CS100J 06 November 2007

More on arrays

Sorting: insertion- selection- quick- sort

Do exercises on pp. 311-312 to get familiar with concepts and develop skill. Practice in DrJava! Test your methods!

Haikus (5-7-5) seen on Japanese computer monitors

Yesterday it worked.
Today it is not working.
Windows is like that.

Serious error.
All shortcuts have disappeared.
Screen. Mind. Both are blank.

A crash reduces
Your expensive computer
To a simple stone.

The Web site you seek
Cannot be located, but
Countless more exist.

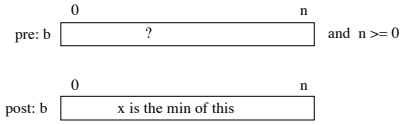
Three things are certain:
Death, taxes and lost data.
Guess which has occurred?

Chaos reigns within.
Reflect, repent, and reboot.
Order shall return.

1

Getting an invariant as picture:
Combine pre- and post-condition

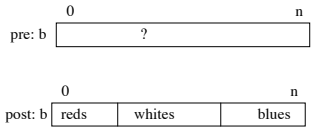
Finding the minimum of an array



2

Getting an invariant as picture:
Combine pre- and post-condition

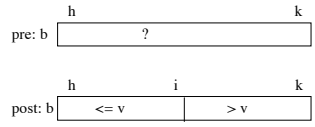
Dutch national flag. Array



3

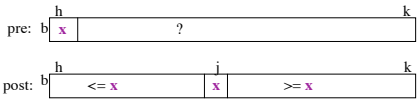
Binary search: Vague spec: Look for v in sorted array segment b[h..k].
Better spec:
Precondition: b[h..k] is sorted (in ascending order).
Store in i to truthify:
postcondition: b[h..i] <= v and v < b[i+1..k]

Below, the array is in non-descending order



4

Partition algorithm: **x is called the pivot value**



change: b 3 5 4 1 6 2 3 8 1 (h, k)

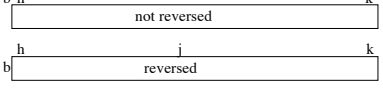
into b 1 2 1 3 5 4 6 3 8 (h, j, k)

or b 1 2 3 1 3 4 5 6 8 (h, j, k)

(x is not a program variable; it just denotes the value initially in b[h].)

5

Reversing array segment b[h..k]



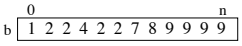
change: b 1 2 3 4 5 6 7 8 9 9 9 9 (h, k)

into b 9 9 9 9 8 7 6 5 4 3 2 1 (h, j, k)

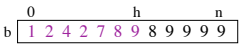
6

Remove adjacent duplicates

change:



into

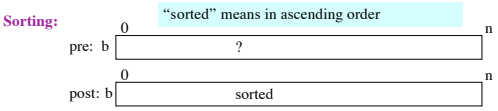


don't care what is in b[k+1..n]

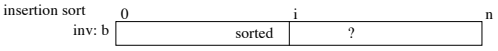
postcondition:

b[0..h] = initial values in b[0..n] but with adj dups removed

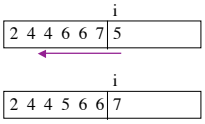
Sorting: "sorted" means in ascending order



insertion sort invariant:

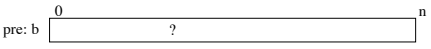


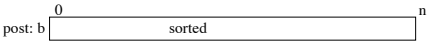

```
for (int i= 0;  i < n;  i= i+1) {
    Push b[i] down into its sorted
        position in b[0..i];
}
```



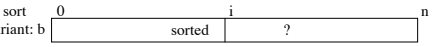
Iteration i makes up to i swaps. In worst case, number of swaps needed is 1 + 2 + 3 + … (n-1) = (n-1)*n / 2.

Called an "n-squared", or n^2, algorithm.



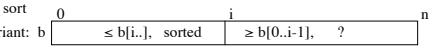


insertion sort invariant:

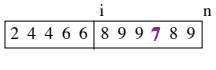


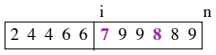
Add property to invariant: first segment contains smaller values.

selection sort invariant:




```
for (int i= 0;  i < n;  i= i+1) {
    int j= index of min of b[i..n-1];
    Swap b[j] and b[i];
}
```





Also an "n-squared", or n^2, algorithm.

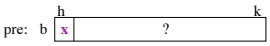
Quicksort

```
/** Sort b[h..k] */
public static void qsort(int[] b, int h, int k) {
    if (b[h..k] has fewer than 2 elements)
        return;
    int j= partition(b, h, k);
    // b[h..j–1] <= x < b[j+1..k]
    // Sort b[h..j–1]  and  b[j+1..k]
    qsort(b, h, j–1);
    qsort(b, j+1, k);
}
```

To sort array of size n. e.g. 2^{15}

Worst case: n^2 e.g. 2^{30}

Average case: n log n. e.g. $15 * 2^{15}$ = 2^{15} = 32768



j= partition(b, h, k);

Sir Tony Hoare, Quicksort author

Fritz Bauer, My advisor

Pioneer in computing in the 50's, 60's. Developer of the historical computing section of the Deutsches Museum

Marienplatz, Munich

Keypunch machine. In the 1960s and early 1970s, a program was punched on such cards. A card deck containing a program to be run would be taken out to the computer, near the Ithaca airport, and run. Get output back 3-4 hours after handing in the card deck.

Zuse built the first working computer in the late '30s, the Z3, completed in 1941. The original machine no longer exists.

Konrad Zuse's Z4