(0 \text{ to } n-1)$
 $10 \quad 11 \quad 12 \Rightarrow i = 1$ \rightarrow $0, 1, 2$ $\hookrightarrow \text{loop}(0 \text{ to } n-1)$

$$\begin{array}{l} n : 2 \\ m : 3 \end{array}$$

0 0	0 1	0 2
1 0	1 1	1 2

Handwritten diagram illustrating the iterative steps of a loop from $i=0$ to $m-1$. The diagram shows the values of i and m at each step, with arrows indicating the progression of the loop.

Step	i	m
0	0	0
1	1	1
2	2	2

Arrows indicate the progression of the loop, with a final arrow pointing to $\text{loop}(0 \text{ to } m-1)$.

Day 10 Page 2

~~~~~

|   |   |   |
|---|---|---|
| 5 | 6 | 8 |
| 2 | 3 | 4 |
| 7 | 8 | 9 |

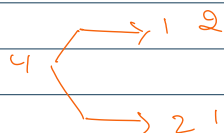
|   |   |   |
|---|---|---|
| 5 | 2 | 7 |
| 6 | 3 | 8 |
| 8 | 4 | 9 |

→ transpose of A

matrix(A)

|     |     |     |
|-----|-----|-----|
| 0 0 | 0 1 | 0 2 |
| 1 0 | 1 1 | 1 2 |
| 2 0 | 2 1 | 2 2 |

|     |     |     |
|-----|-----|-----|
| 0 0 | 1 0 | 2 0 |
| 0 1 | 1 1 | 2 1 |
| 0 2 | 1 2 | 2 2 |



$$arr[i][j] \rightarrow (i,j) \Rightarrow (j,i)$$

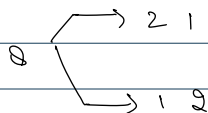
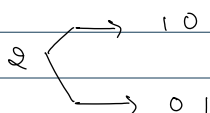
Transpose the matrix in-place

|   |   |   |
|---|---|---|
| 5 | 6 | 8 |
| 2 | 3 | 4 |
| 7 | 8 | 9 |

|   |   |   |
|---|---|---|
| 5 | 2 | 7 |
| 6 | 3 | 8 |
| 8 | 4 | 9 |

→ transpose of A

matrix(A)



|   |   |   |
|---|---|---|
| 5 | 6 | 8 |
| 2 | 3 | 4 |
| 7 | 8 | 9 |

|   |   |   |
|---|---|---|
| 5 | 2 | 7 |
| 6 | 3 | 8 |
| 8 | 4 | 9 |

lower triangular matrix → 10, 20, 21

↪ i > j

```
public class Main {
    public static void main(String[] args) {
```

```
int[][] arr = {  
    {5, 8, 9},  
    {0, 7, 3},  
    {1, 7, 9}  
};
```

```
int rows = arr.length;  
int cols = arr[0].length;
```

```
for(int i=0; i<rows; i++){  
    for(int j=0; j<cols; j++){  
        System.out.print(i+" "+j+" ");  
    }  
    System.out.println();  
}
```

```
System.out.println("*****");
```

```
for(int i=0; i<rows; i++){  
    for(int j=0; j<cols; j++){  
        System.out.print(arr[i][j]+" ");  
    }  
    System.out.println();  
}
```

```
System.out.println("*****");
```

```
for(int i=0; i<cols; i++){  
    for(int j=0; j<rows; j++){  
        System.out.print(j+" "+i+" ");  
    }  
    System.out.println();  
}
```

```
System.out.println("*****");
```

```
for(int i=0; i<cols; i++){  
    for(int j=0; j<rows; j++){  
        System.out.print(arr[j][i]+" ");  
    }  
    System.out.println();  
}
```

```
System.out.println("*****");
```

```
for(int i=0; i<rows; i++){  
    for(int j=0; j<cols; j++){  
        System.out.print(arr[j][i]+" ");  
    }  
    System.out.println();  
}
```

```
}
```

```
for(int i=0; i<rows; i++){  
    for(int j=0; j<cols; j++){  
        if(i>j)  
            swap(arr, i, j);  
    }  
}
```

```
System.out.println("*****");
```

```
for(int i=0; i<rows; i++){  
    for(int j=0; j<cols; j++){  
        System.out.print(arr[i][j]+" ");  
    }  
    System.out.println();  
}
```

```
}
```

```
public static void swap(int[][] arr, int i, int j){  
    int temp = arr[i][j];  
    arr[i][j] = arr[j][i];  
    arr[j][i] = temp;  
}
```

```
}
```

Search in a 2D Matrix II

$n \rightarrow \text{row}$   
 $m \rightarrow \text{col}$

Linear Search  $\Rightarrow O(n * m)$

Binary Search on  $\Rightarrow O(n \log m)$

every ID array

|    |    |    |    |    |
|----|----|----|----|----|
| 1  | 4  | 7  | 11 | 15 |
| 2  | 5  | 8  | 12 | 19 |
| 3  | 6  | 9  | 16 | 22 |
| 10 | 13 | 14 | 17 | 24 |
| 18 | 21 | 23 | 26 | 30 |

target  $\rightarrow 14$

int row = 0

int col = arr[0].length - 1;

while( \_ ) {

if (arr[row][col] == target)

return true;

else if (arr[row][col] > target)

col--;

else if (arr[row][col] < target)

row++;

}

row++;

}

|    |    |    |    |    |
|----|----|----|----|----|
| 1  | 4  | 7  | 11 | 15 |
| 2  | 5  | 8  | 12 | 19 |
| 3  | 6  | 9  | 16 | 22 |
| 10 | 13 | 14 | 17 | 24 |
| 18 | 21 | 23 | 26 | 30 |

target = 14

int row = 0

int col = arr[0].length - 1;

while ( ) {

if (arr[row][col] == t)

return true;

else if (arr[row][col] > t)

col--;

else if (arr[row][col] < t)

row++;

}

|    |    |    |    |    |
|----|----|----|----|----|
| 1  | 4  | 7  | 11 | 15 |
| 2  | 5  | 8  | 12 | 19 |
| 3  | 6  | 9  | 16 | 22 |
| 10 | 13 | 14 | 17 | 24 |
| 18 | 21 | 23 | 26 | 30 |

target = 25

row = matrix.length

|    |    |    |    |    |
|----|----|----|----|----|
| 1  | 4  | 7  | 11 | 15 |
| 2  | 5  | 8  | 12 | 19 |
| 3  | 6  | 9  | 16 | 22 |
| 10 | 13 | 14 | 17 | 24 |
| 18 | 21 | 23 | 26 | 30 |

target = 0

col = 0

|    |    |    |    |    |
|----|----|----|----|----|
| 1  | 4  | 7  | 11 | 15 |
| 2  | 5  | 8  | 12 | 19 |
| 3  | 6  | 9  | 16 | 22 |
| 10 | 13 | 14 | 17 | 24 |
| 18 | 21 | 23 | 26 | 30 |

⇒ очрм)

