

Introduction to Multi-Dimensional Array

Hui Chen

Department of Computer & Information Science

Brooklyn College

Objectives

- To give examples of representing data using two-dimensional arrays (§8.1).
- To declare variables for two-dimensional arrays, create arrays, and access array elements in a two-dimensional array using row and column indexes (§8.2).
- To program common operations for two-dimensional arrays (displaying arrays, summing all elements, finding the minimum and maximum elements, and random shuffling) (§8.3).
- To pass two-dimensional arrays to methods (§8.4).
- To write a program for grading multiple-choice questions using two-dimensional arrays (§8.5).
- To solve the closest-pair problem using two-dimensional arrays (§8.6).
- To check a Sudoku solution using two-dimensional arrays (§8.7).
- To use multidimensional arrays (§8.8).

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
```

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

Declare/Create Two-dimensional Arrays: Examples

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

or

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

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

```
for (int i = 0; i < matrix.length; i++)
```

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

```
        matrix[i][j] = (int)(Math.random() * 1000);
```

```
double[][] x;
```

Illustrating Two-dimensional Array

	[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)

Using Shorthand Notations

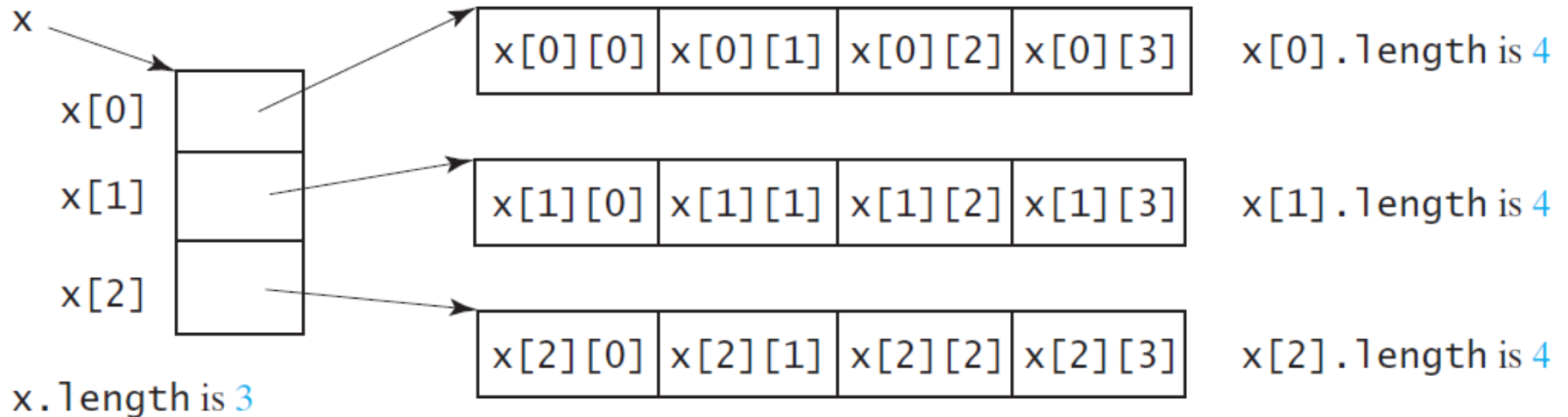
```
int[][] array = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9},  
    {10, 11, 12}  
};
```

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];
```



Ragged Arrays

- In Java, the rows can have different lengths. Such an array is known as a ragged array

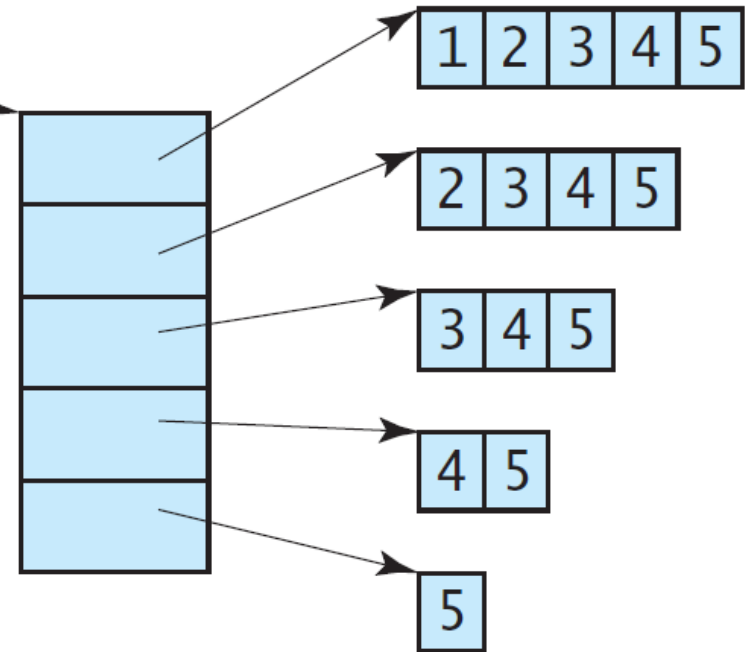
```
int[][] matrix = {  
    {1, 2, 3, 4, 5},  
    {2, 3, 4, 5},  
    {3, 4, 5},  
    {4, 5},  
    {5}  
};
```

```
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
```


Illustrating Ragged Arrays

Array of Arrays

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



Questions?

