23

Introduction To 2D Arrays

* Multidimensional Array

1D Array of int → Array of Integers

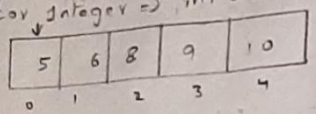
Multidimensional Array → Array of Arrays

```
datatype [1st dimension][2nd dimension][]..[Nth dimension]
arrayName = new datatype[size1][size2]...[sizeN]
```

1D Array

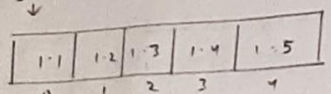
for Integer => int[] a

eg - a



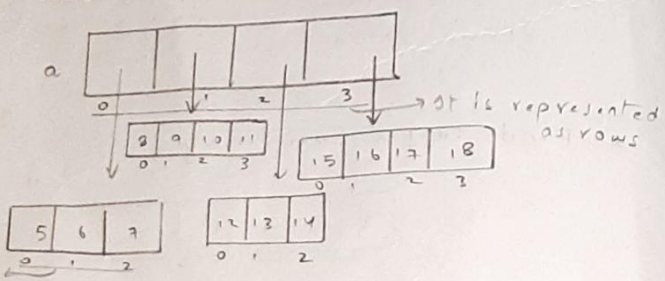
for Float => float[] a

a



2D Array

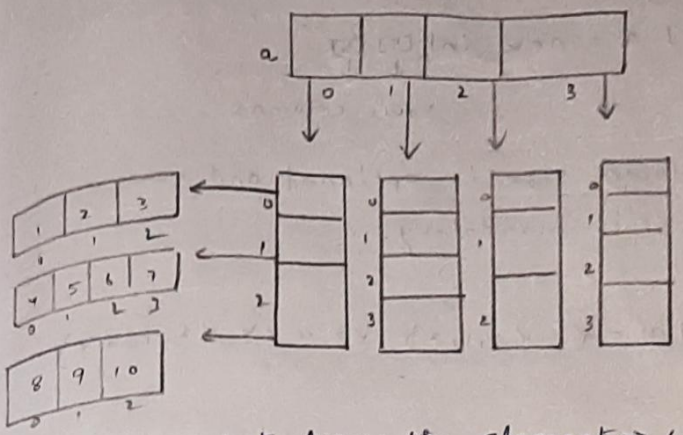
int[][] a



* We want to Access eg. value = 13 then

```
Sout(a[2][1]);
o/p = 13
```

* 3D Arrays

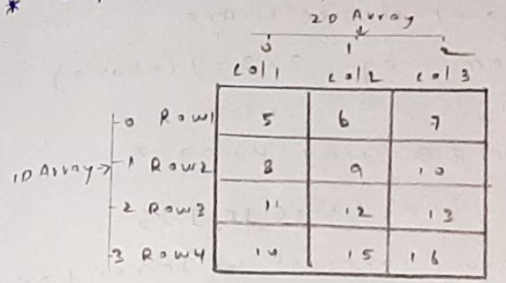


* We want to Access the element → 6 then

Sout(a[0][1][2]) => o/p → 6 //

* representation of 2D Array.

(Grid/Matrix/Table)



Jagged Arrays

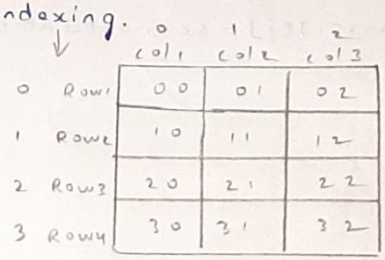
Different no of

* The Arrays in which we have ~~same~~ no of colomns known as Jagged Arrays

* The above representation is (4 x 3) dimension 2D Array

(4 → rows, 3 → colomns)

* Indexing.



row no, column no

00 01 ...

* Methods to Initialize 2D-Arrays

1.

int[][] a = new int[3] [6]

(size size) ↓ ↓ rows colomns

* colomn size is optional and row size is mandatory.

