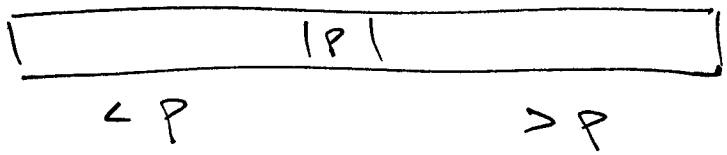
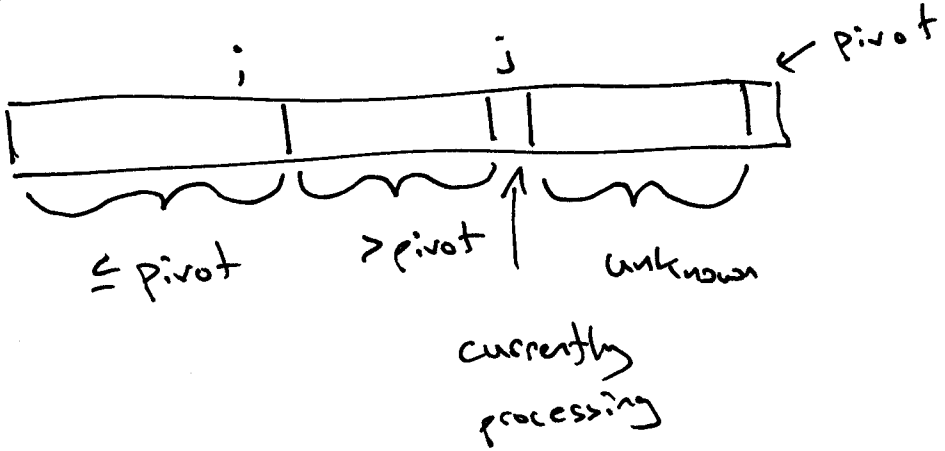
Quick Sort



Partition



```
PARTITION(A, p, r)
1  x = A[r]
2  i = p − 1
3  for j = p to r − 1
4        if A[j] ≤ x
5              i = i + 1
6              exchange A[i] and A[j]
7  exchange A[i + 1] and A[r]
8  return i + 1
```

Input size? $r - p = n$

Basic op? comparison between $A[j]$ and x

Complexity? $\Theta(n)$

```
QUICKSORT(A, p, r)
1  if p < r
2        q = PARTITION(A, p, r)
3        QUICKSORT(A, p, q − 1)
4        QUICKSORT(A, q + 1, r)
```

Input size? $r - p = n$

Basic op? comparison from inside partition

Best/Worst/Average?

Worst case? sorted or reverse sorted

the pivot is always the smallest or largest element

Best case? the pivot is the median element at every step

Worst case:

$$T(n) = T(n-1) + n$$

(number of comparisons done by partition)

$$T(0) = 0$$

$$T(n) = T(n-2) + (n-1) + n$$

$$= T(n-3) + (n-2) + (n-1) + n$$

$$= T(n-i) + (n-i+1) + \dots + n$$

$$= T(0) + 1 + 2 + \dots + n$$

$$= \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \in \Theta(n^2)$$

Best Case:

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

$$T(1) = 0$$

$$T(n) = 4T\left(\frac{n}{4}\right) + 2n$$

$$= 8T\left(\frac{n}{8}\right) + 3n$$

$$= 2^i T\left(\frac{n}{2^i}\right) + in$$

$$= 2^{\lg n} T(1) + n \lg n$$

$$= n \lg n \in \Theta(n \log n)$$

Average case

- Let $y_1, y_2, y_3, \ldots, y_n$ be the same values as the input in sorted order
- Let X_{ij}, where $i < j$, be an indicator random variable where

  $X_{ij} = 1$ iff y_i and y_j are compared at any point during Quicksort's running time and 0 otherwise
- Let X be the total number of comparisons