- Concept of two-dimensional arrays?

Processing Two-Dimensional Arrays

- Several common processing examples
 1. Initializing arrays with input values
 2. Printing arrays
 3. Summing all elements
 4. Summing all elements by column
 5. Which row has the largest sum
 6. Finding the smallest index of the largest element
 7. *Random shuffling*

Initializing arrays with input values

```
java.util.Scanner input = new 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 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 arrays

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


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[i].length);
        // Swap matrix[i][j] with matrix[i1][j1]
        int temp = matrix[i][j];
        matrix[i][j] = matrix[i1][j1];
        matrix[i1][j1] = temp;
    }
}
```

Questions?

Programming Examples

- Problem: Grading Multiple-Choice Test
- Problem: Finding Two Points Nearest to Each Other
- Problem: Checking Whether a Sudoku Solution Is Correct

Problem: Grading Multiple-Choice Test

Students' answer

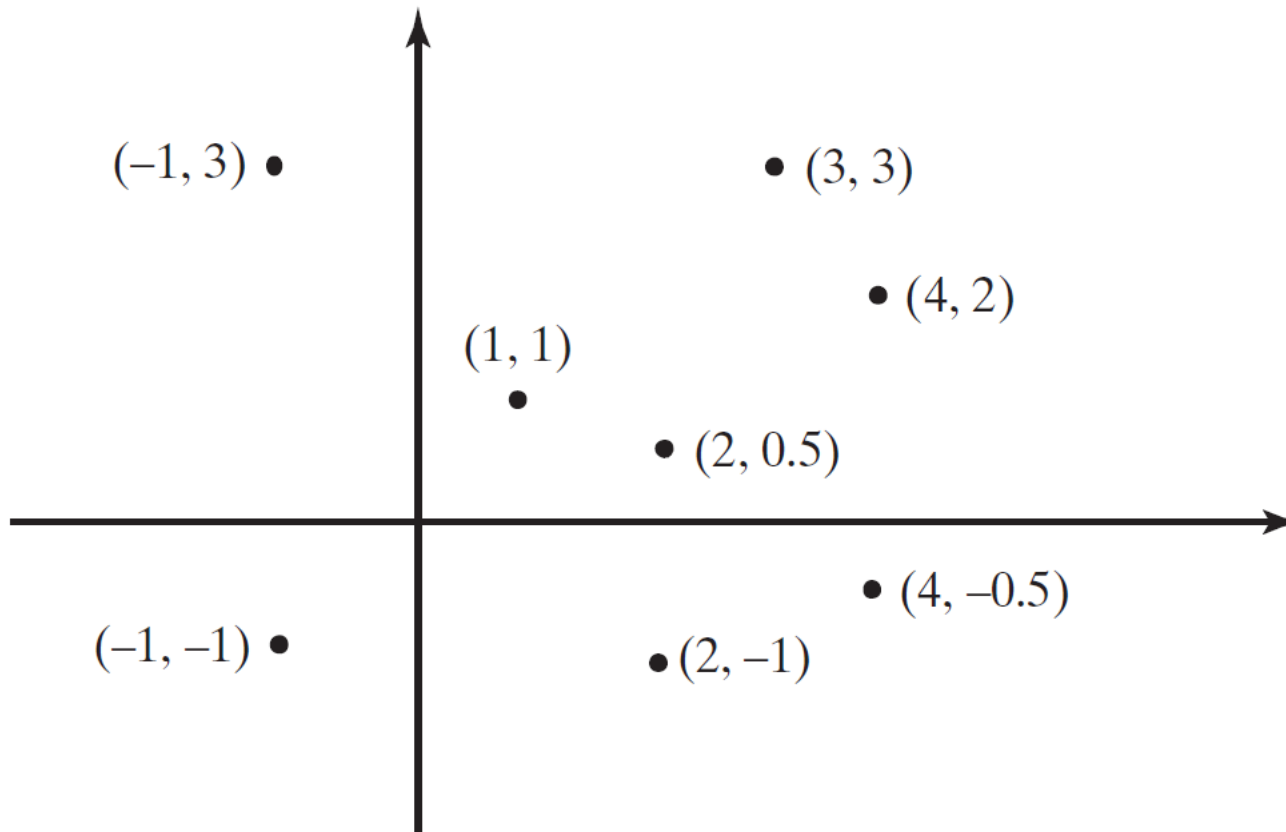
	0	1	2	3	4	5	6	7	8	9
Student 0	A	B	A	C	C	D	E	E	A	D
Student 1	D	B	A	B	C	A	E	E	A	D
Student 2	E	D	D	A	C	B	E	E	A	D
Student 3	C	B	A	E	D	C	E	E	A	D
Student 4	A	B	D	C	C	D	E	E	A	D
Student 5	B	B	E	C	C	D	E	E	A	D
Student 6	B	B	A	C	C	D	E	E	A	D
Student 7	E	B	E	C	C	D	E	E	A	D

Objective: write a program that grades multiple-choice test.

