Searching

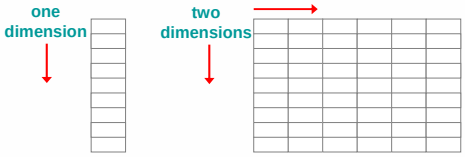
October 16, 2006

ComS 207: Programming I (in Java)
Iowa State University, FALL 2006
Instructor: Alexander Stoytchev

Quick review of last lecture

Two-Dimensional Arrays

- A *one-dimensional array* stores a list of elements
- A *two-dimensional array* can be thought of as a table of elements, with rows and columns



Two-Dimensional Arrays

- To be precise, in Java a two-dimensional array is an array of arrays
- A two-dimensional array is declared by specifying the size of each dimension separately:

```
int[][] scores = new int[12][50];
```

- A array element is referenced using two index values:

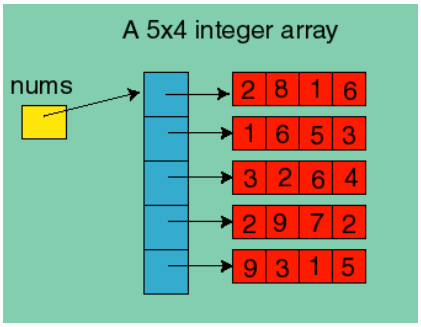
```
value = scores[3][6]
```

- The array stored in one row can be specified using one index

Arrays in Java

- Java represents 2D arrays as an array of arrays!
- In other words, a 2D integer array is really a 1D array of references to 1D integer arrays.
- The concept generalizes to N-dimensions

Anatomy of a 2D Array

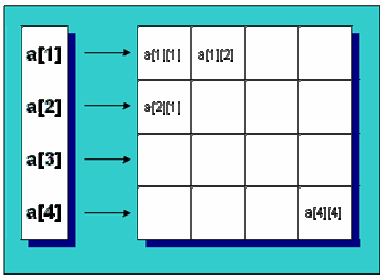


[http://www.willamette.edu/~gorr/classes/cs231/lectures/chapter9/arrays2d.htm]

Two-Dimensional Arrays

Expression	Type	Description
`table`	`int[][]`	2D array of integers, or array of integer arrays
`table[5]`	`int[]`	array of integers
`table[5][12]`	`int`	integer

Example of a regular 2D array



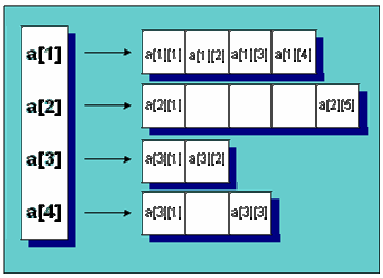
Note: In Java the first index should be 0 not 1!

[http://livedocs.macromedia.com/coldfusion/5.0/Developing_ColdFusion_Applications/arrayStruct2.htm]

Multidimensional Arrays

- An array can have many dimensions – if it has more than one dimension, it is called a *multidimensional array*
- Each dimension subdivides the previous one into the specified number of elements
- Each dimension has its own `length` constant
- Because each dimension is an array of array references, the arrays within one dimension can be of different lengths
  - these are sometimes called *ragged arrays*

Example of a Ragged Array



Note: In Java the first index should be 0 not 1!

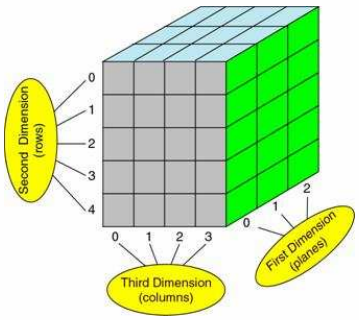
[http://livedocs.macromedia.com/coldfusion/5.0/Developing_ColdFusion_Applications/arrayStruct2.htm]

Two-Dimensional Arrays

Expression	Type	Description
`table`	`int[][]`	2D array of integers, or array of integer arrays
`table[5]`	`int[]`	array of integers
`table[5][12]`	`int`	integer

3D Array Example

3D Arrays



[http://web.ics.purdue.edu/~cs154/lectures/lecture017.htm]

