

CS100J 25 March 2007
Algorithms on arrays Reading: 8.3-8.5

The searching, sorting, and other algorithms will be on the course website, along with a JUnit testing class for them.

Please punctuate this:

Dear John, I want a man who knows what love is all about you are generous kind thoughtful people who are not like you admit to being useless and inferior you have ruined me for other men I yearn for you I have no feelings whatsoever when we're apart I can be forever happy will you let me be yours

Gloria

1

A neat example of the ambiguity of English! We try to use English properly and precisely, but ambiguity tends to creep in because of difference in cultures in which people grow up and simply because of differences of opinion.

I want a man who knows what love is all about. You are generous, kind, thoughtful. People who are not like you admit to being useless and inferior. You have ruined me for other men. I yearn for you. I have no feelings whatsoever when we're apart. I can be forever happy -- will you let me be yours? Gloria

I want a man who knows what love is. All about you are generous, kind, thoughtful people, who are not like you. Admit to being useless and inferior. You have ruined me. For other men, I yearn. For you, I have no feelings whatsoever. When we're apart, I can be forever happy. Will you let me be? Yours, Gloria .

2

Today and Thursday

- Look at horizontal notation for writing assertions about arrays.
- Develop several methods that process arrays. The idea is to help you learn how to develop algorithms.
 - Write a function to tell whether two arrays are equal.
 - Write a function to copy an array.
 - Write a function to tell whether two DNA sequences are complements of each other.
 - Look at other algorithms that manipulate arrays
- Look at storing a table of values in a Java array, including adding a value to the table, deleting the last value of the table, deleting some other value from the table. [Material on tables: See Sec. 8.4.](#)

3

Horizontal notation for arrays, strings, Vectors

$$b \begin{array}{|c|c|} \hline 0 & k \\ \hline \leq \text{sorted} & \geq \\ \hline \end{array} b.length$$

Example of an assertion about an array b. It asserts that:

- b[0..k-1] is sorted (i.e. its values are in ascending order)
- Everything in b[0..k-1] is \leq everything in b[k..b.length-1]

$$b \begin{array}{|c|c|} \hline 0 & k \\ \hline \leq \text{sorted} & \geq \\ \hline \end{array} b.length$$

- b[0..k] is sorted (i.e. its values are in ascending order)
- Everything in b[0..k] is \leq everything in b[k+1..b.length-1]

4

Maintain a table of values in an array

A program may have to maintain a table of values, say temperatures, within an array. The table will start out empty; then values will be added to it. We must say *where* in the array the values are stored.

```
int[] b= new int[5000]; // The n values in table are in b[0..n-1]
int n= 0; // 0 ≤ n ≤ 5000
```

$$b \begin{array}{|c|c|} \hline 0 & n \\ \hline \text{table of values} & \text{this part is unused} \\ \hline \end{array} b.length$$

// Add t to the table: // Delete last element of table
// (assuming it exists).

Deleting an element of the table doesn't require to change the array at all! It's just that a value that was in the table is now in the unused part.

5

Maintain a table of values in an array

$$b \begin{array}{|c|c|} \hline 0 & j & n \\ \hline \text{table of values} & & \text{this part is unused} \\ \hline \end{array} b.length$$

// Delete value b[j] from the table.

If the order of values in the table doesn't matter:

If order of values in table does matter:

```
n= n - 1;
// move b[j+1..n] to b[j..n-1];
```

// post: b[j+1..n] have been moved

6

Check whether two arrays are equal

```

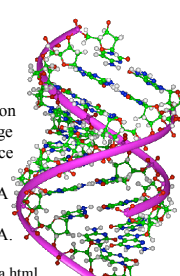
/** = "b and c are equal" (both null or both contain
    arrays whose elements are the same) */
public static boolean equals(int b, int c) {

}
    
```