Key to the Questions:

	0	1	2	3	4	5	6	7	8	9
Key	D	B	D	C	C	D	A	E	A	D

Problem: Finding Two Points Nearest to Each Other



	x	y
0	-1	3
1	-1	-1
2	1	1
3	2	0.5
4	2	-1
5	3	3
6	4	2
7	4	-0.5

Problem: Solving Sudoku

- Every row, column, 3x3 box contains the numbers 1 to 9

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6							
			4	1	9			5
				8			7	9

A 9x9 Sudoku grid is shown with a 3x3 box highlighted in red. The numbers 1 through 9 are highlighted in yellow in each row, column, and 3x3 box. The grid is as follows:

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Questions?

Multidimensional Arrays

- A generalization of 2-dimensional arrays

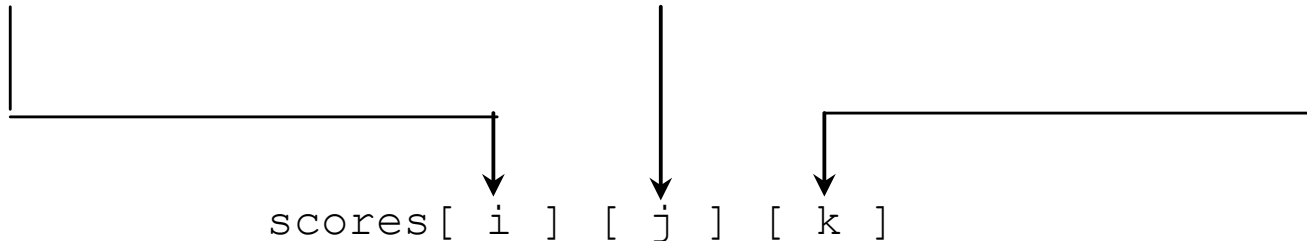
Example: 3-dimensional Array

```
double[][][] scores = {  
    {{7.5, 20.5}, {9.0, 22.5}, {15, 33.5}, {13, 21.5}, {15, 2.5}},  
    {{4.5, 21.5}, {9.0, 22.5}, {15, 34.5}, {12, 20.5}, {14, 9.5}},  
    {{6.5, 30.5}, {9.4, 10.5}, {11, 33.5}, {11, 23.5}, {10, 2.5}},  
    {{6.5, 23.5}, {9.4, 32.5}, {13, 34.5}, {11, 20.5}, {16, 7.5}},  
    {{8.5, 26.5}, {9.4, 52.5}, {13, 36.5}, {13, 24.5}, {16, 2.5}},  
    {{9.5, 20.5}, {9.4, 42.5}, {13, 31.5}, {12, 20.5}, {16, 6.5}}  
};
```

Which student

Which exam

Multiple-choice or essay



Example Problems

- Problem: Calculating Total Scores
- Problem: Weather Information

Problem: Calculating Total Scores

- Objective: write a program that calculates the total score for students in a class.
 - Suppose the scores are stored in a three-dimensional array named scores.
 - The first index in scores refers to a student, the second refers to an exam, and the third refers to the part of the exam.
 - Suppose there are 7 students, 5 exams, and each exam has two parts--the multiple-choice part and the programming part. So, `scores[i][j][0]` represents the score on the multiple-choice part for the *i*'s student on the *j*'s exam.
 - Your program displays the total score for each student

Problem: Weather Information

- Objective: Your task is to write a program that calculates the average daily temperature and humidity for the 10 days.
 - Suppose a meteorology station records the temperature and humidity at each hour of every day and stores the data for the past ten days in a text file named `weather.txt`.
 - Each line of the file consists of four numbers that indicate the day, hour, temperature, and humidity.

Data Representation for Weather Information

- Using 3 dimensional array
 - `weather[i][j][k]`: i for day, j for hour, and k for humidity or temperature

```
1 1 76.4 0.92
1 2 77.7 0.93
...
10 23 97.7 0.71
10 24 98.7 0.74
```

(a)

```
10 24 98.7 0.74
1 2 77.7 0.93
...
10 23 97.7 0.71
1 1 76.4 0.92
```

(b)