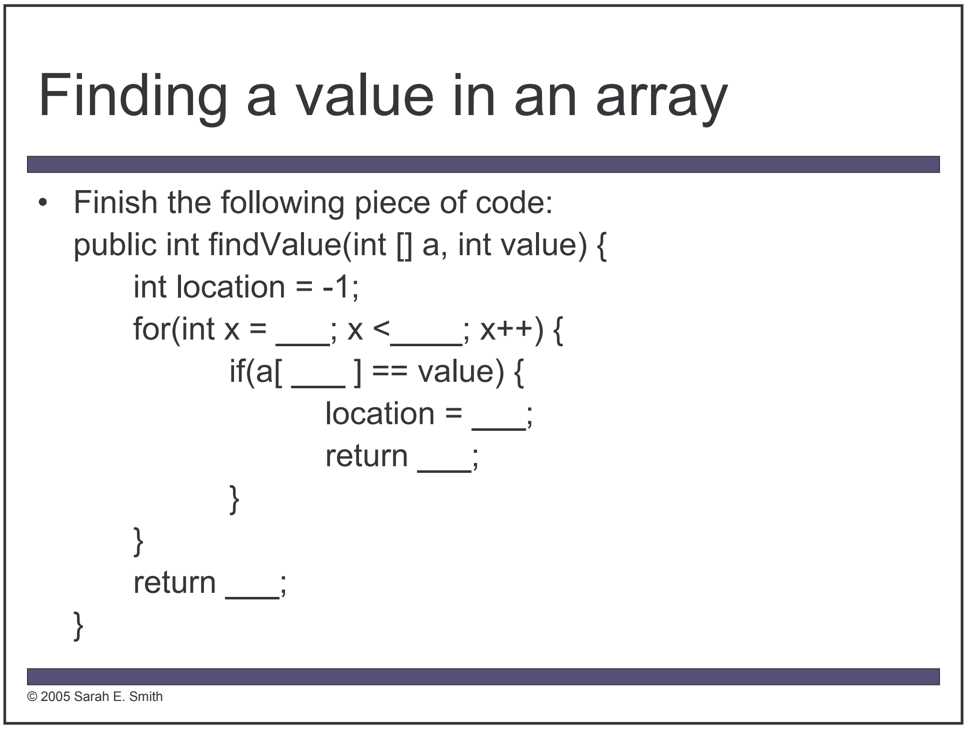
Array Manipulations and 2-D Arrays

CSC 116 – Section 002
April 4, 2005

Finding a value in an array

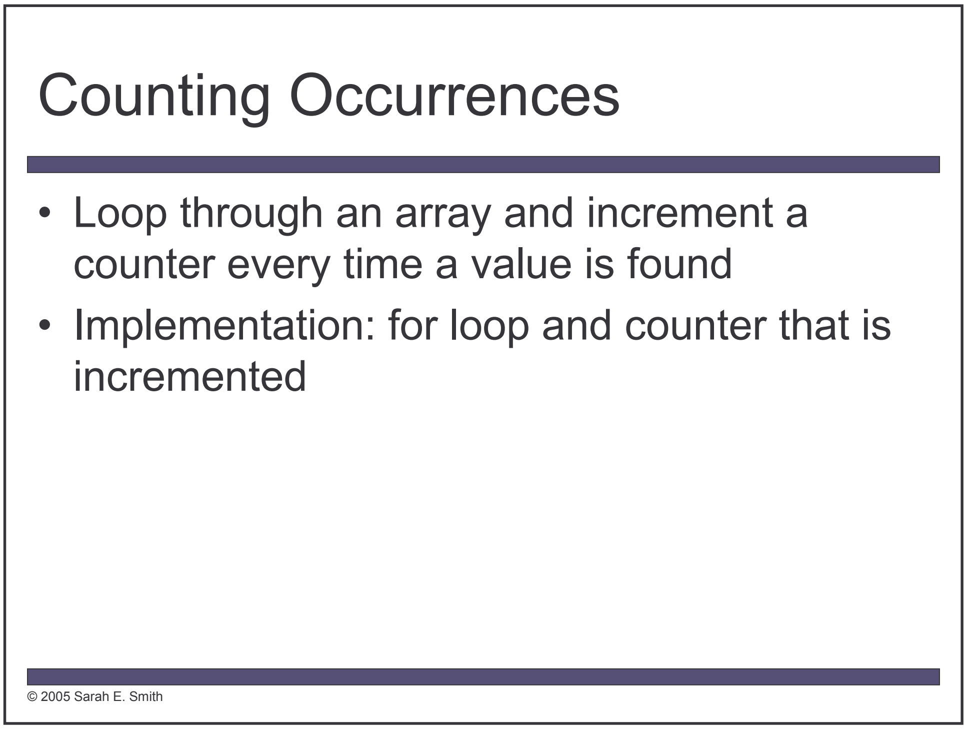
- Loop through the array until the value is found
- Implementation: for loop and an integer to record the location where the first value is found.



