Arrays

2-D Arrays

Chapter 8+

1-Dimension

Example: 20, floating point values

- **float[ ] myNums;**   //declare array
- **myNums = new float[20];**   //create array
- Average the numbers
- Find the largest value

Example: 50 Phone numbers (555-1212)

- **String[ ] phoneList;**   //declare array
- **phoneList = new String[50];** //create array (no strings yet)
- then, perhaps, read phone numbers from a file

1-Dimension

Example: 20 JPanels

JPanel[ ] shapes;   //declare array   (an array of JPanels)

- shapes = new JPanel[20];

Create <u>**space**</u> for 20 JPanels

No JPanels exist!!

Each array location still needs to be created and initialized with appropriate values as needed; for example…

```
for (int i=0; i< 20; i++) {
 shapes[i] = new JPanel(…);
 shape[i].setBackground (c);
 shape[i]. ???
}
```

1-Dimension

Initializer Lists

Declare and initialize all at once

- **int [ ] ep = {1,2,3,5,7,11,13,17,19,23};**
- **char [ ] vowels = {‘A’, ‘E’, ‘I’, ‘O’, ‘U’ };**
- **String [ ] beatles ={“John”,”Paul”,”George”,”Ringo”};**

1-D Project

Maury Povich (or Dr. Phil or …) selects program titles by choosing a subject phrase (“People who”, “Women who”, “Men who” etc) and combine that with a verb (“love”, “hate”, “sleep with”, “cheat on”, etc) and finally another noun phrase (“their sister”, “their spouse’s sister”, “their girlfriend”, “their boyfriend”, etc)

Expand the choices of each to 5 and then write a program to *randomly* generate a completely new program title each time it is run.

Parameter lists

Arrays as parameters

- `public int sumValues (int[] vals)`
  - `vals` is the ***formal parameter***
  - `sumValues` gets a reference to an array ‘vals’
  - how many items in the array?
  - `vals.length`

Calling method - assumes x is an array of integers

- x is the ***actual parameter***
- `int mySum = sumValues (x);`

Parameter practice

Declare a method that takes an array of positive floats and returns the maximum value in the array

Declare a method that takes an array of int's and returns a sorted array of int's

1-D Array

Write a method that takes a 1-D array of integers and finds the sum of those values

Parameters (aside)

Variable Length Parameter Lists

What if I want to sum the values like this

```
 int sum1 = sumValues (2,43,9,12);
```

or

```
 int sum2 = sumValues (2,4,6,7,8,9,3,5);
```

A Java method can be defined to accept a varying number of parameters! (chap 8.5)

Variable Length Parameter Lists

```
public sumValues (int … list)
```

the ellipsis means it will accept any number of ints

list is automatically built as an array

Varying parameters MUST be the last formal parameter

```
public void foo (int x, String … name)
```

Example

```
//class to demonstrate variable length parameter lists
public class Family
{
  private String [ ] members;
  //constructor: set up names of family members
  public Family (String … names)
  {
      members = names;
  }
  //string representation of this family
  public String toString ( )
  {
     String result = “”;
     for (String aname: members)
        result += aname + “\n”;
     return result;
  }
}
```

1-D Array Quiz

- **int [ ] nums = {2, 5, 9, 8, 10, 20, 30};**

What is

nums.length

nums [1]

nums [2] + nums [4]

nums [2 + 4]

nums [nums [0] * 2]

2-Dimension

Grid, map, game boards, images, etc.

array of arrays

- **int[ ][ ] g = new int[20][100];**

technically: array of 20 element, each of 100 ints

g.length ??

g[0].length ??

2-D Examples

“Rows and columns”

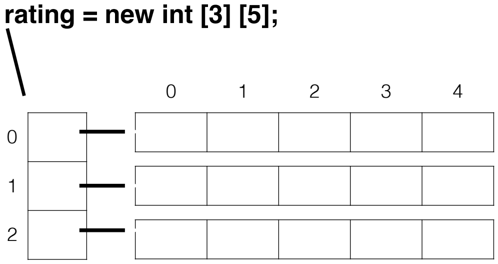
Spreadsheet of user movie ratings

rating [0] [4]   user 0; movie 4

2-D Array

Array of references to 1D arrays

- **int [ ] [ ] rating = new int [3] [5];**



2-D Arrays

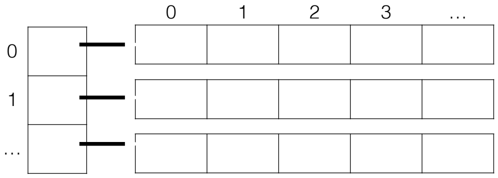
Write a method that is given a 2-D array of user movie ratings. Return the average user rating for all movies

Write a method that is given a 2-D array of user movie ratings and a “threshold” value. Return the number of ratings above the threshold

2-D Arrays of Objects

```
House [][] h;
h = new House[SIZE_A][SIZE_B];
for (int i=0; i<SIZE_A; i++)
   for (int j=0; j < SIZE_B; j++)
      h[i][j] = new House();
```

null



2-D Arrays

```
Illustrates the “array of arrays” concept
int[ ][ ] myArray = { {236, 189, 189,   0},
                      {236,  80, 189, 189},
                      {236,   0, 189,  80},
                      {236, 189, 189,  80} };
```

-Better to use a file to read values

```
String[][] myStudents = {{“John”,”Paul”},
      {“Amber”,”Bennett”,”Carl”,”Dwayne”},
      {“Helen”,”Isaac”,”Joe”}};
```

Row length: `myStudent[r].length`