

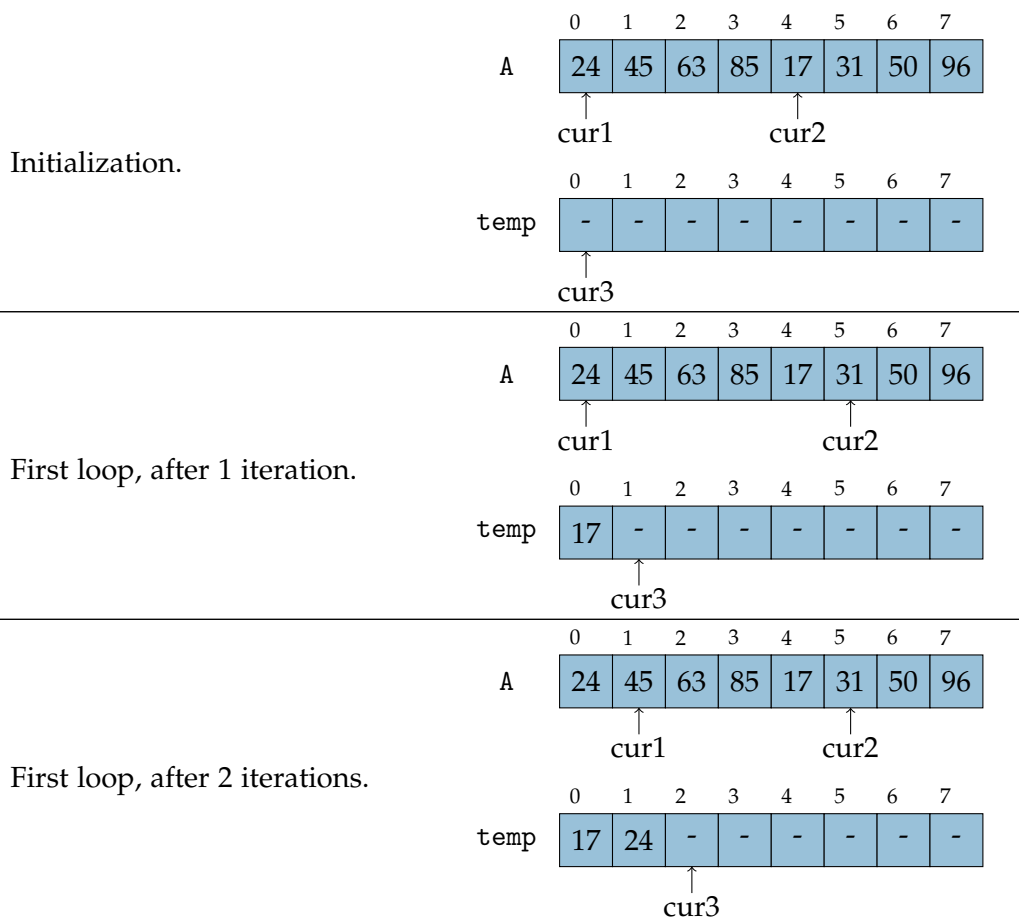
Lecture 23 Exercise Solutions

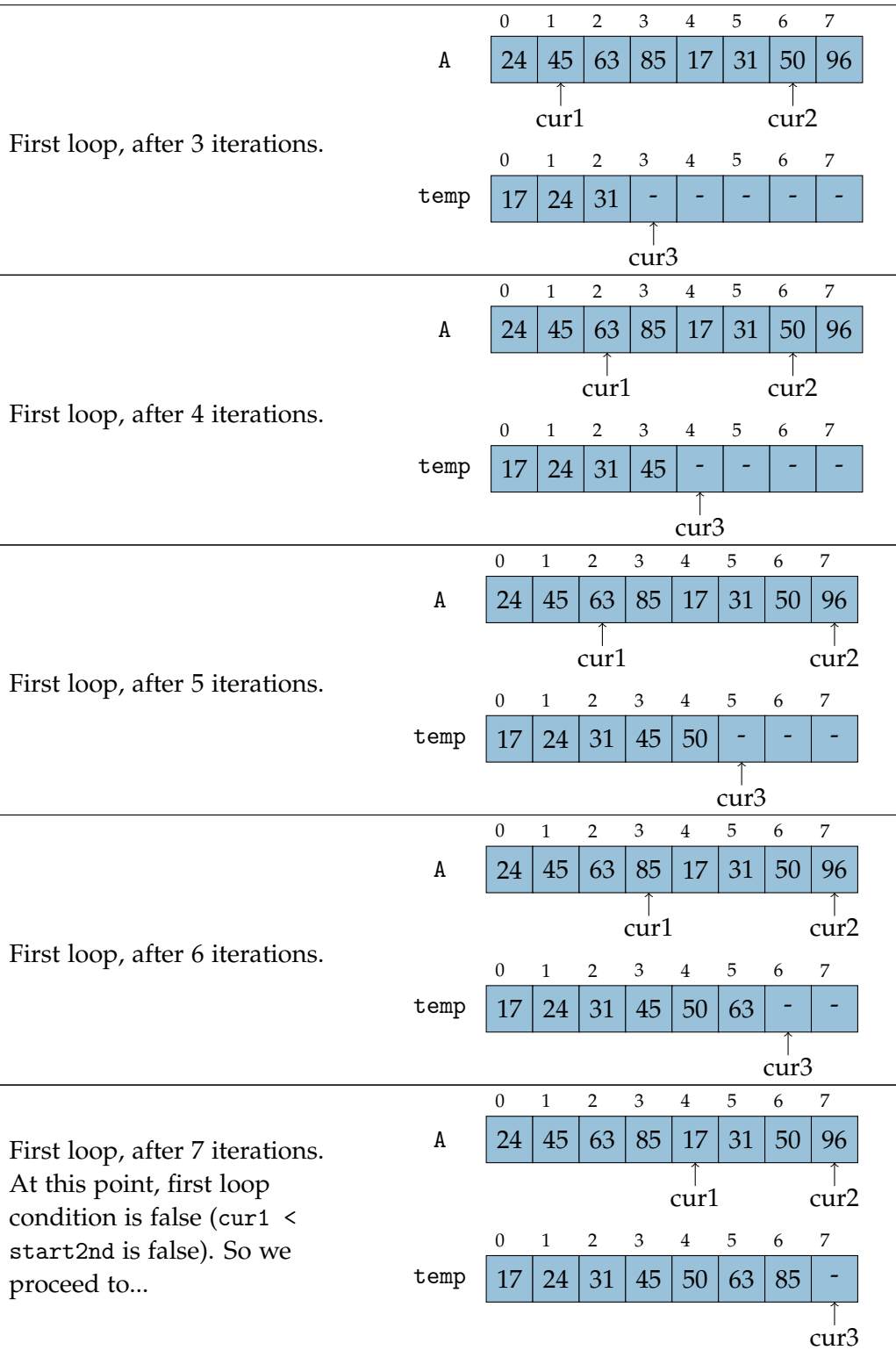
Mark Eramian

Exercise 1

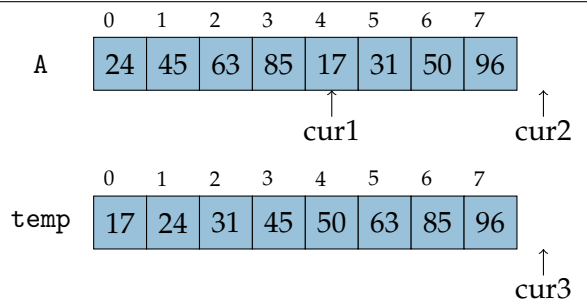
a) // Assuming A is the array to be sorted...
int[] temp = new int[A.length];
merge(A, temp, 0, 4, 7);

b) Step-by-step merge operation:

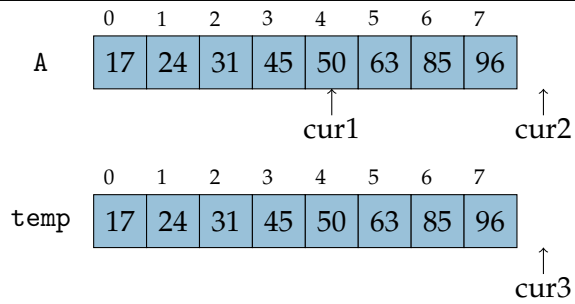




... second loop! It's condition is false so we end up in the third loop. Arrays shown after 1 iteration of third loop.

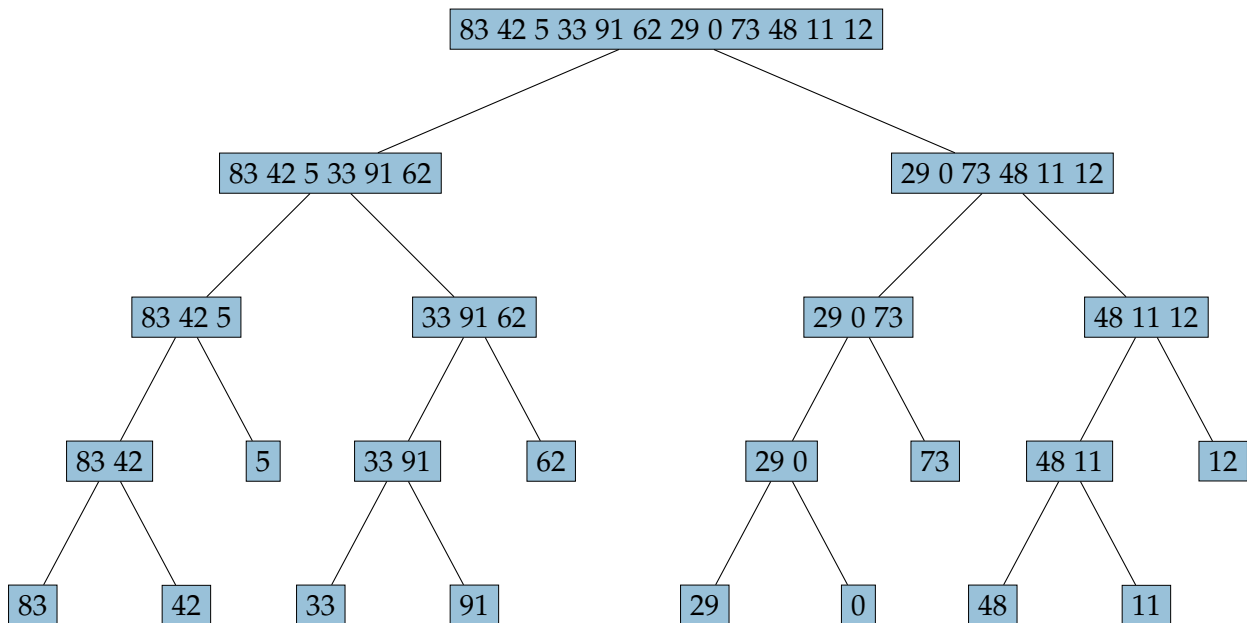


The third loop condition is now false, so we copy temp back into A and we're done. A is sorted!

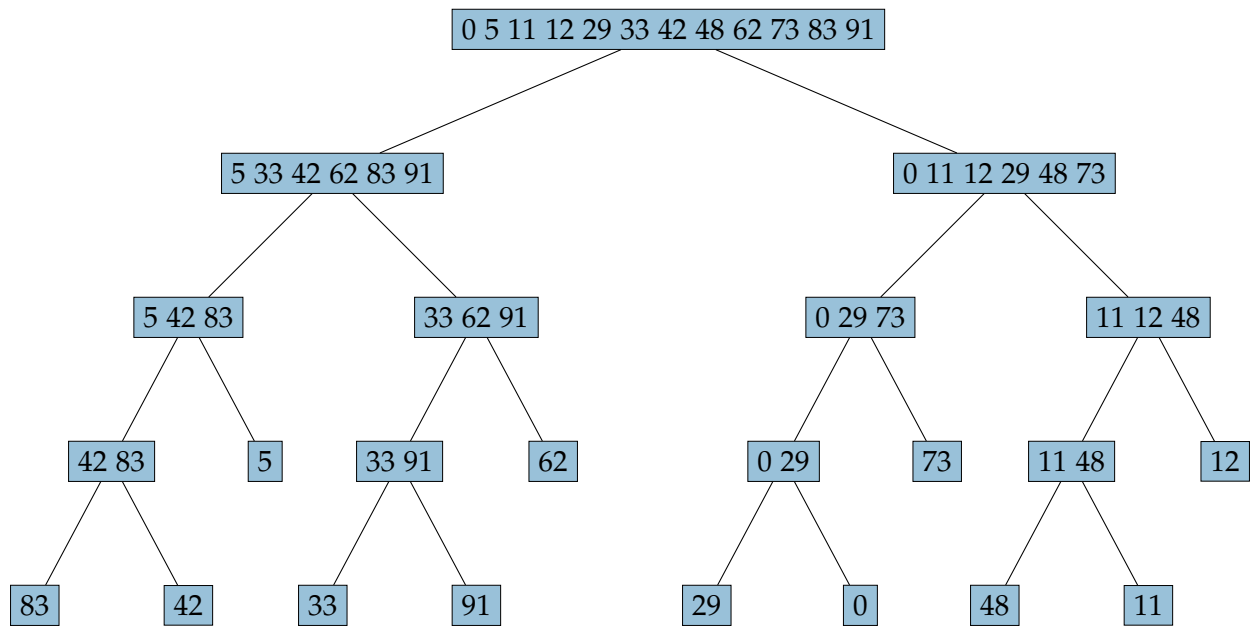


Exercise 2

The merge sort tree – input sequences:



The merge sort tree – output sequences:



Exercise 3

a) // A is the array to be sorted
 partition(A, 0, 7);

b) Step-by-step trace of partitioning:

1. Initial Setup

0	1	2	3	4	5	6	7
19	42	77	71	41	9	83	68
$l = 0$		$r = 6$					

2. Advance l and r

0	1	2	3	4	5	6	7
19	42	77	71	41	9	83	68
$l = 2$		$r = 5$					

3. Swap

0	1	2	3	4	5	6	7
19	42	9	71	41	77	83	68
$l = 2$		$r = 5$					

4. Advance l and r

0	1	2	3	4	5	6	7
19	42	9	71	41	77	83	68
$l = 3$		$r = 4$					

5. Swap

0	1	2	3	4	5	6	7
19	42	9	41	71	77	83	68
$l = 3$		$r = 4$					

6. Advance l and r ,
now $l > r$

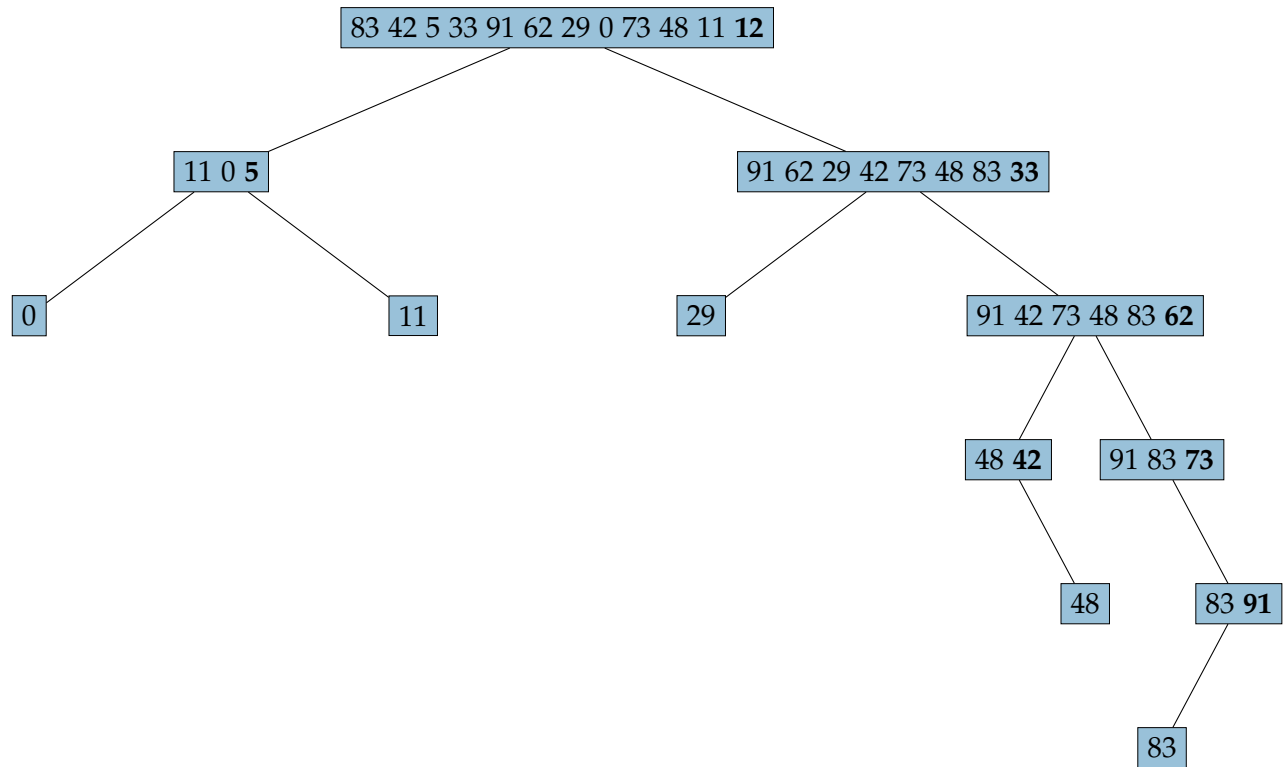
0	1	2	3	4	5	6	7
19	42	9	41	71	77	83	68
$r = 3$		$l = 4$					

