

Programming 2

Searching an element in an array

Francisco Iacobelli

Department of Computer Science
Northeastern Illinois University

September 15, 2011

- Fast Retrieval of Information.
- Think databases, names, events, etc.

Sequential Search

SequentialSearch (list, target)

```
for each element of the list
  if the element is equal to the target
    element is found and return its position
if reached the end of list
  return "Not Found"
```

Example: Search "4" in {5, 3, 1, 6, 8, 4, 7, 2}

SequentialSearch

Pseudocode

```
procedure SequentialSearch( A : list, e: element)
  for i=0 to length(A) - 1
    if A[i] = e
      return i
  return "Not Found"
end procedure
```

Requires sorted list

```
procedure BinarySearch(A: list, e: element)
  divide the list in half
  if e is equal to the midpoint of the list
    return it
  else
    if e is greater than the midpoint do
      BinarySearch(upper half of A, e)
    else
      BinarySearch(lower half of A, e)
```

Example: search 4 and 8 in {5, 3, 1, 6, 8, 4, 7, 2}

Binary Search

Recursive

```
BinarySearch(A[0..N-1], value, low, high)
    if (high < low)
        return -1 // not found
    mid = low + (high - low) / 2
    if (A[mid] > value)
        return BinarySearch(A, value, low, mid-1)
    else if (A[mid] < value)
        return BinarySearch(A, value, mid+1, high)
    else
        return mid // found
```

Binary Search

Traditional

```
min := 0;
max := N-1;
repeat
  mid := (min+max) div 2;
  if x > A[mid] then
    min := mid + 1;
  else
    max := mid - 1;
until (A[mid] = x) or (min > max);
if
```

Example: Sort {5, 3, 1, 6, 8, 4, 7, 2}