Math/CSE 1019:
Discrete Mathematics for Computer Science

Fall 2011

Suprakash Datta

datta@cse.yorku.ca

Office: CSEB 3043

Phone: 416-736-2100 ext 77875

Course page: http://www.cs.yorku.ca/course/1019

Problem 3: Detecting palindromes

- “reads the same forwards and backwards” e.g. pop, noon
- Using an auxiliary array
- Without using an auxiliary array

A Harder Problem

INPUT: A[1..n] - an array of integers, k, 1 ≤k ≤length(A)
OUTPUT: an element m of A such that m is the kth largest element in A.

Think for a minute

Brute Force:

Find the maximum, remove it.
Repeat k-1 times.
Find maximum.

Q: How good is this algorithm?

Q: Is there a better algorithm? For some k, YES!

Sorting and Searching

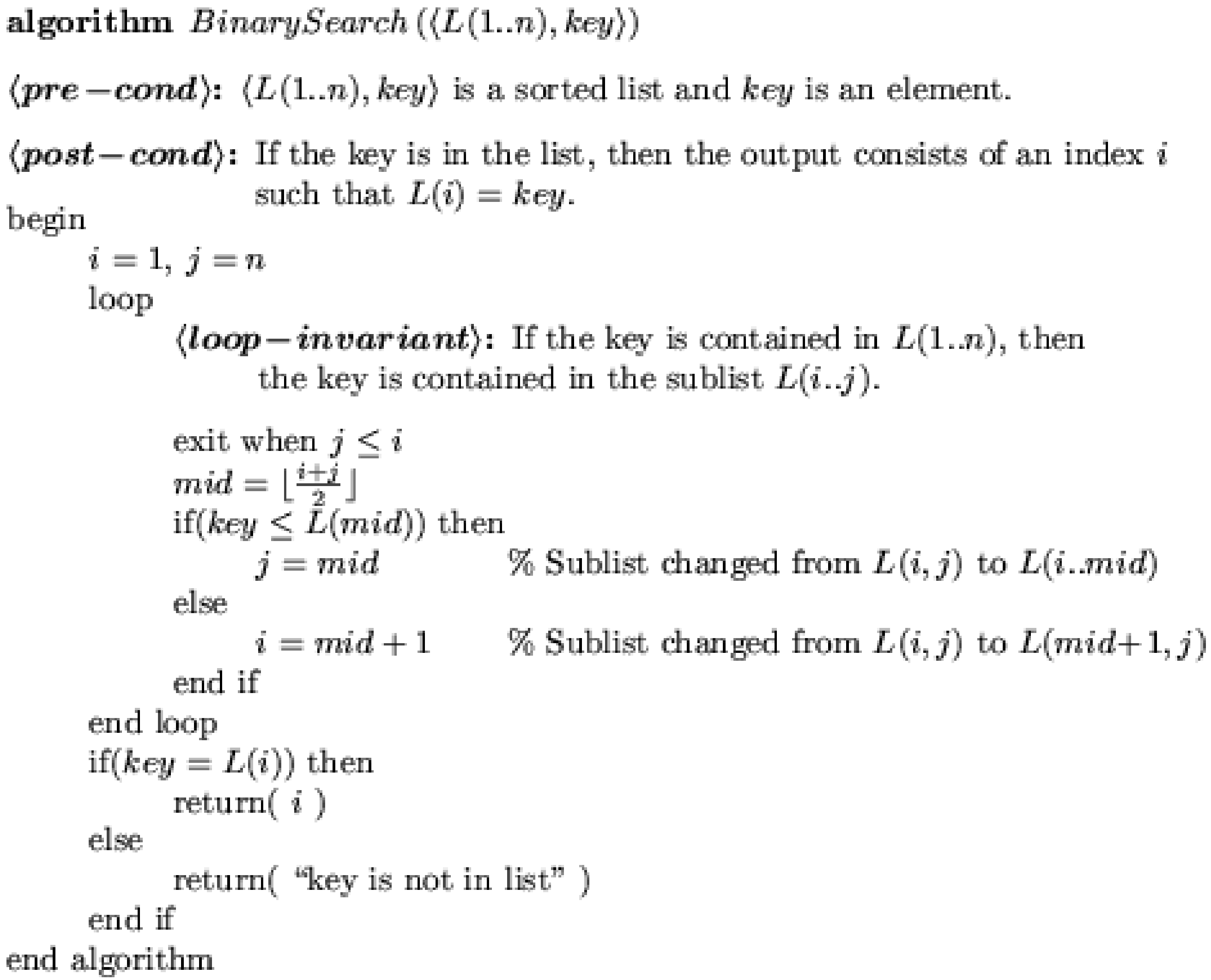
- Very basic operations
- Used very,very often in real applications
- LOTS of new ideas

Searching an array

- Given an array A[1..m] does there exist a number (key) n?
- Unsorted array: linear search
- Sorted array: Can you do better?
- YES!

Binary search: Use the sorted property to eliminate large parts of the array.

Pseudocode for binary search



```
algorithm $BinarySearch(\langle L(1..n), key \rangle)$

$\langle pre-cond \rangle$: $\langle L(1..n), key \rangle$ is a sorted list and $key$ is an element.

$\langle post-cond \rangle$: If the key is in the list, then the output consists of an index $i$
                  such that $L(i) = key$.
begin
    $i = 1$, $j = n$
    loop
        $\langle loop-invariant \rangle$: If the key is contained in $L(1..n)$, then
            the key is contained in the sublist $L(i..j)$.

        exit when $j \leq i$
        $mid = \lfloor \frac{i+j}{2} \rfloor$
        if($key \leq L(mid)$) then
            $j = mid$            % Sublist changed from $L(i,j)$ to $L(i..mid)$
        else
            $i = mid + 1$        % Sublist changed from $L(i,j)$ to $L(mid+1, j)$
        end if
    end loop
    if($key = L(i)$) then
        return( $i$ )
    else
        return( “key is not in list” )
    end if
end algorithm
```

Data structures

- By preprocessing (sorting) the data into a data structure (sorted array), we were able to speed up search queries.
  Very common idea in Computer Science

- Many other data structures are commonly used: linked lists, trees, hash tables,….

- CSE 2011: Data Structures
- CSE 4101: Advanced Data Structures

Sorting

- Simple algorithm using FindMax

```
j=n
while (j>1){
    maxindex = index of max A[1..j]
    swap (A[maxindex], A[j])
    j=j-1
}
```

- Is this the fastest possible sort?