Math 135 - Algorithms (Ch. 3.1)

Note Title 2/24/2010

Announcements

- HW due next Monday at beginning of class

- My office hours tomorrow are at the usual time (1-2 pm)

Algorithm

A set of instructions for solving a problem

(NOT necessarily a program!)

Examples:

- recipe
- program
- tying a shoe

We often use pseudocode to write down computer algorithms.

Common programming concepts:

- if statements
- loops
- variables
- functions or procedures
- input/output

Ex: Pseudocode to find the maximum element in a sequence $a_1 .. a_n$

```
FINDMAX($a_1, a_2, \dots, a_n$):

    max := $a_1$
    for i := 2 to n
        if max < $a_i$
            max := $a_i$
    return max
```

for loop
if statement

Why := ?

if max = a_i
(==)

Searching

Suppose I give you a list of numbers $a_1, \ldots, a_n$ and ask if $x \in \{a_1, \ldots, a_n\}$.

How would you check?

Go through & check if each element is equal to x.

If we reach the end of the list, then x is not in the list.

```
LINEARSEARCH($x, a_1, \ldots, a_n$):
    $i := 1$
    while ($i \le n$ and $x \ne a_i$)
        $i := i + 1$
    if $i \le n$
        location $:= i$
    else
        location $:= 0$
    return location
```

as long as $i \le n$ and $x \ne a_i$, this loop ~~keeps~~ happening

Another Search Strategy:

Ex: Take out your book & open it to page 171.

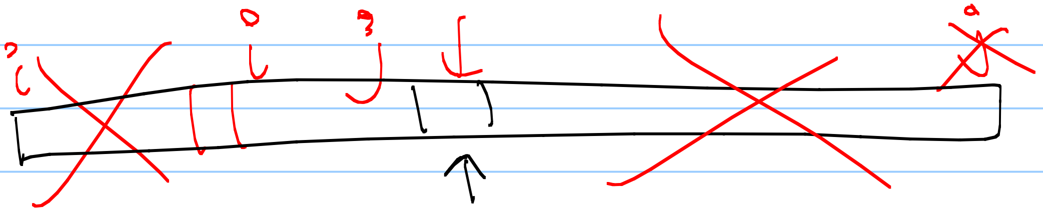
How does your algorithm to do this differ from the linear search algorithm?

Open book & check middle (not turn to page 1).

If page was too big, go left, otherwise go right.

When searching in a sorted list, we can do a ~~faster~~ search called **binary search**.

- Compare to middle element of list.



$\frac{i+j}{2}$

- If that element is bigger than x, search in left half.
- If that element is smaller than x, search in right half.

(skipping pseudocode for now – see book for details)

Sorting:

Fundamental CS problem:

Given a list of n things, put them in order

How?

Many ways!

Bubble Sort:

Compare adjacent elements + switch them if in wrong order

$\underbrace{3 \quad 2}$ 4 1 5

(1st pass) $\underbrace{2 \quad 3}$ 1 4 5

(2nd pass) 2 1 3, $\underbrace{4 \quad 5}$

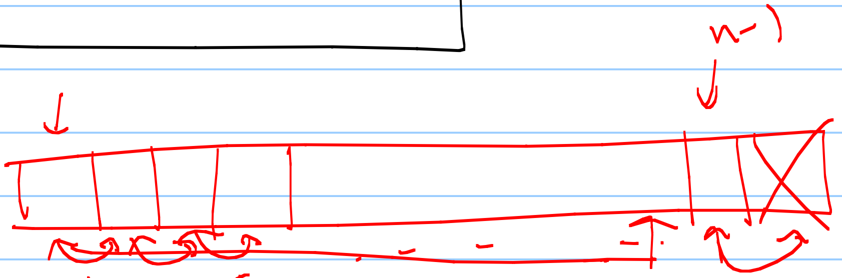
3 5 2 4 1

3 2 4 1 5

(At end of 1st pass, max. element is in correct spot)

Pseudocode

```
BUBBLE SORT $(a_1 \ldots a_n)$:
    for $i := 1$ to $n-1$
        for $j := 1$ to $n-i$
            if $a_j > a_{j+1}$
                swap $a_j$ and $a_{j+1}$
```



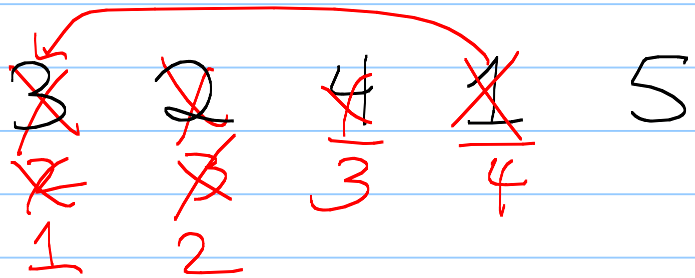
~~$i=1$~~

~~$i=2$~~ $i=3$

$j = 1$ up to ~~$n-1$~~ ~~$n-2$~~ $n-3$

Insertion Sort

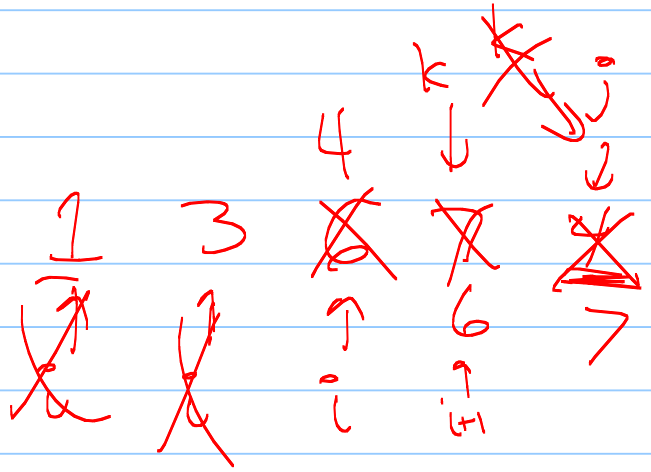
- If first i items are sorted, take $(i+1)$st and put it in correct spot.



Pseudo code

```
INSERTION SORT ($a_1 .. a_n$):
    for $j := 2$ to $n$
        $i := 1$
        while ($a_j > a_i$)
            $i := i+1$
        temp $:= a_j$
        for $k := j$ to $i+1$
            $a_k := a_{k-1}$
        $a_i :=$ temp
```

opies everything bigger than a_j down 1 spot (to make room for a_j)



temp ← 4