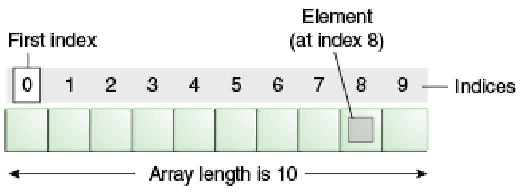
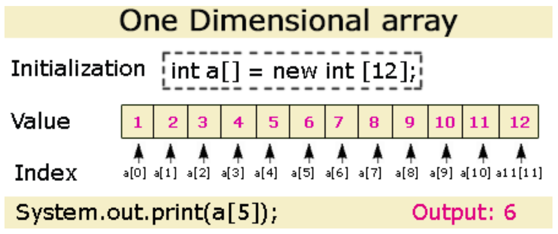
2-D Arrays

1-D Arrays





One Dimensional array

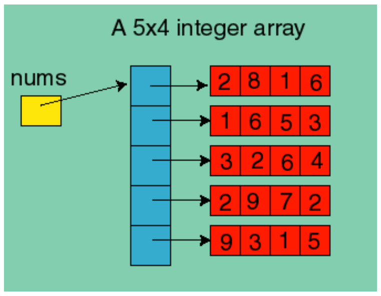
Initialization `int a[] = new int [12];`

Value	1	2	3	4	5	6	7	8	9	10	11	12
Index	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]	a[10]	a11[11]

`System.out.print(a[5]);` Output: 6

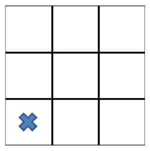
2-D Arrays

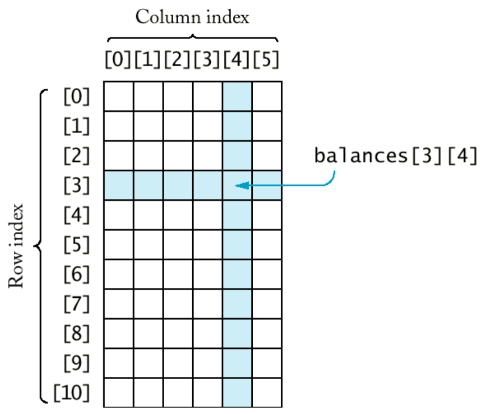
- `int[][] nums = new int[5][4];`



2-D Array as a table

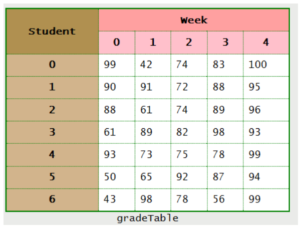
```
int numRows = 3;
int numCols = 3;
String[][] table =
          new String[numRows][numCols];
table[2][0] = “x”;
```





Question

- Determine the value stored in each of the following:
  - gradeTable[0][0]
  - gradeTable[1][1]
  - gradeTable[3][4]
  - gradeTable[5][2]



Student	Week				
	0	1	2	3	4
0	99	42	74	83	100
1	90	91	72	88	95
2	88	61	74	89	96
3	61	89	82	98	93
4	93	73	75	78	99
5	50	65	92	87	94
6	43	98	78	56	99

gradeTable

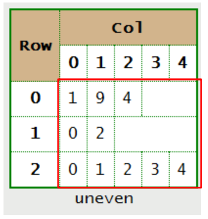
Different Numbers of Cells per Row

- Usually you think of a 2D array as a table
- Is this possible:

```
int[][] uneven =
                {{ 1, 9, 4 },
                 { 0, 2},
                 { 0, 1, 2, 3, 4 }};
```

Different Numbers of Cells per Row

- You have to think of "arrays of references to 1D arrays.“

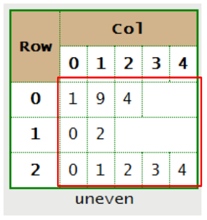


Row	Col				
	0	1	2	3	4
0	1	9	4		
1	0	2			
2	0	1	2	3	4

uneven

Different Numbers of Cells per Row

- Determine the value stored in each of the following:
  - uneven[0][0]
  - uneven[1][1]
  - uneven[0][4]
  - uneven[2][4]



