Mergesort

Given: array \(a[1:n]\), order elements of array.

\(a[1:n] = \) ________

Split into two pieces

\(a[1:n/2] = \) ________, \(a[n/2+1:n] = \) ________

Split into two pieces

Sort them individually, then merge the results.

\(a[1:n] = \) ________

Done

What's the algorithm?
def mergesort(a[1:n]):
    if n == 1: return
    mergesort(a[1:n/2])
    mergesort(a[n/2+1:n])
    merge(a[1:n/2], a[n/2+1:n])
    return

def merge(x[1:k], y[1:l]):
    make z[1:k+l]
    merge-aux(x, y, z)
    copy z into x y

def merge-aux(x[1:k], y[1:l], z[1: l+k]):
    if k == 0: copy y into z, return
    if l == 0: copy x into z, return
    if x[1] <= y[1]:
        z[1] = x[1]
        merge-aux(x[2:k], y[1:l], z[2: l+k])
    else:
        z[1] = y[1]
        merge-aux(x[1:k], y[2:l], z[2: l+k])
Merge Sort

Runtime?

Recurrence Relation

\[ T(n) = 2T(n/2) + O(n) \]

Master Theorem

\[ a = 2, \ b = 2, \ d = 1 \]

\[ \frac{a}{b^d} = \frac{2}{2^1} = 1 \quad \Rightarrow \quad O(n^d \log n) = O(n \log n) \]