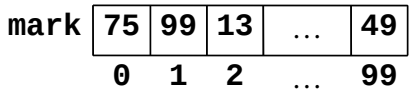
CS133: Developing Programming Principles

Lecture 13

Multi-Dimensional Arrays

Motivation

- So far, our arrays have been one-dimensional:
  `int[] mark = new int[100];`



- Sometimes it may make sense to store information in more than one dimension.

Motivation continued

- Examples:
  - Periodic table of elements
  - Prices of flights between cities
  - Matrix of numbers
  - All assignment grades for all students in a course.

Two-dimensional arrays

- Tables (or tabular data) map nicely to two-dimensional arrays.
- Later, we will see how to generalize to any (fixed) number of dimensions.

Declaring a 2D array

- Syntax:

  ***Type*`[][]` *ArrayName* `=`**
  `new` *`Type`*`[`*`Length_1`*`][`*`Length_2`*`];`

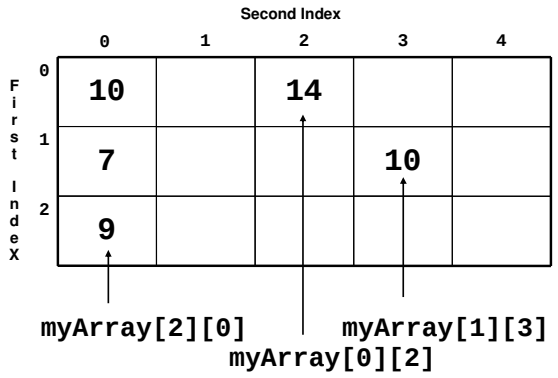
- Example:

```
int[][] myArray = new int[3][5];
```


Accessing the array

```
myArray[0][0] = 10;
myArray[1][0] = 7;
myArray[0][2] = 14;
myArray[1][3] = 10;
myArray[2][0] = 9;
```

Graphical representation of a 2D array



Size of a multi-dimensional array

- Suppose you have a 2D array.
- How could you determine the dimensions of the array?
- 2D arrays are actually 1D arrays where each entry is a 1D array.

Size

- Thus, to find the first dimension size:
  `myArray.length`
- To find second dimension size
  `myArray[0].length`
- Example:
  > **`int[][] myArray = new int[3][5];`**
  > **`System.out.println(myArray.length);`**
  Result: _____
  > **`System.out.println(myArray[0].length);`**
  Result: _____

Template for iterating through 2D arrays

```
for (i = 0; i < max index of 1st dimension; i++)
{
 for (j = 0; j < max index of 2nd dimension; j++)
 {
   do stuff with myArray[i][j];
 }
}
```

Iterating through a 2D array

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


More than 2 dimensions

- Imagine you have
  - A collection of 3 filing cabinets, which have
  - A collection of 4 drawers each, which have
  - A collection of 100 files each, which have
  - A collection of 200 pieces of paper each
  - Each piece of paper having 10 numbers

Example continued

- Write the declaration for this array, called fileSystem.
- Write a for loop to put the numbers from 1 to 10 on each page of the array.

Solution

Passing multidimensional arrays

- Similar structure to passing one-dimensional arrays.
- Example: using fileSystem as a parameter

```
public void numZeroOnPages(
        int[][][][][] paramArray)
{ … }
```


Returning a multi-dimensional array

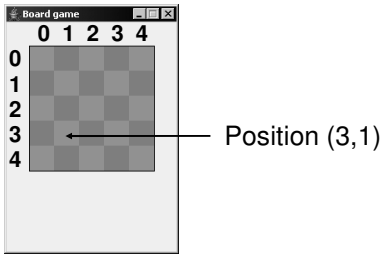
- Again, similar to one-dimensional arrays.
- Example: Suppose we wish to return a 3-dimensional array

```
public int[][][] someMethod()
{ … }
```

Board class and arrays

- You have already seen a 2D array: the Board.

```
Board grid = new Board(5,5);
```



Inside the Board

```
public Board(int rows, int cols) {
  // Window behaviour stuff omitted
  grid = new Color[cols][rows];
  for (int i = 0; i < cols; i++) {
    for (int j = 0; j < rows; j++) {
      grid[i][j] = invisible(i, j);
    }
  }
}
public Board(int cols) {
  this(1, cols);
}
```

Summary

- Multi-dimensional arrays
  - Declaration
  - Structure
  - Accessing and altering
  - Passing
  - Returning
  - Applications **`Board`** class

ERROR: undefined
OFFENDING COMMAND:

STACK: