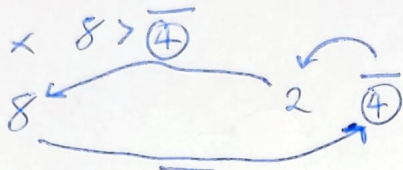6) 3, 1, 8, 5, 2, ④, 9, 5, 7

a)
? (Check 3rd element)

× 8 > ④

8 ← 2 ④

b)
3, 1, 2, 5, ④, 8, 9, 5, 7

7) a)
? (Check 3rd element, again, as it has been swapped).

✓

8) 3, 1, 2, 5, ④, 8, 9, 5, 7

a)
? (Check 4th element)

× 5 > ④

5 ② ④

①

3, 1, 2, ④, 5, 8, 9, 5, 7

9) a)
② → pivot index = cursor index

to-the-left is smaller & to-the-right is ~~greater~~ bigger.

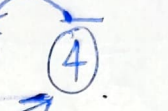10) Split array around pivot (but _not_ including the pivot point in the splits).

11) Use split halves as parameters for the quick-sort() function again to recurse 1 level deeper.

~~12)/13) Check list~~

3, 7, 8, 5, 2, 1, 9, 5, ④ → pivot

1) a) ? (check 1st element)
   b) ✓ (3 < ④)

2) a) ? (check 2nd element) ③
   b) 7 > ④
      7

5 ④ ②

7 greater than pivot, move to pivot.
Move pivot left 1.
Move element that was left 1 to Old pivot position to 7.

(5 is left 1)

3) a) 3, 5, 8, 5, 2, 1, 9, ~~5~~, ④, 7

a) ? check 2nd element again (as it has been swapped)
   ✗ 5 > ④ ③   ②

b) 5 ←   9   ④

4) 3, 9, 8, 5, 2, 1, ④, 5, 7

a) ? (check 2nd element again, as it has been swapped again).
   9 > ④   ③   ②

b) 9   1 ④

→ pivot

5) 3, 1, 8, 5, 2, ④, 9, 5, 7

a) ? (Check 2nd element, again, bla bla...)
   ✓

TURN OVER