Student Activity

```
Mystery(int array a[]) {
  for (int p = 1; p < length; p++) {
     int tmp = a[p];
     for (int j = p; j > 0 && tmp < a[j-1]; j--)
          a[j] = a[j-1];
     a[j] = tmp;
  }
}
```

What sort is this? Insertion

What is its running time?
Best?
Avg?
Worst?

Merge Sort: Complexity

Base case: T(1) = c
T(*n*) = 2 T(*n*/2) + *n*
…
T(*n*) = O(*n* log *n*)
(best, worst)

We Want:
$n/2^k = 1$
$n = 2^k$
log n = k

Base case: T(1) = c

T(*n*) = 2 T(*n*/2) + *n*
= 2 (2T(n/4) +n/2) + n
=4T(n/4)+n+n
=4T(n/4)+2n
=4 (2T(n/8) +n/4) + 2n
=8T(n/8)+n+2n
=8T(n/8)+3n
=2^k T(n/2^k) + kn

= nT(1) +n logn
= n+ n log n

QuickSort: Best case complexity

T(*n*) = 2T(*n*/2) + *n*
…
T(*n*) = O(*n* log *n*)

Same as Mergesort

What is best case? Always chooses a pivot that splits array in half at each step

QuickSort: Worst case complexity

T(*n*) = *n* + T(*n*-1)
…
T(*n*) = O(n^2)

T(1) = c
T(*n*) = *n* + T(*n*-1)

T(n) = n + (n-1) + T(n-2)
T(n) = n + (n-1) + (n-2)+ T(n-3)
T(n) = 1 + 2+ 3+…+N
…
T(*n*) = O(n^2)

Always chooses WORST pivot – so that *one array is empty* at each step