Other Stuff

Arrays as Parameters

- An entire array can be passed as a parameter to a method
- Like any other object, the reference to the array is passed, making the formal and actual parameters aliases of each other
- Therefore, changing an array element within the method changes the original
- An individual array element can be passed to a method as well, in which case the type of the formal parameter is the same as the element type

Java Example: Printing an Array

ArrayList Class

(Take a look at Section 7.7)
(We will come back to it later)

The ArrayList Class

- The `ArrayList` class is part of the `java.util` package
- Like an array, it can store a list of values and reference each one using a numeric index
- However, you cannot use the bracket syntax with an `ArrayList` object
- Furthermore, an `ArrayList` object grows and shrinks as needed, adjusting its capacity as necessary

The ArrayList Class

- Elements can be inserted or removed with a single method invocation
- When an element is inserted, the other elements "move aside" to make room
- Likewise, when an element is removed, the list "collapses" to close the gap
- The indexes of the elements adjust accordingly

The ArrayList Class

- An `ArrayList` stores references to the `Object` class, which allows it to store any kind of object
- See `Beatles.java` (page 405)
- We can also define an `ArrayList` object to accept a particular type of object
- The following declaration creates an `ArrayList` object that only stores `Family` objects

```
ArrayList<Family> reunion = new ArrayList<Family>
```

- This is an example of *generics*, which are discussed further in Chapter 12

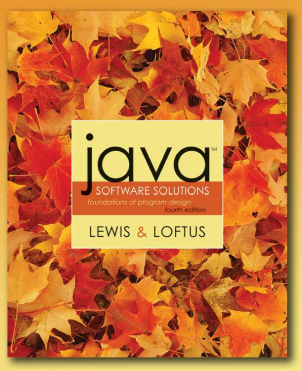
Example: `Beatles.java` (page 405)

ArrayList Efficiency

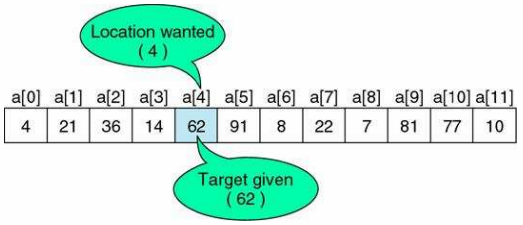
- The `ArrayList` class is implemented using an underlying array
- The array is manipulated so that indexes remain continuous as elements are added or removed
- If elements are added to and removed from the end of the list, this processing is fairly efficient
- But as elements are inserted and removed from the front or middle of the list, the remaining elements are shifted

Searching

Not in the Textbook



Search

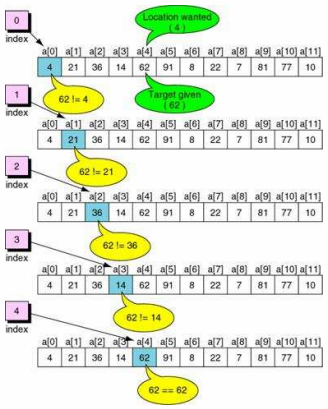


[http://web.ics.purdue.edu/~cs154/lectures/lecture011.htm]

Linear Search

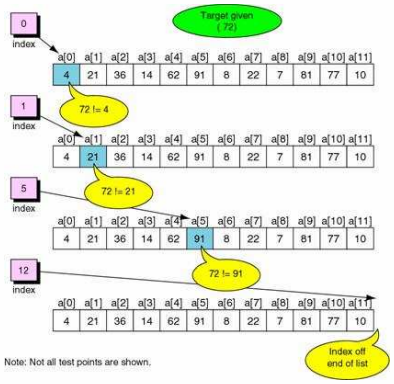
- The most basic
- Very easy to implement
- The array DOESN'T have to be sorted
- All array elements must be visited if the search fails
- Could be very slow

Example: Successful Linear Search



[http://web.ics.purdue.edu/~cs154/lectures/lecture011.htm]

Example: Failed Linear Search



[http://web.ics.purdue.edu/~cs154/lectures/lecture011.htm]

Java Example: Linear Search

Java Example: Finding the minimum number in an array of unsorted integers

THE END