

Multidimensional Arrays

CSE 114, Computer Science 1

Stony Brook University

<http://www.cs.stonybrook.edu/~cse114>

Multidimensional Arrays

- A two-dimensional array to represent a matrix or a table
 - Example: the following table that describes the distances between the cities can be represented using a two-dimensional array.

Distance Table (in miles)

	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

Declare/Create

Two-dimensional Arrays

```
// Declare array ref var
```

```
dataType[][] refVar;
```

```
// Create array and assign its reference to variable
```

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

```
// Combine declaration and creation in one statement
```

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

```
// Alternative syntax - Not preferred!
```

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

Default Values

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

- Default values:
 - numeric types get 0
 - boolean get false
 - char gets '\u0000'
 - reference types (e.g., String): null

```
System.out.print(matrix[1][2]); // 0
```

Indexed Variables

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

- Indexed variables: index for first dimension first (row), then second dimension (column)

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

```
matrix[1][1] = matrix[1][0] + 3;
```

Lengths of a 2-dimensional array

- `matrix.length` is the size of the first dimension (number of rows)
- `matrix[i].length` is the size of the row `i` (i.e., the number of columns in row `i`)

- Example:

```
for (int i = 0; i < matrix.length; i++)  
    for (int j = 0; j < matrix[i].length; j++)  
        matrix[i][j] = (int) (Math.random() * 1000);
```

Two-dimensional Array Lengths

	0	1	2	3	4
0					
1					
2					
3					
4					

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

`matrix.length?` 5

`matrix[0].length?` 5

	0	1	2	3	4
0					
1					
2		7			
3					
4					

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

	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}  
};
```

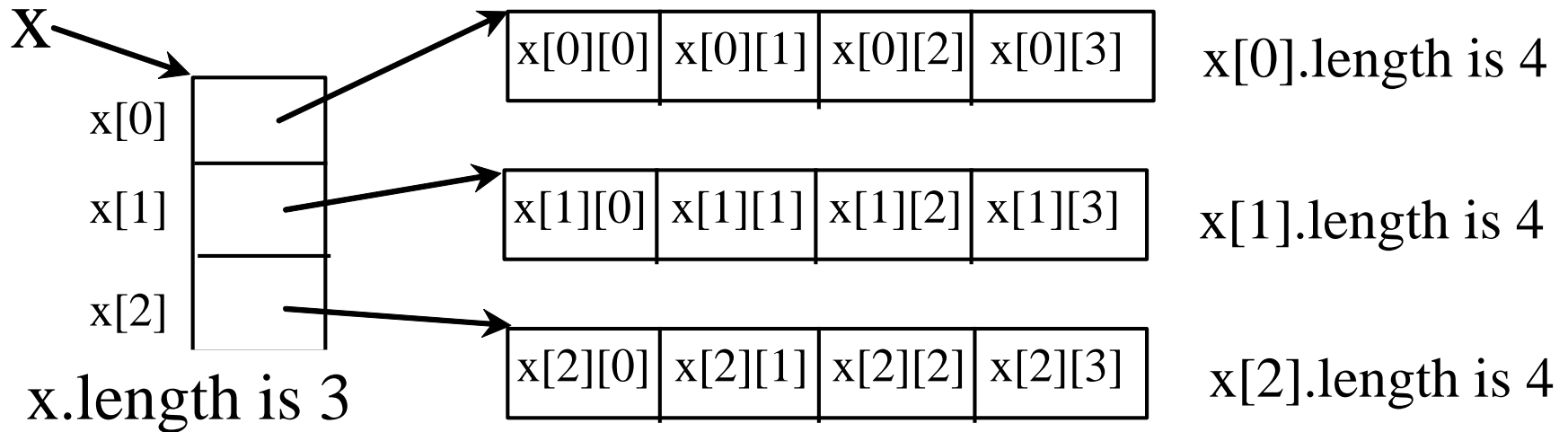
`array.length?` 4

`array[0].length?` 3

Two-dimensional Arrays

Storage

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



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[][] 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

- A ragged array is an array where rows can have different lengths:

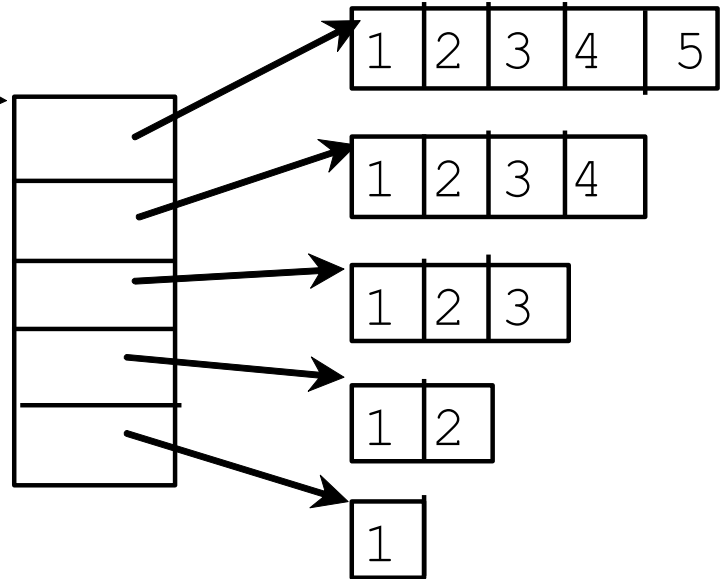
```
int[][] matrix = {  
    {1, 2, 3, 4, 5},  
    {2, 3, 4, 5},  
    {3, 4, 5},  
    {4, 5},  
    {5}  
};
```

