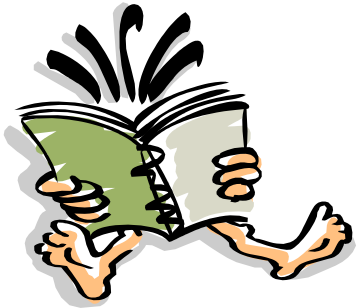
# Quick Sort & Selection Sort



# Introduction to Quick Sort

- Divide
  - Partition the array A[p.. r] into two sub arrays A[p..q-1] and A[q+1..r]
  - All element in A[p..q-1] is less than or equal to A[p]
  - All element in A[q+1,r] is greater than A[p]
- Conquer
  - Sort the sub array A[p..q-1] & A[q+1] by recursive call to Quick Sort.
- Combine
  - Since the sub array are sorted no work is needed to combine them.

# Quick Sort(Continue..)

QuickSort(A,p,r)

if p<r

then q is partition(A,p,r)

QuickSort(A,p,q-1)

QuickSort(A,q+1,r)

- To sort entire array A, the initial call is QuicSort(A,1,length[A])

# Partitioning the array

```
Partition(A,p,r)
x := A[r]
i:=p-1
for j:=p to r-1
{
    if A[j]<=x then
          i:=i+1
          exchange A[i]  with A[j]
}
exchange  A[i+1] with A[j]
return i+1
```

# Analysis of Quick sort

- *Best case*: split in the middle

  $\Theta(n \log n)$

- *Worst case*: sorted array!

  $\Theta(n^2)$

- *Average case*: random arrays

  $\Theta(n \log n)$

# Selection Sort

- Partition algorithm can also be used  to obtain an efficient solution for the selection problem
- Selection sort algorithm is used to find k smallest element.

# Selection Sort(Continue ..)

```
Select(A, r, k)
{
repeat
   {
     q:=partition(A,p,r)
     if(k =q) then return
              else if(k<j) then r:=q-1
                       else p:=q+1
   }until(false)
}
```

# Application

- Cache

# Scope of Research

- Life-critical (medical monitoring, life support in aircraft and space craft)
- Mission-critical (monitoring and control in industrial and research plants handling dangerous materials, control for aircraft, defense, etc)

# Assignment

- Q1)What is quick sort?
- Q.2)Explain  behavior of Quick sort when list is in sorted order.
- Q.3) Find the $6^{th}$ smallest element of given list using  partition Selection sort. List element are

  22,88,77,11,33,55,66,44