Finding a value in an array

- Finish the following piece of code:

```
public int findValue(int [] a, int value) {
      int location = -1;
      for(int x = ___; x <____; x++) {
            if(a[ ___ ] == value) {
                  location = ___;
                  return ___;
            }
      }
      return ___;
}
```



Counting Occurrences

- Loop through an array and increment a counter every time a value is found
- Implementation: for loop and counter that is incremented

Counting Occurrences

- Finish the following piece of code

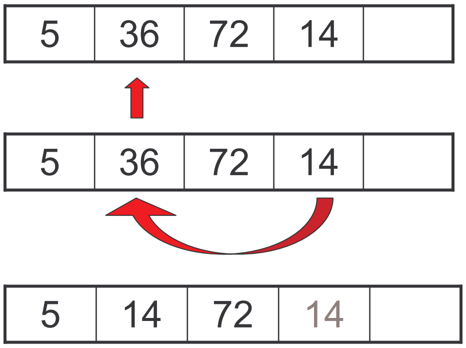
```
public int countOccur(int [] a, int value) {
     int count = 0;
     for(int x = ___; x < ___; x++) {
          if(a [ ___ ] == ___) {
               ___;
          }
     }
     return ___;
}
```

Removing an Element

- An array is a fixed size container, however, you can use a partially filled array to emulate a variable sized container
- Unsorted Array – copy the last element into the position that you want to delete and decrement counter
- Sorted Array – shift elements above the position to be deleted down and decrement counter
  - Must start shifting at the point of removal

Removing an Element (2)

- Removing from an unsorted array



5 36 72 14

Counter = 4

5 36 72 14

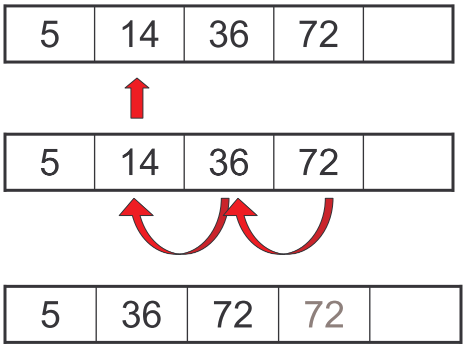
Counter = 4

5 14 72 14

Counter = 3

Removing an Element (3)

- Removing an element from a sorted array



5 14 36 72

Counter = 4

5 14 36 72

Counter = 4

5 36 72 72

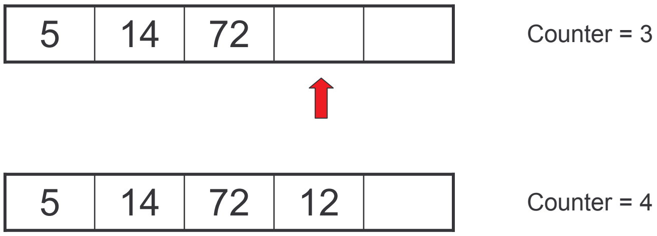
Counter = 3

Inserting an Element

- An array is a fixed size container, however, you can use a partially filled array to emulate a variable sized container
- Unsorted Array – add element at end of list (if room) and increment counter
- Sorted Array – locate where to insert element (if room), shift elements above position up one, insert element, increment counter.
  - Start shifting at last valid element (index counter-1)