2. int[][] a = { {0,1,2}, {3,4,5}, {6,7,8} }

(row Indexing: 0, 1, 2; column Indexing: 0 1 2, 3 4 5, 6 7 8)

above is - 3×3 2D Array

* The total No of elements are rows × coloms eg- 3×3 = 9 (above)

* we can print an 2D Array using 2 for loops i.e Sout (arr[i][j]);

* outer loop for row traversal & Inner for Coloumn traversal, we find the coloumn length using arr[i].length.

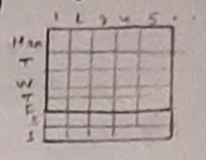
* we can take Input of 2D Array using 2 for loops.
i.e arr[i][j] = sc.nextInt();

→ why Multidimensional Arrays?

1. Graphs (we represent graph Data structure i/p in 2D Arrays)

2 Grids questions

3 storing Certain kind of Information like timetable, googlesheets we use 2D Array ↓



4 Fast & Easy Access

* Addition of 2 Matrices

$$\underset{A}{\begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}} + \underset{B}{\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}} = \underset{Sum}{\begin{bmatrix} 3 & 5 \\ 7 & 9 \end{bmatrix}}$$

2×2 (r_1 c_1) 2×2 (r_2 c_2) 2×2

• only Matrices with same dimensions are added (row & colomn should be equal)

Sum[i][j] = A[i][j] + B[i][j]

* Write a program to display Multiplication of two matrices entered by users.

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \end{bmatrix} * \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \\ b_{30} & b_{31} & b_{32} \end{bmatrix} = \begin{bmatrix} c_{00} & c_{01} & c_{02} \\ c_{10} & c_{11} & c_{12} \end{bmatrix}$$

$r_1 \times c_1$ (2×4) $r_2 \times c_2$ (4×3) $r_1 \times c_2$ (2×3)

• The column of first Matrix must be equal to row of Second matrix to satisfy Multiplication of 2 Matrices

• The resultant Matrix is row of first Matrix * column of Second Matrix.

$C_{00} = a_{00} \cdot b_{00} + a_{01} \cdot b_{10} + a_{02} \cdot b_{20} + a_{03} \cdot b_{30}$

$C_{01} = a_{00} \cdot b_{01} + a_{01} \cdot b_{11} + a_{02} \cdot b_{21} + a_{03} \cdot b_{31}$

$C_{02} = a_{00} \cdot b_{02} + a_{01} \cdot b_{12} + a_{02} \cdot b_{22} + a_{03} \cdot b_{32}$

Similarly C_{010}, C_{011}, C_{12}

multiply[i][j] = multiply[i][j] + a[i][k] * b[k][j]

eg-

1 (00)	2 (01)	1 (02)
3 (10)	1 (11)	2 (12)

2×3, r1 c1 — i is constant

×

2 (00)	1 (01)
1 (10)	3 (11)
1 (20)	1 (21)

3×2, r2 c2 — j is constant

=

2+2+1 = 5 (00)	1+6+1 = 8 (01)
6+1+2 = 9 (10)	3+3+2 = 8 (11)

2×2, r1 c2

condition → c1 == r2

code-

```
for(int i=0; i<r1; i++){
    for(int j=0; j<c2; j++){
        for(int k=0; k<c1/r2; k++){   // how many times we traverse multiply the rows, we dont no so we are taking this loop i.e k.
            mul[i][j] = mul[i][j] + a[i][k] * b[k][j]
        }
    }
}
```

* i & j are constant so we are taking the value k in logic to traverse the Multiplication.