```
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

Storing a ragged array:

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



Initializing 2D arrays with input values

```
int matrix[][] = new int[10][10];
java.util.Scanner input =
    new java.util.Scanner(System.in);
System.out.println("Enter " +
    matrix.length + " rows and "
    + matrix[0].length + " columns: ");
for (int row = 0; row < matrix.length;
    row++) {
    for (int column = 0;
        column < matrix[row].length;
        column++) {
        matrix[row][column] = input.nextInt();
    }
}
```

Initializing 2D arrays with random values

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

Printing 2D arrays

```
for(int row=0; row<matrix.length; row++){  
    for(int column = 0;  
        column<matrix[row].length;  
        column++){  
        System.out.print(matrix[row][column]  
            + " ");  
    }  
    // new line after each row  
    System.out.println();  
}
```

Printing 2D arrays with **for-each**

```
for(int[] row:matrix) {  
    for(int elem:row) {  
        System.out.print(elem + " ");  
    }  
    // new line after each row  
    System.out.println();  
}
```


Summing all elements

```
int total = 0;
for(int row = 0; row < matrix.length;
    row++) {
    for(int column = 0;
        column < matrix[row].length;
        column++) {
        total += matrix[row][column];
    }
}
```

Summing all elements with **for-each**

```
int total = 0;
for (int[] row:matrix) {
    for (int elem:row) {
        total += elem;
    }
}
```

Summing elements by column

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

2D Random shuffling

```
for (int i = 0; i < matrix.length; i++) {  
    for (int j = 0; j < matrix[i].length; j++) {  
        int i1 = (int) (Math.random() * matrix.length);  
        int j1 = (int) (Math.random() * matrix[i1].length);  
        // Swap matrix[i][j] with matrix[i1][j1]  
        int temp = matrix[i][j];  
        matrix[i][j] = matrix[i1][j1];  
        matrix[i1][j1] = temp;  
    }  
}
```

N-dimensional Arrays

- Not just only 2-dimensional arrays!!!
- N-dimensional data structures example:
 - grades on multiple dimensions (5 courses, 30 students, 25 lab grades):

```
int[][][] scores = new int[5][30][25];
```

```
scores[0][0][0] = 1;
```

N-dimensional Arrays

- Printing N-dimensional Arrays:

```
int[][] [] m = new int[2][3][4];
```

```
for(int[] [] slice:m) {  
    for(int[] row:slice) {  
        for(int e:row)  
            System.out.print(e + " ");  
        System.out.println();  
    }  
    System.out.println();  
}
```

N-dimensional Arrays

- Printing N-dimensional Arrays:

```
int[][] [] m = new int[2][3][4];
```

```
for(int i=0; i<m.length; i++) {  
    for(int j=0; j<m[i].length; j++) {  
        for(int k=0; k<m[i][j].length; k++)  
            System.out.print(m[i][j][k] + " ");  
        System.out.println();  
    }  
    System.out.println();  
}
```