Inserting an Element (2)

- Inserting an element into an unsorted array
- ArrayManipulations.unsortedInsert(12)



5	14	72		

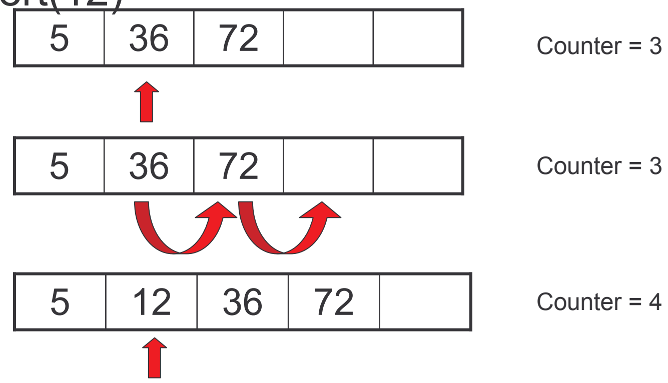
Counter = 3

5	14	72	12	

Counter = 4

Inserting an Element (3)

- Inserting an element into a sorted list
- a.insert(12)



5	36	72		

Counter = 3

5	36	72		

Counter = 3

5	12	36	72	

Counter = 4

String Split

- split(String regularExpression, int limit) is a method of the String class which can be used in place of StringTokenizer
  - StringTokenizer is being deprecated in new versions of Java
- A given String is divided up based on a given regular expression
  - Ex: “\\s” splits on white space separations
  - Info about regular expressions can be found here: http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html#sum

Sting Split Example

```
public class StringSplit {
    public static void main(String[] args) {
        String input = "This is an example.";
        String[] output=input.split("\\s");
        System.out.println("The input is: "+input);
        System.out.println("The output is: ");
        for (int i=0; i < output.length; i++)
                System.out.println(" "+output[i]);
    }
}
```

