private static void sort(Comparable[] a, int lo, int hi) {
    if (hi <= lo) return; // constant
    int mid = lo + (hi - lo) / 2;
    sort(a, lo, mid); // (n/2)
    sort(a, mid+1, hi); // (n/2)
    merge(a, lo, mid, hi); // n
}

public static void merge(Comparable[] a, int lo, int mid, int hi) {
    int i = lo, j = mid+1;
    for (int k = lo; k <= hi; k++) // a*n+b linear time
        aux[k] = a[k];
    for (int k = lo; k <= hi; k++) // linear time as well
        if (i == mid) a[k] = aux[j++];
    else if (j == hi) a[k] = aux[i++];
    else if (less(aux[j], aux[i])) a[k] = aux[j++];
    else a[k] = aux[i++];
}

```
merge time is \( \alpha n^3 \)
```
#### Mergesort

\[ T(1) = c \]

\[ T(n) = 2 T \left( \frac{n}{2} \right) + an + b \]

\[ \Theta(n) \]

1. Find recurrence relation and base case
2. Solve
3. If asked: prove by induction

---

Solve recurrence by substitution:

...
Assume $n = 2^k$

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

$$2 \cdot T \left( \frac{n}{2} \right) = 2 \cdot T \left( \frac{n}{4} \right) + 2^{k-1}$$

$$T \left( \frac{n}{4} \right) = 2 \cdot T \left( \frac{n}{16} \right) + 2^{k-2}$$

$$\ldots$$

$$2^{k-1} \cdot T(1) = 1 \cdot 2^k$$

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

$$T(1) = 2$$

$$T(n) = (k+1) \cdot 2^k$$

$$T(n) = (\log n + 1) \cdot n$$

$\Rightarrow T(n)$ is $\Theta(n \log n)$
Proof by induction:

1. Big-Oh: \( \exists C, n_0, T(n) \leq C \log n \cdot n \) for \( n_0 \geq n_0 \)

(we do not fix \( C \) yet)
but we have to fix \( n_0 \) for the base case?

**BASE**

pick \( n_0 = 2 \)

\( n = 2 \quad T(2) = 2 \cdot T(4) + 2 = 4 \)

\( 2 \cdot \log 2 = 2 \)

\( 4 \leq C \cdot 2 \)

already constraint on \( C \)

**IND-HYP**

Assume that the claim

\[ T(j) \leq C \cdot j \cdot \log j \text{ for } 2 \leq j \leq k \]

strong induction

**IND-STEP**

remember we have
to show that \( T(k+1) \leq C \cdot (k+1) \cdot \log (k+1) \)
\[ T(k+1) = 2T\left(\frac{k+1}{2}\right) + k+1 \]

from \[ HYP \quad T\left(\frac{k+1}{2}\right) \leq C \frac{k+1}{2} \log \left(\frac{k+1}{2}\right) \]

This is why we needed the **STRONG induction**.

\[ 2T\left(\frac{k+1}{2}\right) + k+1 \leq \left( C \frac{k+1}{2} \log \frac{k+1}{2} \right) + k+1 \]

= \[ c(k+1)(\log(k+1) - \log 2) + k+1 \]

= \[ c(k+1)\log(k+1) - c(k+1) + (k+1) \]

= \[ c(k+1)\log(k+1) - (c-1)(k+1) \]

**Base case:** \[ c \geq 2 \]

\[ \leq c(k+1)\log(k+1) \]

For \( c=2 \)

Observe that we were not able to fix \( c \) until the end?
DIVIDE AND CONQUER

closest pair of two points in the plane

1. closest pair of numbers on a line (1D)
   - compute all differences \( \frac{n(n-1)}{2} \)
   - points are already sorted \( D \)
   - just look at right neighbor.
   - incl. sorting \( \Theta(n \log n) \) for sort \( \Theta(n) \) for closest pair.

2. 2D sorting
   - Simple: compute all \( \frac{n(n-1)}{2} \) distances and find min.
which $O(n^2)$

$\rightarrow$ idea:
$sort \uparrow \downarrow x$

close is $x$ but not close in
distance $d = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$

still use $x$-coordinate
to **divide**: median of $x$-coordinate be $l$ and divide my points with a vertical line at $x=l$
Base case \( n=2 \)

If \( n>2 \) then divide into two halves:

\[
\delta_{\text{left}} = \text{find closest (left half)} \\
\delta_{\text{right}} = \text{find closest (right half)} \\
\delta = \min(\delta_{\text{left}}, \delta_{\text{right}})
\]

combine

The combine function will check if any distance crossing the vertical line is smaller than \( \delta \).
how can we find an efficient algorithm to check if any of the green lines is less than the shortest green line?

Take all points with \( 0.5 \leq x \leq 0.5 \) and sort them in y-coordinate, call this array \( S_y \).

Find min \( y \) in \( i = \frac{1}{2}, \ldots, \frac{1}{2} \) \( \Theta(1/S_y^2) \).
Lemma: For any pair \((s_i, s_j)\) if \(|\|s_i - s_j\|| \leq 5\) then \(|i - j| \leq 7\).

By checking only this box, I have to check only 8 points. (\(|i - j| \leq 7\)).

Why is this important?

find closest (left half) \(T(\frac{n}{2})\)
find closest (right half) \(T(\frac{n}{2})\)
combine \(7n\) (sort by \(x\) then \(y\) is advance)

\[ T(n) = 2 \cdot T\left(\frac{n}{2}\right) + 5 \rightarrow \Theta(n \log n) \]
What is the solution checking?

\[ \sqrt{\frac{5}{2}} < \delta \]