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

- The $E[X]$ is the average number of comparisons

$$E[X] = E\left[\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} X_{ij}\right]$$

linearity of expectation

$$= \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} E[X_{ij}]$$

- Since X_{ij} is an indicator random variable

$$E[X_{ij}] = pr(X_{ij} = 1)$$

- Elements y_i and y_j are compared iff y_i or y_j is chosen as the pivot before any element in $\{y_{i+1}, \ldots y_{j-1}\}$ is chosen
- Assume the pivot is chosen uniformly at random
- $pr(X_{ij} = 1) = \frac{2}{j - i + 1}$

$$
\begin{aligned}
E[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} \\
&= \sum_{k=2}^{n} \sum_{i=1}^{n+1-k} \frac{2}{k} \\
&= \sum_{k=2}^{n} \left[\frac{2}{k} \sum_{i=1}^{n+1-k} 1 \right] \\
&= \sum_{k=2}^{n} \left[\frac{2}{k} (n+1-k) \right] \\
&= \sum_{k=2}^{n} \left[\frac{2}{k} (n+1) \right] - \sum_{k=2}^{n} \frac{2}{k} k \\
&= \sum_{k=2}^{n} \left[\frac{2}{k} (n+1) \right] - 2(n-1)
\end{aligned}
$$

$$= \sum_{k=2}^{n} \left[\frac{2}{k} (n+1) \right] - 2(n-1)$$

$$= 2(n+1) \sum_{k=2}^{n} \frac{1}{k} - 2(n-1)$$

$$= 2(n+1) \sum_{k=1}^{n} \frac{1}{k} - 4n$$

$$\approx 2(n+1) \ln n - 4n$$

$$\in \Theta(n \log n)$$

$$Pr(X_{ij} = 1) = \frac{2}{j - i + 1}$$

$$Pr(X_{ij} = 0) = \frac{j - i + 1 - 2}{j - i + 1}$$

Pivot Selection

options

- last element
- median of first, middle, last
- random element
- find the median

Binary Insertion Sort

In the INSERTIONSORT implementation we showed in class we compared the element we were placing with each element in the sorted subarray until we found the correct position. Since the subarray is sorted we can use binary search to find the correct final position. (Reminder: BINARYSEARCH performs $\Theta(\log n)$ comparison in the worst case)

BINARYINSERTIONSORT(A)

```
1  for i = 1 to A.length − 1
2      ins = BINARYSEARCH(A, 0, i, A[i])
3      tmp = A[i]
4      for j = i − 1 downto ins
5          A[j + 1] = A[j]
6      A[ins] = tmp
```

(a) What is the total number of comparisons performed in the worst case?

$\sum_{i=1}^{n} \log(i) \in \Theta(n \log n)$

(b) Will this change actually improve the worst case running time?

(a) What is the total number of comparisons performed in the worst case?

Solution: For each step in the loop the binary search will perform approximately $\lg n$ comparisons in the worst case, where n is the size of the sorted subarray. This leads to the following summation for the number of comparisons.

$$\sum_{i=1}^{n-1} \lg(i)$$

This is somewhat unpleasant to solve exactly, but we can easily give an upper bound of $O(n \log n)$. Here's just one of many ways to show this:

$$\begin{aligned}\sum_{i=1}^{n-1} \lg(i) &= \lg(1) + \lg(2) + \lg(3) + ... + \lg(n-1) \\ &\leq \lg(n) + \lg(n) + \lg(n) + ... + \lg(n) \\ &\leq n \lg(n) \in O(n \log n)\end{aligned}$$

It also turns out that this has a lower bound of $\Theta(n \log n)$ as well, but this is a bit more complicated to show. The general idea here is to throw away the first half of the sum on

line 3.

$$\sum_{i=1}^{n-1} \lg(i) = \lg(1) + \lg(2) + \lg(3) + ... + \lg(n-1) \tag{1}$$

$$\geq \lg(\frac{n}{2}) + \lg(\frac{n}{2}+1) + \lg(\frac{n}{2}+2) + ... + \lg(n-1) \tag{2}$$

$$\geq \lg(\frac{n}{2}) + \lg(\frac{n}{2}) + \lg(\frac{n}{2}) + ... + \lg(\frac{n}{2}) \tag{3}$$

$$\geq \frac{n}{2}\lg(\frac{n}{2}) \tag{4}$$

I'll leave things off here, but really a complete proof would add a few more steps to remove the $\frac{n}{2}$ from inside the lg.

(b) Will this change actually improve the worst case running time?

Solution: Unfortunately no, this will not improve the worst case running time. It does reduce the number of comparisons, but since the comparison is no longer inside the innermost loop it can no longer be considered the basic operation. No matter how many comparisons occur we still need to swap elements to the right in order to make room for the item we're inserting.