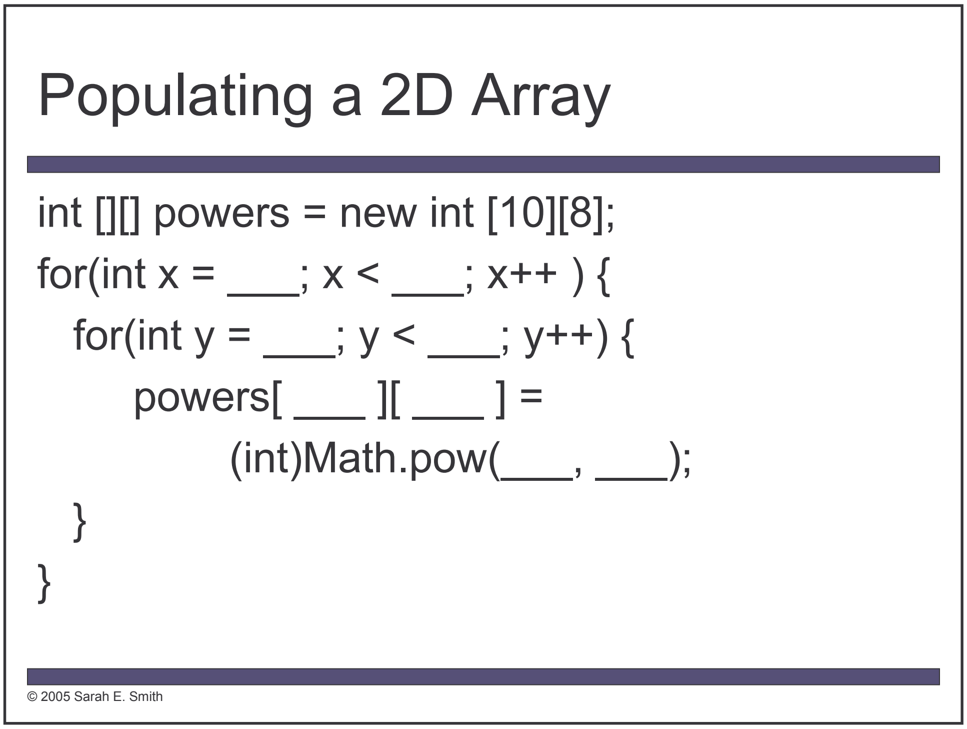
2D Arrays

- 2D arrays are useful to represent matrices and tables
- Creating a 2D array:

  `int [][] powers = new int [10][8];`
- Accessing rows:

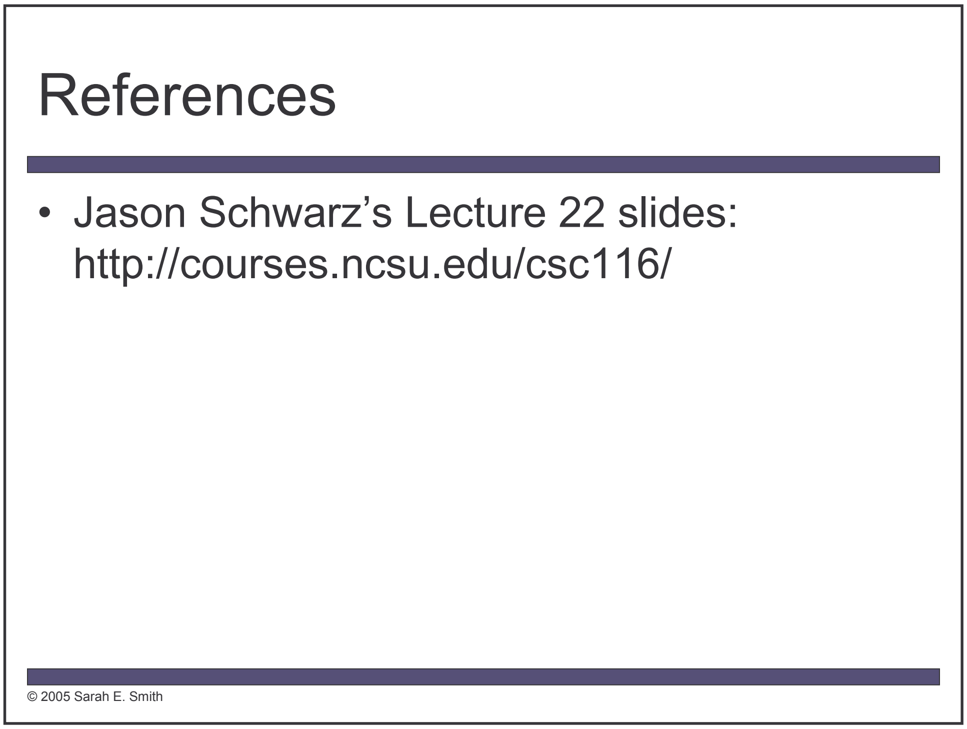
  ```
  System.out.println(“Num rows: ” +
       powers.length);
  ```
- Accessing columns of rows:

  ```
  System.out.println(“Num cols in row 0: ” +
       powers[0].length);
  ```



Populating a 2D Array

```
int [][] powers = new int [10][8];
for(int x = ___; x < ___; x++ ) {
   for(int y = ___; y < ___; y++) {
        powers[ ___ ][ ___ ] =
                (int)Math.pow(___, ___);
   }
}
```



References

- Jason Schwarz’s Lecture 22 slides: http://courses.ncsu.edu/csc116/