Row	Col				
	0	1	2	3	4
0	1	9	4		
1	0	2			
2	0	1	2	3	4

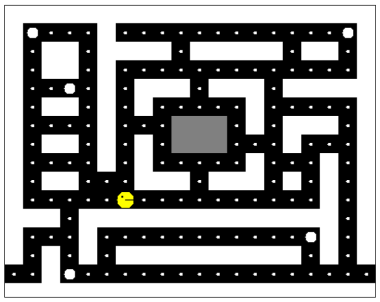
uneven

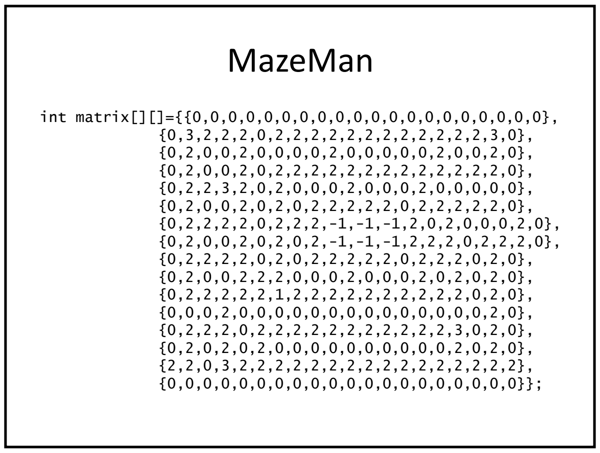
Printing a 2D Array

```
// declare and construct a 2D array
int[][] uneven = { { 1, 9, 4 },
                   { 0, 2},
                   { 0, 1, 2, 3, 4 } };
// print out the array
for(int row=0; row < uneven.length; row++)
{
  System.out.print("Row " + row + ": ");
  for(int col=0; col < uneven[row].length; col++)
    System.out.print(uneven[row][col] + " ");
  System.out.println();
}
```

MazeMan

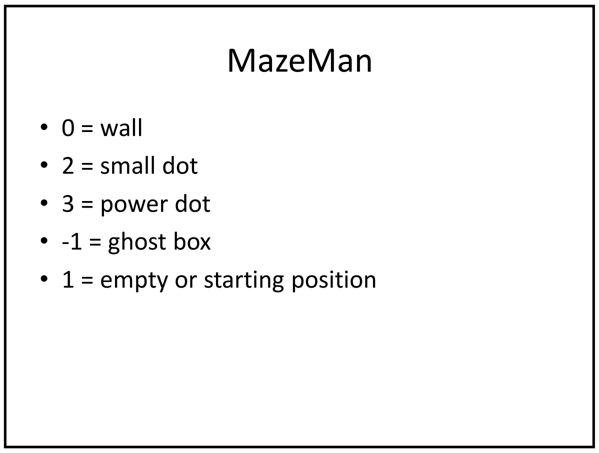
16x20





MazeMan

```
int matrix[][]={{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
                {0,3,2,2,2,0,2,2,2,2,2,2,2,2,2,2,2,2,3,0},
                {0,2,0,0,2,0,0,0,0,2,0,0,0,0,0,2,0,0,2,0},
                {0,2,0,0,2,0,2,2,2,2,2,2,2,2,2,2,2,2,2,0},
                {0,2,2,3,2,0,2,0,0,0,2,0,0,0,2,0,0,0,0,0},
                {0,2,0,0,2,0,2,0,2,2,2,2,2,0,2,2,2,2,2,0},
                {0,2,2,2,2,0,2,2,2,-1,-1,-1,2,0,2,0,0,0,2,0},
                {0,2,0,0,2,0,2,0,2,-1,-1,-1,2,2,2,0,2,2,2,0},
                {0,2,2,2,2,0,2,0,2,2,2,2,2,0,2,2,2,0,2,0},
                {0,2,0,0,2,2,2,0,0,0,2,0,0,0,2,0,2,0,2,0},
                {0,2,2,2,2,2,1,2,2,2,2,2,2,2,2,2,2,0,2,0},
                {0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0},
                {0,2,2,2,0,2,2,2,2,2,2,2,2,2,2,2,3,0,2,0},
                {0,2,0,2,0,2,0,0,0,0,0,0,0,0,0,0,2,0,2,0},
                {2,2,0,3,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2},
                {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}};
```



MazeMan

- 0 = wall
- 2 = small dot
- 3 = power dot
- -1 = ghost box
- 1 = empty or starting position