HW
* Reverse each row

1	2	3	10
4	5	6	11
7	8	9	12

⟹

10	3	2	1
11	6	5	4
12	9	8	7

Basic Matrix Problems

* Write a program to display transpose of Matrix entered by the user

A (Input)

1 (00)	2 (01)	3 (02)
4 (10)	5 (11)	6 (12)
7 (20)	8 (21)	9 (22)

A^T (output)

1 (00)	4 (01)	7 (02)
2 (10)	5 (11)	8 (12)
3 (20)	6 (21)	9 (22)

The row is converted into column

original	Transpose
(0,0)	(0,0)
(0,1)	(1,0)
(0,2)	(2,0)
(1,0)	(0,1)
(1,1)	(1,1)
(1,2)	(2,1)
(2,0)	(0,2)
(2,1)	(1,2)
(2,2)	(2,2)

i,j → j,i

- row ↔ column (swap)
- Original elements are same because row, column values are same

method 1-

- **ans[i][j] = A[j][i]**
- eg- ans[0][1] 0 is i, 1 is j so in ans variable store a[j][i] i.e j=1 i=0 find (1,0) and store that value in ans variable

- A (2×3):

1	2	3
4	5	6

A^T (3×2):

1	4
2	5
3	6

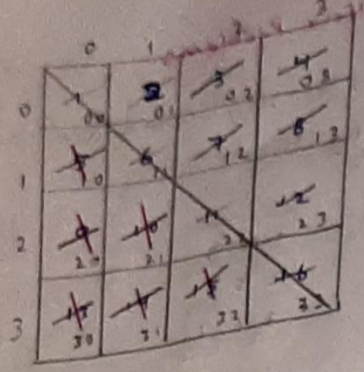
r×c → c×R so

int[][] Ans = new int [c][R];

method 2 - only for equal dimensions

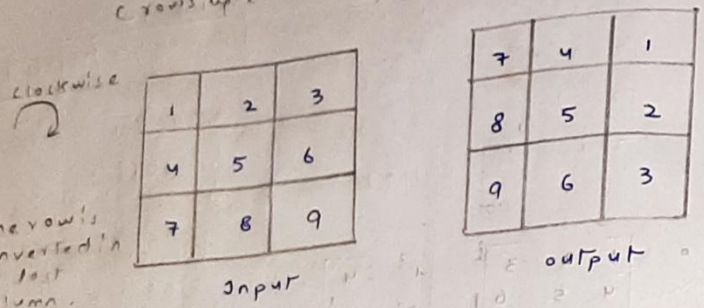
Swap (A[i][j], A[j][i]) → for In place,

means In same Array Transpose the Matrix.

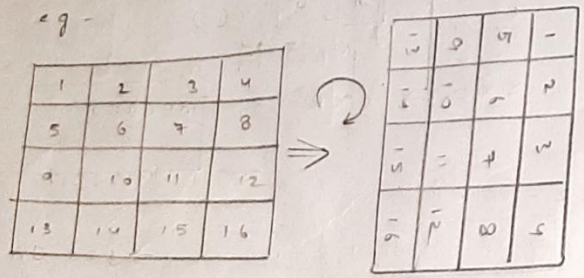


- So, for column variable start the iteration from i because if you start with 0 it again swap the values.

* Given a square matrix, turn it by 90 degrees in a clockwise direction without using any extra space (rows, & columns are equal)



eg -



eg -

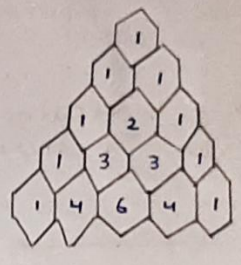
1 2 ⇒ 1 3 ⇒ 3 1
3 4 ⇒ 2 4 ⇒ 4 2

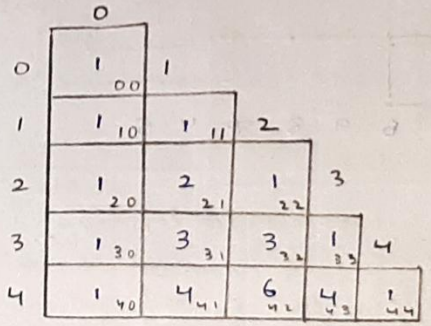
Matrix → transpose → reverse each row
(rows converted into columns)

* Given an Integer n, return the first n rows of pascals triangle.

