

Chapter 8 Multidimensional Arrays

CS1: Java Programming
Colorado State University

Original slides by Daniel Liang
Modified slides by Chris Wilcox



Announcements

- Lab Wednesday
- Review in Lecture Wednesday – Bring Questions
- Exam Friday
- Spring Break next week – get some rest!



Motivations

Thus far, you have used one-dimensional arrays to model linear collections of elements. You can use a two-dimensional array to represent a matrix or a table. For example, the following table that describes the distances between the cities can be represented using a two-dimensional array.

	Chicago	Boston	New York	Atlanta	Miami	Dallas	Houston
Chicago	0	983	787	714	1375	967	1087
Boston	983	0	214	1102	1763	1723	1842
New York	787	214	0	888	1549	1548	1627
Atlanta	714	1102	888	0	661	781	810
Miami	1375	1763	1549	661	0	1426	1187
Dallas	967	1723	1548	781	1426	0	239
Houston	1087	1842	1627	810	1187	239	0

Other Representations?

What are some other representations of multi-dimensional arrays?



Declare/Create Two-dimensional Arrays

```
// Declare array ref var  
dataType[][] refVar;
```

```
// Create array and assign its reference to  
variable  
refVar = new dataType[10][10];
```

```
// Combined  
dataType[][] refVar = new dataType[10][10];
```



Declaring Variables of Two-dimensional Arrays and Creating Two-dimensional Arrays

```
int[][] matrix = new int[10][10];
```

```
matrix[0][0] = 3;
```



Two-dimensional Array Illustration

	[0]	[1]	[2]	[3]	[4]
[0]	0	0	0	0	0
[1]	0	0	0	0	0
[2]	0	0	0	0	0
[3]	0	0	0	0	0
[4]	0	0	0	0	0

```
matrix = new int[5][5];
```

(a)

	[0]	[1]	[2]	[3]	[4]
[0]	0	0	0	0	0
[1]	0	0	0	0	0
[2]	0	7	0	0	0
[3]	0	0	0	0	0
[4]	0	0	0	0	0

```
matrix[2][1] = 7;
```

(b)

	[0]	[1]	[2]
[0]	1	2	3
[1]	4	5	6
[2]	7	8	9
[3]	10	11	12

```
int[][] array = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9},  
    {10, 11, 12}  
};
```

(c)

matrix.length? 5

matrix[0].length? 5

array.length? 4

array[0].length? 3

Declaring, Creating, and Initializing Using Shorthand Notations

You can also use an array initializer to declare, create and initialize a two-dimensional array. For example,

```
int[][] array = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9},  
    {10, 11, 12}  
};
```

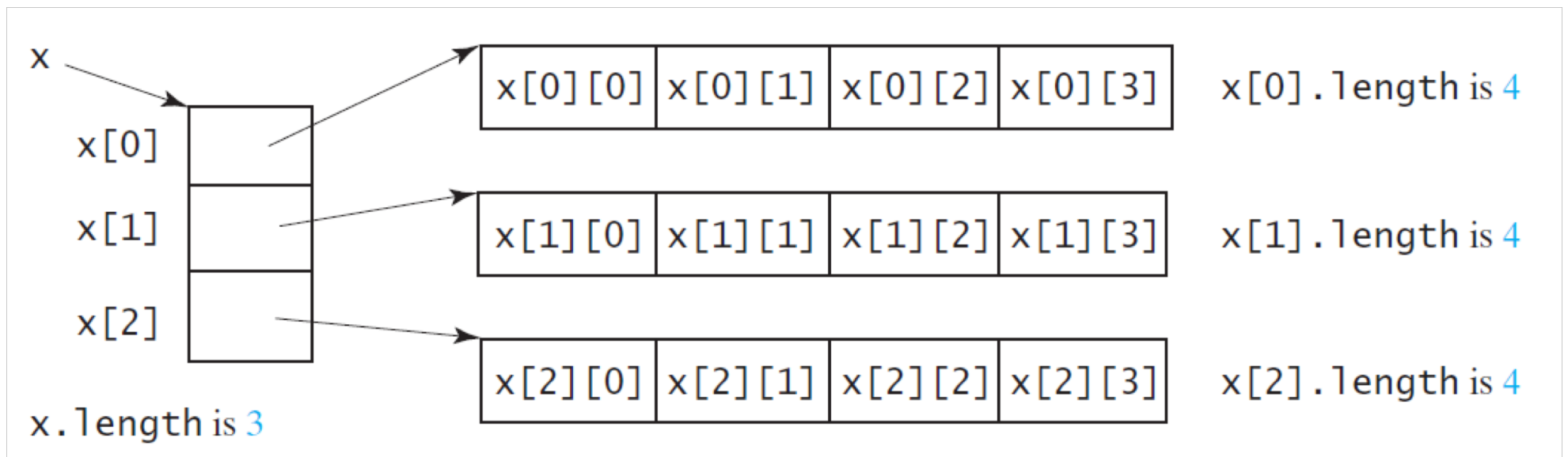
Same as

```
int[][] array = new int[4][3];  
array[0][0] = 1; array[0][1] = 2; array[0][2] = 3;  
array[1][0] = 4; array[1][1] = 5; array[1][2] = 6;  
array[2][0] = 7; array[2][1] = 8; array[2][2] = 9;  
array[3][0] = 10; array[3][1] = 11; array[3][2] = 12;
```



