

# Intro to 2D Lists

Tic-Tac-Toe Board is organized as: Board[Row][Column]

	Column			
Row		0	1	2
	0	Board[0][0]	Board[0][1]	Board[0][2]
	1	Board[1][0]	Board[1][1]	Board[1][2]
	2	Board[2][0]	Board[2][1]	Board[2][2]

Board is set up like this:

```
Board = [ [ Board[0][0], Board[0][1], Board[0][2] ],  
          [ Board[1][0], Board[1][1], Board[1][2] ],  
          [ Board[2][0], Board[2][1], Board[2][2] ] ]
```

## SETUP

```
rows, columns = map(int, input().split())  
matrix = []
```

```
for row in range(rows):  
    line = list(map(int, input().split()))  
    matrix.append(line)
```

```
for row in range(rows):  
    for column in range(columns):  
        print(matrix[row][column], end = " ")  
    print()
```

## SCALE

```
rows, columns = map(int, input().split())  
matrix = []
```

```
for row in range(rows):  
    line = list(map(int, input().split()))  
    matrix.append(line)
```

```
factor = int(input())
```

```
for row in range(rows):  
    for column in range(columns):  
        print(matrix[row][column] * factor, end = " ")  
    print()
```

## MAXIMUM

```
rows, columns = map(int, input().split())  
matrix = []
```

```
maxVal, maxRow, MaxCol = 0, 0, 0
```

```
for row in range(rows):  
    line = list(map(int, input().split()))  
    matrix.append(line)
```

```
for row in range(rows):  
    for column in range(columns):  
        current = matrix[row][column]  
        if current > maxVal:  
            maxVal, maxRow, maxCol = current, row, column  
print(maxRow, maxCol, end="")
```

## DIAGONALS

```
rows = columns = int(input())  
matrix = []
```

```
for row in range(rows):  
    line = list(map(int, input().split()))  
    matrix.append(line)
```

```
for row in range(rows):  
    for column in range(columns):  
        print(_____, end = " ")  
    print()
```

### SNOWFLAKE

```
rows = columns = int(input())
middle = rows // 2
matrix = []
```

```
# Creating a 2D list with dynamic dimensions
matrix = [[0 for _ in range(columns)] for _ in range(rows)]
```

```
for row in range(rows):
    for column in range(columns):
        if row + col == rows - 1: # Right Diagonal
            print("*", end = " ")
        elif row == col : # Left Diagonal
            print("*", end = " ")
        elif col == middle:
            print("*", end = " ")
        else:
            print(".", end = " ")
    print()
```

5

	0	1	2	3	4
0	*	.	*	.	*
1	.	*	*	*	.
2	*	*	*	*	*
3	.	*	*	*	.
4	*	.	*	.	*

### SWAP COLUMNS

```
rows, columns = map(int, input().split())
matrix = []
```

```
for row in range(rows):
    line = list(map(int, input().split()))
    matrix.append(line)
```

```
col1, col2 = map(int, input().split())
```

```
for row in range(rows):
    for column in range(columns):
        if column == col1:
            print(matrix[row][col2], end = " ")
        elif column == col2:
            print(matrix[row][col1], end = " ")
        else:
            print(matrix[row][column], end = " ")
    print()
```

3 4

	0	1	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34

0 1

	1	0	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34

**BONUS: TIC TAC TOE**

```
def checkForWinner(Board):  
    # Check for horizontal wins  
    for row in range(3):  
        if Board[row][0] == Board[row][1] == Board[row][2]:  
            return Board[row][0]  
    # Check for vertical wins  
    for col in range(3):  
        if Board[0][col] == Board[1][col] == Board[2][col]:  
            return Board[0][col]  
    # Check for diagonal wins  
    if Board[_][_] == Board[_][_] == Board[_][_]:  
        return Board[0][0]  
    if Board[_][_] == Board[_][_] == Board[_][_]:  
        return Board[0][2]  
    # If no winner, return None  
    return None
```

```
Board = [ [ 7, 8, 9],  
          [ 4, 5, 6],  
          [ 1, 2, 3] ]  
turns = 9
```

```
for turn in range(turns):  
    if turn % 2 == 0:  
        Symbol = "X"  
    else:  
        Symbol = "O"  
    print("Player", Symbol, "it's your turn!")  
  
    choice = input()  
    Row = 2 - (choice // 3)  
    Column = choice % 3  
    Board[Row][Column] = Symbol  
  
    for row in range(3):  
        for column in range(3):  
            print(Board[row][column], end = " ")  
        print()  
    print()  
  
    winner = checkForWinner(Board)  
    if winner:  
        print("Player", winner, "wins!")  
        break
```