

LATE ASSIGNMENT 20% off each day.

Wed morning: Submit to Lora at the Department office.

After that: Put in the box outside the Department office.

Remember to email the instructor about your submission.

Heapsort

1. Build the heap (BuildHeap – time $\mathcal{O}(n)$)
2. Move the largest element to the last position
3. Reduce heapsize (s) by 1.
4. Make the first s elements into a heap:
by one call to $Heapify(A, 1, s)$
5. Go back to 2.
 - Running time $\mathcal{O}(n \log n)$
 - An in-space sorting algorithm.

Mergesort

Divide-and-Conquer technique (will learn more today)

Mergesort(A, p, r)

If $p < r$:

1. **Divide (into 2 halves):** $q = \lfloor \frac{p+r}{2} \rfloor$
2. **Conquer:** Mergesort(A, p, q)
3. **Conquer:** Mergesort($A, q+1, r$)
4. **Combine:** Merge(A, p, q, r)

Merge(A,p,q,r) Merge (sorted) $A[p \dots q]$, $A[(q + 1) \dots r]$

- Copy $A[p \dots q]$ to $B[1 \dots (q - p + 1)]$
- Merge
 1. $k \leftarrow p$; $i \leftarrow q + 1$; $j \leftarrow 1$
 2. while $j \leq (q - p + 1)$ do
 3. if $i > r$ then
 4. $A[k] \leftarrow B[j]$; $k \leftarrow k + 1$; $j \leftarrow j + 1$
 5. else do
 6. if $A[i] < B[j]$ then do
 7. $A[k] \leftarrow A[i]$; $i \leftarrow i + 1$
 8. else do
 9. $A[k] \leftarrow B[j]$; $j \leftarrow j + 1$
 10. end if
 11. $k \leftarrow k + 1$

12. end if

13. end while

(NOT an in-place sorting algorithm.)

The Master Theorem

If for some $a, b, d > 0$:

$$T(n) = aT(n/b) + \mathcal{O}(n^d)$$

then

$$T(n) = \begin{cases} \mathcal{O}(n^d) & \text{if } a < b^d \\ \mathcal{O}(n^d \log(n)) & \text{if } a = b^d \\ \mathcal{O}(n^{\log_b(a)}) & \text{if } a > b^d \end{cases}$$

We argued for the case $n = b^m$, using Assignment 1.
Other cases can be reduced to this case.