In pascals triangle, each number is the sum of the two numbers directly above it as shown.

For n = 5





eg - P[4][2] = P[3][2] + P[3][1]

6 = 3 + 3

ith(row) = (ith row - 1) above + previous sums

Properties -

③ 1. p[i][j] = p[i-1][j] + p[i-1][j-1]

② 2 In every row first & last element is 1

① 3 Jagged Arrays → ith row has i+1 columns

Implement order for code

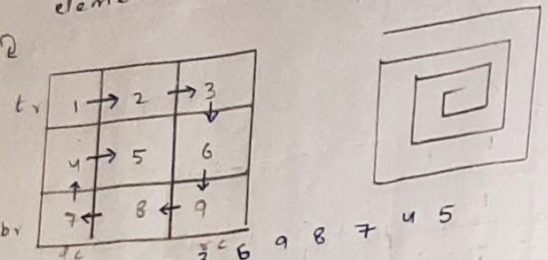
25

Spiral Matrix Traversal & Generation

Starting 2 min → Motivation (for how to solve problems)

Pattern - Spiral Matrix

* Given an n x m matrix 'a', return all elements of the matrix in spiral order



o/p - 1 2 3 6 9 8 7 4 5

- print
  1. top row - traverse from left column to right column
  2. right column - traverse from top row to bottom row
  3. bottom row - traverse from right col to left col
  4. left column - traverse from bottom row to top row

Algorithm.

1. In while loop traverse if total elements < row * column value, initialized first total elements = 0.
2. traverse the top row from left column to right column & increment the top row.
3. traverse the right column from top row to bottom row and decrement the right column.
4. traverse the bottom row from right column to left column and decrement the bottom row.
5. traverse the left column from bottom row to top row and increment the left column.
6. there is a chance that the total elements will print extra because they are 4 for loops inside it & for which loop after which element the extra element will print we can't say so give condition in inside loops also that total elements < r * c.

* Given a positive integer n, generate an n x n matrix filled with elements from 1 to n^2 in spiral order.

1. n = 3 n x n = 3 x 3 = 9 elements.

	0	1	2
0	1 →	2 →	3 ↓
1	8 →	9	4 ↓
2	7 ←	6 ←	5

2. n = 5 5 x 5 = 25 elements

	0	1	2	3	4
0	1	2	3	4	5
1	16	17	18	19	6
2	15	24	25	20	7
3	14	23	22	21	8
4	13	12	11	10	9

HW * print & Access the elements in spiral order (Anticlockwise direction). G

Algorithm - From above problem instead of printing/traversing assign the values. i.e from current = 1 to current < = n x n & inc current

26

Prefix Sum in a Matrix

Pattern: prefix sums in 2D Arrays

* Given a matrix 'a' of dimension $n \times m$ (r × c) and 2 coordinates (l_1, r_1) & (l_2, r_2).

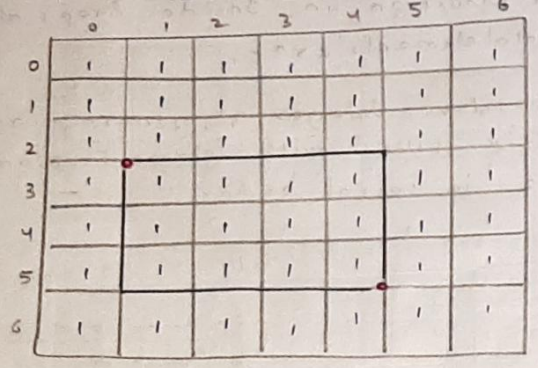
return the sum of the rectangle from (l_1, r_1) to (l_2, r_2)

- $l_2 \geq l_1$, $r_2 \geq r_1$
- $0 \leq l_1, l_2 < n$
- $0 \leq r_1, r_2 < m$

method 1: Brute force (we are using 2 nested loops)

not good Approach

eg - $l_1 = 3$ $r_1 = 1$ (means 3rd row 1st column to upto)
$l_2 = 5$ $r_2 = 4$ (5th row 4th column)



we have to sum the inside elements
o/p - ans = 12

- rows & columns need to be of same dimensions. & the elements also need not to be the same they can be any value 1, 2, 3, ...