7

Deoxyribonucleic acid (DNA): building block of life. DNA strand is two strings of bases twisted into a double helix. The 4 possible bases are represented by G, A, T and C. A and T bond together, as do C and G. The two sequences in a helix are complements. For example, these two sequences are complements of each other:

sequence 1: ACGTTAC
 sequence 2: TGCAATG



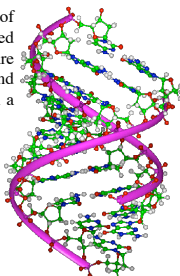
Paired bases meet at an angle. DNA is a very large molecule; image shows only a tiny fraction of typical molecule. For *Escherichia coli*, image would be 80 kilometers long. For a typical piece of DNA from an eukaryote cell, image would stretch from Dallas to Washington, D. C.! DNA is not fully stretched out inside a cell but is wound around proteins, which protect the DNA.

From www.ucmp.berkeley.edu/glossary/gloss3/dna.html

8

Deoxyribonucleic acid (DNA): building block of life. DNA strand is two strings of bases twisted into a double helix. The 4 possible bases are represented by G, A, T and C. A and T bond together, as do C and G. The two sequences in a helix are complements. For example, these two sequences are complements of each other:

sequence 1: ACGTTAC
 sequence 2: TGCAATG



```

/** = "Strings s1 and s2 are complements" */
public static boolean areComps(String s1, String s2) {

}
    
```

9

Find first position of x in array b. x is guaranteed to be in the array.
We use the horizontal notation to write assertions about arrays.

precondition: $b \left[\begin{array}{|c|c|} \hline 0 & ? \text{ (x is in here)} \\ \hline \end{array} \right] n$

postcondition: $b \left[\begin{array}{|c|c|c|} \hline 0 & i & n \\ \hline \end{array} \right] \left[\begin{array}{|c|c|} \hline \text{x not here} & \text{x} \\ \hline \end{array} \right] \text{ ? (x is in here)}$

The invariant is found easily from the postcondition

invariant: $b \left[\begin{array}{|c|c|} \hline 0 & i \\ \hline \end{array} \right] \left[\begin{array}{|c|c|} \hline \text{x not here} & \text{? (x is in here)} \\ \hline \end{array} \right] n$

1. Makes invariant true
2. When this is true, so is postcondition
3. Make progress toward termination
4. Invariant is true after repetend.

```

i = 0;
while (b[i] != x) {
    i = i + 1;
}
    
```

10

- Dutch national flag. Vague spec.:** $b[0..n-1]$ contains only red, white, blue balls. Sort it using only swaps.
- Better spec.:** Precondition: $n \geq 0$

Permute $b[0..n-1]$ to truthify:

postcondition: $b[0..h-1]$ are red balls
 $b[h..k-1]$ are white balls
 $b[k..n-1]$ are blue balls

precondition: $b \left[\begin{array}{|c|} \hline 0 & ? \\ \hline \end{array} \right] n$

postcondition: $b \left[\begin{array}{|c|c|c|} \hline 0 & h & k & n \\ \hline \end{array} \right] \left[\begin{array}{|c|c|c|} \hline \text{reds} & \text{whites} & \text{blues} \\ \hline \end{array} \right]$

invariant :

11

Partition pre: $b \left[\begin{array}{|c|c|} \hline h & k \\ \hline \end{array} \right] \left[\begin{array}{|c|} \hline \text{x} & ? \\ \hline \end{array} \right]$

algorithm:

Permute the values of the array and store a value in j to truthify:

post: $b \left[\begin{array}{|c|c|c|} \hline h & j & k \\ \hline \end{array} \right] \left[\begin{array}{|c|c|} \hline \leq x & \geq x \\ \hline \end{array} \right]$

inv: $b \left[\begin{array}{|c|c|c|} \hline h & i & j & k \\ \hline \end{array} \right] \left[\begin{array}{|c|c|c|} \hline \text{x} & \leq x & ? & \geq x \\ \hline \end{array} \right]$

12