Perfect Size and Oversize Arrays

DEBORAH A. TRYTTEN

CS 1324

Tip of the Day

- One purpose of covering this material is to help you get familiar with what method signatures are saying between the lines
- Reading between the lines is as necessary to understanding programming as it is to understanding literature

Properties of Arrays

- Homogeneous
  - Objects (only String so far) or primitive data
- Sequential
- Contents are zero indexed
- length is data (not a method) and is unit indexed
- Cannot resize without reconstruction

Perfect Size Arrays

- Sometimes you know the perfect size for an array in advance (rare)
- Usual Strategy: Calculate an array size
- If the size has to change, the array must be reconstructed

Example: Perfect Size Array

- String[] split(String source)
  - Splits source using whitespace (use Scanner)
  - “These are four words” becomes {“These”, “are”, “four”, “words”}
- Plan
  - Step through source once to find number of substrings
  - Construct array to perfect size
  - Step through a second time to store data in array
- Implement

Perfect Size Array Logic

- The size of the array must be calculated in the method
  - Array cannot be constructed in advance
- The method must return the array reference
  - Method cannot return the size
  - Array must be perfect size
- No reason to pass array reference as a parameter

Instant Quiz Question 1

- Suppose you wanted to write a method that would create an array with a given number of harmonic numbers (1, 1/2, 1/3, 1/4, 1/5, etc.). Which signature would create a perfect size array?
  a) int harmonicArray(double[] harmonic, int size)
  b) double[] harmonicArray(int size)
  c) double harmonicArray(int size)
  d) double[] harmonicArray(double[] harmonic, int size)

Oversize Arrays

- Sometimes, array size is not known or changes during program execution
- Strategy: Make the array too big, and keep track of the size in a separate variable
- Revisit the split method

```
final int LOTS_OF_SPACE=1000;
String[] word = new String[LOTS_OF_SPACE];
int wordCount=0;
int split(String source, String[] result)
```

Example: Oversize Array

- Splits a String into separate words using whitespace (use Scanner)
  - “These are four words” becomes {“These”, “are”, “four”, “words”}
- Find signature
- Implement method

Oversize Array Logic

- Since a method may change the size of the array, the size must be returned
  - The array reference cannot be returned because only one thing can be returned from a method
  - The array cannot be reconstructed in the method
- The array must be constructed before the method starts
  - The array reference must be a parameter

Instant Quiz Question 2

- Suppose you wanted to write a method that would create an array with a given number of harmonic numbers (1, 1/2, 1/3, 1/4, 1/5, etc.). Which signature would create a oversize array?
  a) int harmonicArray(double[] harmonic, int size)
  b) double[] harmonicArray(int size)
  c) int harmonicArray(int size)
  d) void harmonicArray(double[] harmonic, int size)

Example

- Suppose you want to use an array as a Stack
- What is a stack (Last in, first out)
  - Add to end (called push)
  - Remove from end (called pop)
- Examples
  - Pez dispenser
  - Stack of plates
  - Dr. Trytten’s email ☹
- What would the method signatures be with a perfect size array?

Example Revisited

- Suppose you want to use an oversize array as a Stack
  - Push
  - Pop
- What would the method signatures be with an oversize array?
- Which implementation would you choose (perfect size or oversize)?

Instant Quiz Question 3

- What would the method signature be to reverse the elements in a stack, implemented as a perfect size array
- Assume the array is reconstructed (although this would be unnecessary and wasteful)

a) int[] reverse(int[] source)

b) void reverse(int[] source)

c) int[] reverse ()

d) int reverse (int[] source)

Instant Quiz Question 4

- What would the method signature be to reverse the elements in a stack, implemented as an oversize array
- Assume the array is NOT reconstructed

a) int[] reverse(int[] source, int sourceSize)

b) void reverse(int[] source)

c) int[] reverse ()

d) int reverse (int[] source, int sourceSize)

Think Pair Share

- Suppose you wanted to write a method fill that puts a given number of repetitions of a given integer in a perfect size array
  - What would the method signature be?
- Suppose you wanted to write a method fill that puts a given number of repetitions of a given integer in a oversize array
  - What would the method signature be?

Instant Quiz Question 5

- Suppose we have a file of unknown length that needs to be read into a String[]. Which of the following signatures would produce a oversize array?

a) void read(String[] data, String fileName)

b) String[] read (String fileName)

c) int read(String[] data, String fileName)

d) String[] read (String fileName, int size)

e) String[] read(String fileName, String[] data)

Adventures in the API

- These two implementations exist all over the Java API
- Examine Arrays class
  - Which methods could be used with perfect size arrays?
  - Which methods could be used with oversize arrays?
- What kind of arrays are the split and toCharArray methods using in the String class?
  - Perfect size?
  - Oversize?
  - How can you tell?