Algorithm -

* traverse outer loop for row from l_1 to l_2
* traverse inner loop for column from r_1 to r_2
- add & store the total elements enclosed in that rectangle's boundaries. i.e Sum = Sum + a[i][j]

(By using method 2 we are saving the nested loop)

method 2 - pre calculating the horizontal sum for each row in the matrix

Better Approach

- if we are changing in place (in same Array) it doesn't return anything (3-methods)

Algorithm - matrix → prefix sum row wise → calculate.

- firstly calculate the prefix sum of entire Array row wise (traverse horizontally to calculate row wise sum.

```
i.e  for(int i=0; . . .)
       for(int j=1; . . .)
         arr[i][j] = arr[i][j] + a[i][j-1]
       }
     }
```

- traverse outer loop (of rectangle Sum) for row from l_1 to l_2.
- we don't need to traverse the inner loop for column traverse because we have the prefix sum.
- for certain range i.e from l_1 to l_2 calculate sum by

(for i=0, i=1, ... until l_2) | Sum = Sum + (a[i][r_2] - a[i][r_1 - 1])

(r_1: first column for which we want the sum. r_2: last column for which we want the sum)

if r_1 is greater than =1

else

Sum = Sum + (a[i][r_2])

if r_1 is == 0 we don't need anything to subtract so we are using else block

method 3 - by using method 3 we are saving the 2 for loops

optimized Approach

(prefix Sum over columns and Rows Both)

Matrix a

	0	1	2	3	4	5	6
0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1

Sum of rectangle is 18

Prefix sum of a for rows

	0	1	2	3	4	5	6
0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
2	1	2	3	4	5	6	7
3	1	2	3	4	5	6	7
4	1	2	3	4	5	6	7
5	1	2	3	4	5	6	7
6	1	2	3	4	5	6	7

6 representing the sum of the row, so if we add 6+6+6 it will give the sum of rows from $l_1=0, r_1=0$ to $l_2=2$ $r_2=5$ i.e 18

Prefix sum of a for columns

	0	1	2	3	4	5	6
0	1	2	3	4	5	6	7
1	2	4	6	8	10	12	14
2	3	6	9	12	15	18	21
3	4	8	12	16	20	24	28
4	5	10	15	20	25	30	35
5	6	12	18	24	30	36	42
6	7	14	21	32	35	42	49

By using prefix sum column wise we get 18 which is the sum of the elements from $l_1=0$ $r_1=0$ to $l_2=2$ $r_2=5$

a[i][j] = SumofRectangle((0,0) (i,j))

eg-

blue up black = leftup □6

blue = up □15

blue = up
black = left
blue up black = leftup

black = left □12

	0	1	2	3	4	5	6
0	1	2	3	4	5	6	7
1	2	4	6	8	10	12	14
2	3	6	9	12	15	18	21
3	4	8	12	16	20	24	28
4	5	10	15	20	25	30	35
5	6	12	18	24	30	36	42
6	7	14	21	32	35	42	49

l_1,r_1 — l_1,r_2 — l_2,r_1 — l_2,r_2

* we want the sum of the elements inside red boundary.
* 30 is representing the sum of the elements from $l_1=0$ $r_1=0$ to $l_2=5$ $r_2=4$ i.e elements inside colored boundaries □30
* If we subtract up, left & add leftup to sum we get the Answer.

Ans = Sum - up - left + leftup
(blue) (black) (blue up black)

from above,

Sum = a[l2][r2] i.e 30

up = a[l1-1][r2] i.e 15

left = a[l2][r1-1] i.e 12

leftup = a[l1-1][r1-1] i.e 6

Ans = 30 - 15 - 12 + 6

Ans = 9

Algorithm

* convert the Matrix in to prefix sum of in row wise.
* convert that prefix sum of row wise in to prefix sum of column wise
* Implement the logic i.e

  Ans = Total sum - up - left + leftup.

