5.(b) (2D Arrays)

→ A 2D Array is essentially an array of arrays. It stores elements in a table-like Structure with rows and Columns.

(1.) Syntax:

(i) Static Initalization :

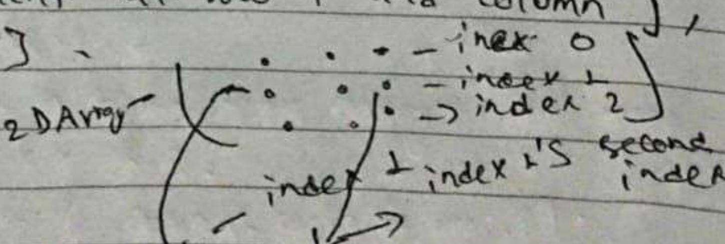
```
int [ ][2] matrix = { {1,2,3},   0th index
                      {4,5,6}    1st index
                    };              .
                                    .
                                  So on.
```

(ii) Dynamic Initalization :

```
                              3 rows
                                        4 columns
int[ ][ ] matrix = new [3] [4] ;
              First row   index of First row.
assign→   matrix [0] [1] = 99 ;
              --- So on.
```

(2) Accessing Elements

→ To access an element at row i and Column j, use matrix [i] [j] .



Syntax:

```
int val = matrix [1] [2];   //value in 2nd row, 
                              3rd Column.
```

(3) Input and Output of 2D Arrays

→ we use nested loops. Outer loop for rows, inner loop for columns.

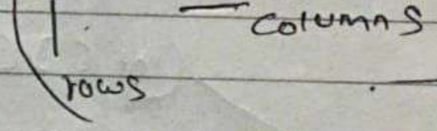
(i) Input Syntax (using Scanner):

(.length prints rows only

```
// Ask user input for rows & colums.
  int rows = --- ; int cols = ---;

// Declare and initilize the 2D array
int[][]matrix = new int [rows][cols];

  // Take input. for each element.
  for(int i=0; i<rows; i++){        → rows
   for(int j=0; j<cols; j++){       → columns
       matrix[i][j] = sc.nextInt();
     }
  }
```



(ii) Output Syntax

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

4. ~~Questions~~ Traversing 2D Arrays

(i) Row-wise Traversal (Nested Loops)

⇒ Row-wise traversal means you visit all elements in a row before moving to next row.

Example :-

```
public class Main {
    psvm (String[] args){
int [][] matrix = { {1,2,3},
                    {4,5,6},
                    {7,8,9} };
    System.out.println("Row-wise Traversal");
for(int i = 0; i<matrix.length; i++){ // outer loop: iterate over rows.
for(int j=0; j<matrix[i].length; j++){ // inner loop: iterate over Columns.
    System.out.print(matrix[i][j]);
        }
    System.out.println(); //move to next line each row
    }
  }
}
```

(ii) Column-wise Traversal {

⇒ visits all elements in a column before moving to the next column.

```
        // Same till this.
    System.out.println ("Column-wise Traversal");
for(int j=0; j<matrix[0].length; j++){ // outer loop -> colums.
for(int i=0; i<matrix.length; i++){ // inner loop -> rows
    // Same after this.
```