2D Arrays
(Savitch, Chapter 7.5)

TOPICS

• Multidimensional Arrays
• 2D Array Allocation
• 2D Array Initialization
• TicTacToe Game

Declaring and initializing 2D arrays

```
// setting up a 2D array
final int M=3, N=4;
int [][] matrix = new int [M][N];
for(int i=0; i<M; i++)
    for (int j=0; j<N; j++)
        matrix[i][j] = fileScanner.nextInt();
```

```
// printing from a 2D array
final int M=100, N=200;
int [][] matrix = new int [M][N];
for(int i=0; i<M; i++)
    for (int j=0; j<N; j++)
        System.out.println(matrix[i][j] + " ");
```

```
// First write code to initialize the matrices m1 and m2 as an exercise
int [][] m3 = new int [M][N];
for(int i=0; i<M; i++)
    for (int j=0; j<N; j++)
        m3[i][j] = m1[i][j] + m2[i][j];
```

The numbers in each cell represent the indices

m1

```
0.0 0.1 0.2 0.3
1.0 1.1 1.2 1.3
2.0 2.1 2.2 2.3
```

m2

```
0.0 0.1 0.2 0.3
1.0 1.1 1.2 1.3
2.0 2.1 2.2 2.3
```

m3

```
0.0 0.1 0.2 0.3
1.0 1.1 1.2 1.3
2.0 2.1 2.2 2.3
```

Printing 2D arrays

```
// printing from a 2D array
final int M=100, N=200;
int [][] matrix = new int [M][N];
for(int i=0; i<M; i++)
    for (int j=0; j<N; j++)
        System.out.println(matrix[i][j] + " ");
```

Adding two matrices

```
// First write code to initialize the matrices m1 and m2 as an exercise
int [][] m3 = new int [M][N];
for(int i=0; i<M; i++)
    for (int j=0; j<N; j++)
        m3[i][j] = m1[i][j] + m2[i][j];
```

The numbers in each cell represent the indices

m1

```
0.0 0.1 0.2 0.3
1.0 1.1 1.2 1.3
2.0 2.1 2.2 2.3
```

m2

```
0.0 0.1 0.2 0.3
1.0 1.1 1.2 1.3
2.0 2.1 2.2 2.3
```

m3

```
0.0 0.1 0.2 0.3
1.0 1.1 1.2 1.3
2.0 2.1 2.2 2.3
```
More on 2D arrays

- `int[][] matrix = new int[3][4];`
- What is `matrix.length`? It is 3
- What is `matrix[0].length`? It is 4
  - So is `matrix[1].length`, `matrix[2].length`, and `matrix[3].length`
- You can access a particular row using `matrix[i]` where `i` refers to the row number between 0 and 2
- Each row is a one-dimensional array
- You cannot access a column like that 😞
- Exercises:
  - Write code that subtracts one matrix from another
  - Write code that transposes the given matrix

Review (Java)

- Assignments & expressions
- Sequential control: if & switch
- Looping control: while, for, do
- Organization: classes & methods
- Tools: Eclipse & debugging

Why? So you can program…

Programming

- ... but programming isn’t about syntax
  - You can program in many languages
- Programming is about problem solving
  - Problem definition/refinement
  - Problem decomposition
  - Managing complexity

Challenge Problem

- So here is a problem to be worked through together:
  - Write a person versus computer TicTacToe game.
Challenge Problem

• Write a TicTacToe game
  – Machine goes first, plays ‘X’
  – Print the board before every user move
  – User plays ‘O’, specifies moves by coordinate
    • 0..2, 0..2.
  – Machine selects random, legal moves
  – Program knows when game is over

Decomposition

• Game board
  – State
• Player moves
  – User I/O
• Computer moves
  – Select random, legal moves
• Manage game
  – Alternate turns until end

Further Decomposition

• Game board
  – Maintain board state
    • 2D array makes sense
  – Mark board square
    • Add an X’ or ‘O’ at row, col
    • Check that row, col are empty
  – Print entire board
    • Show the state of the board
  – Detect game over

Code (Part 1a)

• Focusing on the game state:
  – Board state needed by multiple subtasks
  – Good candidate for an instance variable
• Initialize the board : method
• Mark a square : method
• Print the board : method
• Detect game over : methods
Public vs Private

• What is the difference between public & private variables/methods?
• If something is public, it can be accessed by other objects
  – Think of the String object
  – If it’s length() method weren’t public, you couldn’t use it!
• If something is private, only methods of the same class can access it
  – Note that if something is public, it can be changed at any point (more error checking!)
• General rule: top-level methods should be public, everything else should be private

Code (Part 1b)

• markSquare() & printBoard() are straightforward
  – Note error checking, only valid moves allowed
  – print just iterates the 2D array
• What about gameOver()?
  – When is a game of tic-tac-toe over?
    • When there is a row of X’s or O’s…
    • … or a column
    • … or a diagonal
    • … or the board is full (tie game)
  – So may require decomposition
    • Leading to more methods…

Stop!
Do not pass go or collect $200…

• DO NOT write the whole program at all once
  – Too hard to debug that way
  – Test each piece separately
• Write a temporary main function
  – Have it initialize the board, mark a square, print the result. Does it work?
  – Have it test end of game scenarios too.
  – Then comment out the test code
  – It’s not part of the final product.
    • Think of it like scaffolding…

Code (Part 2)

• OK, now we need to get moves from the user
  – Print a prompt
  – Read in coordinates
  – Call markSquare()
• Probably doesn’t need further decomposition
• But does need to be tested!
Code (Part 3)

• Machine move: picked at random
  – Java has a Random class
    • Generates a stream of pseudo-random numbers
  – Pick a row and col at random
    • Between 0 and 2
  – Check if legal. If not, pick another
  – What will happen if board is full?
• Don’t forget testing....

Initialization

• Problem: which method allocates the board?
  – How about Scanner and Random?
• We want instance variables initialized before any other method is called
  – But we haven’t decided which method will be called first
  – Even if we had, might change when the code is modified
• Solution: A constructor is a method that is called by new when an object is created.

Constructors

• The syntax for constructors is unique
  – Constructors take parameters, but they never return a value
  – The constructor name is always the same as the class name
  – The default constructor has no parameters, but we can add them
  – The constructor is generally used to initialize class instance variables

Code (Part 4)

• Now we’ve done everything but play the game!
• The game is the main function
  – What should happen when the class is executed
  – Hence, public static void main(String[] args)
• The game depends on instance variables, so main needs to instantiate TicTacToe
Eclipse Demo

• Write the program for TicTacToe
• Will be posted on the course website