15. Multidimensional Arrays

Java

Fall 2009
Instructor: Dr. Masoud Yaghini
Multidimensional Arrays

Outline

- Declaring and Creating of Two-Dimensional Arrays
- Ragged Arrays
- Simple Processing on Two-Dimensional Arrays
- Three-Dimensional Arrays
- References
Declaring and Creating of Two-Dimensional Arrays
Multidimensional Arrays

Declaring and Creating of 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];
// This style is allowed, but not preferred
Multidimensional Arrays

Declaring and Creating of Two-Dimensional Arrays

- Example:
  ```java
  int[][] matrix = new int[10][10];
  or
  int matrix[][] = new int[10][10];
  matrix[0][0] = 3;
  ```
You can also use an array initializer to declare, create, and initialize a two-dimensional array:

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

Equivalent to:

```java
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;
```
Multidimensional Arrays

Declaring and Creating of Two-Dimensional Arrays

(a) matrix = new int[5][5];

(b) matrix[2][1] = 7;

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

- Example:
  - MultiDimArrayDemo.java

- The output of program:
  Mr. Smith
  Ms. Jones
Multidimensional Arrays

Obtaining the Lengths of Two-Dimensional Arrays

- A two-dimensional array is actually an array in which each element is a one-dimensional array.

```java
int[][] x = new int[3][4];
```
Ragged Arrays
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:

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

- `triangleArray.length` is 5, `triangleArray[0].length` is 5
- `triangleArray[1].length` is 4, `triangleArray[2].length` is 3
- `triangleArray[3].length` is 2, and `triangleArray[4].length` is 1
Multidimensional Arrays

Ragged Arrays

- you can create a ragged array using the syntax that follows:

```java
int[][] triangleArray = new int[5][];
triangleArray[0] = new int[5];
triangleArray[1] = new int[4];
triangleArray[2] = new int[3];
triangleArray[3] = new int[2];
triangleArray[4] = new int[1];
```
Simple Processing on Two-Dimensional Arrays
Multidimensional Arrays

Processing Two-Dimensional Arrays

- Suppose an array matrix is declared as follows:
  ```java
  int[][] matrix = new int[10][10];
  ```

- Initializing arrays with random values:
  ```java
  for (int row = 0; row < matrix.length; row++) {
    for (int column = 0; column < matrix[row].length; column++) {
      matrix[row][column] = (int)(Math.random() * 100);
    }
  }
  ```
Multidimensional Arrays

Processing Two-Dimensional Arrays

- **Printing arrays:**
  ```java
  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:**
  ```java
  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:

```java
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);
}
```
Which row as the largest sum?

```java
int maxRow = 0;
int indexOfMaxRow = 0; // Get sum of the first row in maxRow
for (int column = 0; column < matrix[0].length; column++) {
    maxRow += matrix[0][column];
}
for (int row = 1; row < matrix.length; row++) {
    int totalOfThisRow = 0;
    for (int column = 0; column < matrix[row].length; column++) {
        totalOfThisRow += matrix[row][column];
        if (totalOfThisRow > maxRow) {
            maxRow = totalOfThisRow; indexOfMaxRow = row;
        }
    }
}
System.out.println("Row " + indexOfMaxRow + " has the maximum sum" + " of " + maxRow);
```
Example: Grading a Multiple-Choice Test

- Suppose there are eight students and ten questions, and the answers are stored in a two-dimensional array.
- Each row records a student's answers to the questions.
- Objective: write a program that grades multiple-choice test.
Example: Grading a Multiple-Choice Test

- **Grading a Multiple-Choice Test**
  - [GradeExam.java](#)

- **The output:**
  
  Student 0's correct count is 7
  Student 1's correct count is 6
  Student 2's correct count is 5
  Student 3's correct count is 4
  Student 4's correct count is 8
  Student 5's correct count is 7
  Student 6's correct count is 7
  Student 7's correct count is 7
Three-Dimensional Arrays
An Example of Three-Dimensional Arrays

- 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 a part of the exam.
- Suppose there are seven students, five exams, and each exam has two parts: a multiple-choice part and a programming part.
- The program calculates the total score for the students in a class.
Multidimensional Arrays

An Example of Three-Dimensional Arrays

- **TotalScore.java**

  - The output:
    
    Student 0's score is 160.0
    Student 1's score is 163.0
    Student 2's score is 147.4
    Student 3's score is 174.4
    Student 4's score is 201.4
    Student 5's score is 181.4
    Student 6's score is 165.9
References
Multidimensional Arrays

References


The End