* If l1, r1, l2, r2 is less than Index 1 It will give negative Index value so check if coordinates are greater than or equal to 1

28

Introduction to ArrayLists

* Wrapper classes
  → a class whose object contains or wraps primitive datatype (int, float ...)
  → object of a wrapper class contains a field which stores PDT

PDT	Wrapper class
int	Integer
float	Float
char	Character
boolean	Boolean
long	Long.

* eg - Integer i = Integer.valueOf (4);
  o/p - 4 Sout (i);

* Refer vscode for Wrapper class theory.

* ArrayList class

→ It is a class which is present in java.util package how can eg. we want to Import this

import java.util.ArrayList;

→ ArrayList overcome the limitation of Arrays with extra functionalities

limitations:

1. Syntax of declaring Array is

   int[] arr = new int[5];

   while making this Array arr we have declared that the size is 5 & we cannot change the size, during whole program the size is 5 which is not convinient we may get some situations while building real life programs like we want to add new element or delete the element, so our option is make another Array of size 6 and from first Array copy the 5 elements and add 6th element which is very tedious task to overcome this limitation Arraylist is used. Arraylist are mostly like arrays only but they have variable size means when we declare an array it is not mandatory that during declaration only we want to declare the size. and during program we can change, add or decrease size. (they donot have fixed size) & they have some methods/utilities Inbuilt which makes easy to code

* Syntax to create ArrayList

List<Anyclass> list = new ArrayList<Anyclass>();

eg -

ArrayList<Integer> l1 = new ArrayList<Integer>();

- ArrayList<Integer> → wrapper class
- l1 → list Name
- <Integer> → optional to write this class
- () → size also optional

* common methods which are Available In ArrayList

→ to add a new element (at end)

```
listName.add(5);
// l1.add(5);
```

→ get an element at index ; (Access)

```
Sout(l1.get(0));
```

- l1 → list Name
- 0 → index Value

→ print with for loop

```
for(int i=0; i<l1.size(); i++){
    Sout(l1.get(i));
}
```

→ printing the ArrayList Directly

```
Sout(l1); // [5, 6, 7, 8]
```

→ Adding element at index ;

```
l1.add(index: 1, element: 100)
```

→ modifying element at index ;

```
l1.set(index: 1, element: 10)
```

→ removing an element at index ;

```
l1.remove(index: 1)
```

→ removing an element e (we dont no the index)

~~element:~~

```
~~l1.remove(10)~~
```

Integer

```
l1.remove(valueOf(8));
```

- It returns boolean value True / False

→ checking if an element exists

It also returns boolean

```
l1.contains(Integer.valueOf(6))
```

→ If you dont specify class, you can put anything inside

eg:

```
ArrayList l = new ArrayList();
l.add("suraj")
l.add(9);
l.add(true);
Sout(l); // [suraj, 9, true]
```

HW

Q. int indexOf(object o)
int lastIndexOf(object o)
boolean isEmpty()

Q. Write a program to Reverse the given ArrayList

Input - [0, 10, 3, 5, 22, 10]

output - [10, 22, 5, 3, 10, 0]

method 1 - using 2 pointers swap method

```
i=0  j= list.size()-1
while (i<j) {
Integer temp = Integer.valueOf(list.get(i));
list.set(i, list.get(j));
list.set(j, temp);
i++;
j--;
}
```

method 2 - Inbuilt method

In java.util package we have collections class we can Access using the method.

```
import java.util.Collections;
Collections.reverse(list);
```

Q Write a program to sort an ArrayList of strings in descending order

→ In collections class we have a method to sort i.e

```
collections.sort(list) //sorts in Ascending order

collections.sort(list, Collections.reverseOrder()); //sorts in Descending order.
```