Announcements

Quiz 2 Wed. in class

HW3 will be posted soon, due Sun @ 11:59 pm

Strong suggestion: read the textbook!

Algorithms (cont.)

Recall:

Def: An algorithm is a finite sequence of precise steps

Properties:

- Input
- Output
- Definiteness: Steps are precisely-defined
- Correctness: Always gives the right answer
- Finiteness: Finite # steps for any input
- Effectiveness: You can actually do each step
- Generality: Works for all possible inputs

Ex: Finding max. elt. in a finite sequence

procedure max($a_1, \ldots, a_n$: integers)

$m := a_1$

for $i := 2$ to n

    if $m < a_i$ then $m := a_i$ — set m equal to a_i

return m

Check properties:

- Input ✓
- Output ✓
- Definiteness: Yes, because we're only incrementing, assigning value, and checking conditions
- Correctness: Yes, always returns max. elt.
- Finiteness: Yes, goes through the list once and terminates
- Effectiveness: Yes, because we're only incrementing, assigning value, and checking conditions
- Generality: Yes, works for all finite lists of integers

Searching algorithms:

General problem: locate an elt. x in a list of distinct elts. $a_1, \ldots, a_n$, or determine it's not in the list

Linear search algorithm

(Use when list is unordered)

Input: integer x

list of integers: $a_1, \ldots, a_n$

Output: Location of x in list (or 0 if not found)

Algorithm:

Set $i := 1$

while ($i \leq n$ and $x \neq a_i$)

$i := i+1$

$$\text{location} := \begin{cases} i, & \text{if } i \leq n \\ 0, & \text{otherwise} \end{cases}$$

return location

Binary search algorithm
(Use when list is ordered)

Input: integer x
list of integers: $a_1, \ldots, a_n$ w/ $a_1 < a_2 < \cdots < a_n$

Output: Location of x in list (or 0 if not found)

Algorithm:

Let $i := 1$ (endpoints of search interval
Let $j := n$

While $i < j$

Let $m := \lfloor \frac{i+j}{2} \rfloor$

if $x > a_m$

$i := m+1$

else

$j := m$

$$\text{location} := \begin{cases} i, & \text{if } x = a_i \\ 0, & \text{otherwise} \end{cases}$$

return location

Sorting algorithms:

General problem: Sort a list in increasing order

Many different algorithms

Bubble sort algorithm:

Input: list of integers $a_1, \ldots, a_n$

Output: list of integers which is the original list in inc. order

Algorithm:

```
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}
return a_1, ..., a_n
```

Class activity: perform bubble sort on the list

3, 2, 4, 1, 5

Insertion sort algorithm (if time):

Input: list of integers $a_1, \ldots, a_n$

Output: list of integers which is the original list in inc. order

Algorithm:

```
for j := 2 to n
    Let i := 1
    While a_j > a_i                (finding the spot for a_j)
        i := i+1
    Let m := a_j                   (insert a_j into spot m)
    for k := 0 to j-i-1
        a_{j-k} := a_{j-k-1}       (shift other elts. to make room)
    a_i := m
return a_1, ..., a_n
```

Class activity: perform insertion sort on the list

3, 2, 4, 1, 5