More Arrays

- **The Partially Filled Array**
  - Often create array large enough to handle worst case
  - That means, in practice, array is seldom fully utilized
  - Need extra variable to keep track of used portion
- (Note that ArrayLists can avoid that problem if used right)
  - Use the proper methods for adding and removing
  - Results in the `size()` method being correct

- **Partially Filled Array Example**
  - Roster of team: may not be filled (assume a `Team` class)
  ```java
  String[] roster = new String[MAX_TEAM_SIZE];
  int numMembers = 0;
  public void addPlayer(String name) {
    roster[numMembers] = name;
    numMembers++;
  }
  ```
  - What was (carelessly) assumed here?

- **Partially Filled Array Example (continued)**
  - Removing a team member can be non-trivial
  ```java
  public void removePlayer(String name) {
    int loc = -1;
    for (int k = 0; k < numMembers; k++) {
      if (roster[k].equals(name)) {
        loc = k;
        break;
      }
    }
    // What if not found: How do we tell?
    roster[loc] = roster[numMembers]; // what's this?
    numMembers--;
  }
  ```
  - What were some of the (dubious??) assumptions here?

More Arrays

- **Collection of methods used with arrays**
  - See API
- **Following very useful**
  - `static void fill(type arrayName, type value)`
    - Arrays initialized to 0 (or null) on creation
    - Allows other values or “make sure”
  - `static void sort(type arrayName)`
    - Will look at sorting later
    - State-of-the-art sorts
    - Data must be `Comparable`
  - `static List asList(type arrayName)`
    - Useful in converting arrays to ArrayLists and other lists
public class FrequencyCounter {
  final static int MAX_VALUE = 10;
  final static int SIZE = MAX_VALUE+1;
  final static String stars =
    "*******************************";
  int[] counters;
  public FrequencyCounter() {
    counters = new int[SIZE];
    Arrays.fill(counters, 0);
  }
  public void getFreq(int[] data) {
    for (int k = 0; k < data.length; k++) {
      int n = data[k];
      counters[n]++;
    }
  }
  public void histogram() {
    for (int k = counters.length - 1; k >= 0; k--) {
      int f = counters[k];
      System.out.println(k + "\t" +
      stars.substring(0, f));
    }
  }
  public static void main(String[] args) {
    int[] test = {10, 9, 10, 7, 4, 9, 10, 8, 9, 8, 10, 9};
    FrequencyCounter fc = new FrequencyCounter();
    fc.getFreq(test);
    fc.histogram();
  }
}