Lengths of Two-dimensional Arrays

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



Lengths of Two-dimensional Arrays, cont.

```
int[][] array = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9},  
    {10, 11, 12}  
};
```

array.length

array[0].length

array[1].length

array[2].length

array[3].length

array[4].length

ArrayIndexOutOfBoundsException



Ragged Arrays

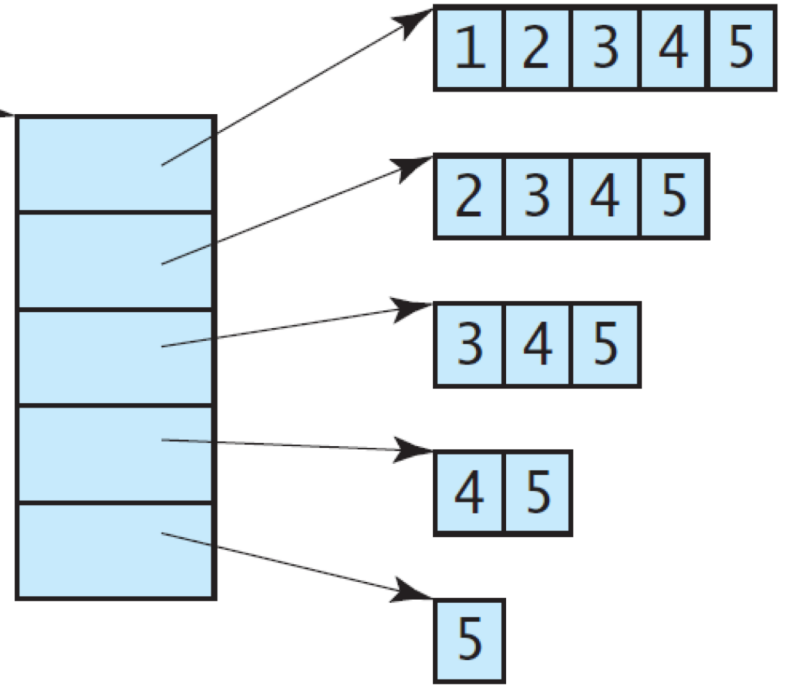
Each row in a two-dimensional array is itself an array. So, the rows can have different lengths. Such an array is known as *a ragged array*. For example,

```
int[][] matrix = {  
    {1, 2, 3, 4, 5},  
    {2, 3, 4, 5},  
    {3, 4, 5},  
    {4, 5},  
    {5}  
};
```

```
matrix.length is 5  
matrix[0].length is 5  
matrix[1].length is 4  
matrix[2].length is 3  
matrix[3].length is 2  
matrix[4].length is 1
```

Ragged Arrays, cont.

```
int[][] triangleArray = {  
    {1, 2, 3, 4, 5},  
    {2, 3, 4, 5},  
    {3, 4, 5},  
    {4, 5},  
    {5}  
};
```



Initializing arrays with random values

```
for (int row = 0; row < mat.length; row++)  
{  
    for (int col = 0; col < mat[row].length; col++)  
    {  
        mat[row][col] = (int) (Math.random() * 100);  
    }  
}
```



Printing arrays

```
for (int row = 0; row < mat.length; row++)
{
    for (int col = 0; col < mat[row].length; col++)
    {
        System.out.print(mat[row][col] + " ");
    }

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



Summing elements by column

```
for (int col = 0; col < mat[0].length; col++)  
{  
    int total = 0;  
    for (int row = 0; row < mat.length; row++)  
        total += matrix[row][column];  
  
    System.out.println(column + " is " + total);  
}
```