7. Swap l with
pivot.

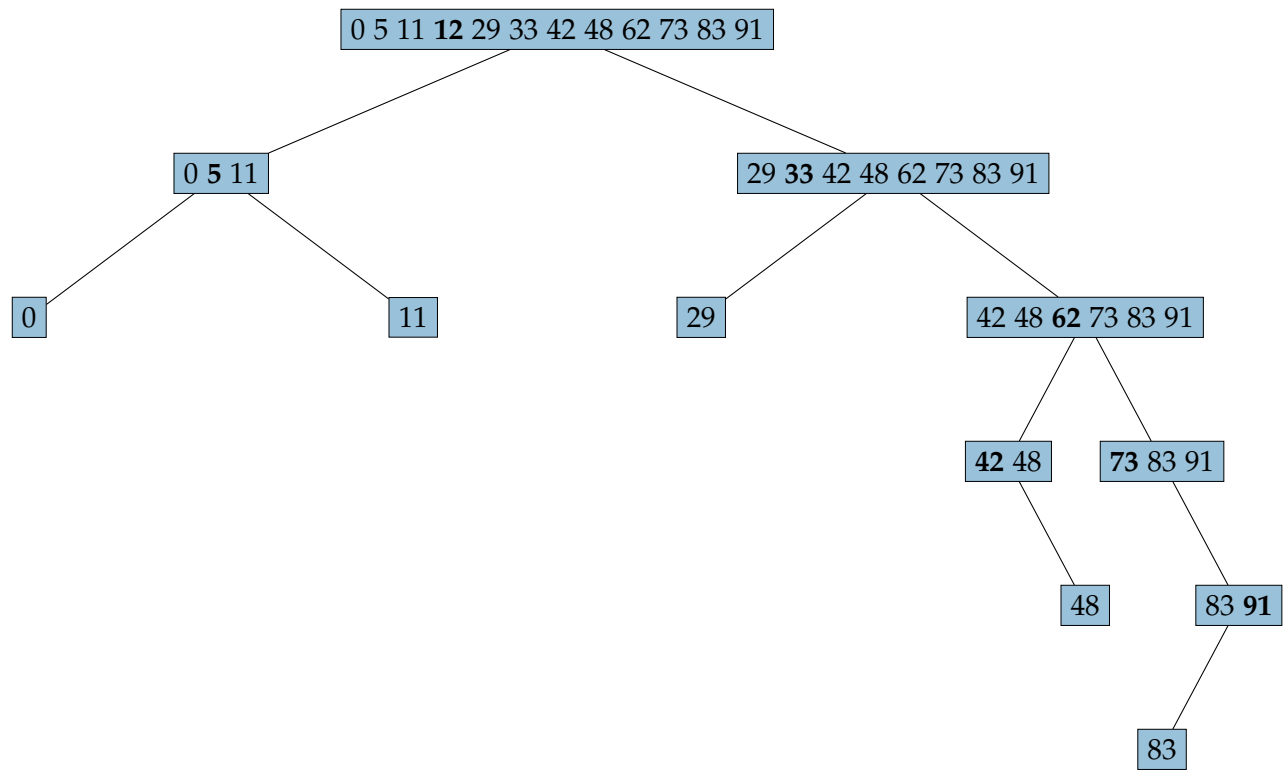
0	1	2	3	4	5	6	7
19	42	9	41	68	77	83	71
$r = 3$		$l = 4$					

Exercise 4

Quick sort tree – input sequences, pivots shown in bold:



Quick sort tree – output sequences, pivots shown in bold:



Exercise 5

1. Initialization,
 $i = 11/2 = 5$

0	1	2	3	4	5	6	7	8	9	10	11
83	42	5	33	91	62	29	0	73	48	11	12

2. `moveDown()` for
 $i = 5$, 0 swaps; 62
is bigger than 12.

0	1	2	3	4	5	6	7	8	9	10	11
83	42	5	33	91	62	29	0	73	48	11	12

3. `moveDown()` for
 $i = 4$, 0 swaps;
91 is bigger than 48
and 11.

0	1	2	3	4	5	6	7	8	9	10	11
83	42	5	33	91	62	29	0	73	48	11	12

4. `moveDown()` for
 $i = 3$, 1 swap

0	1	2	3	4	5	6	7	8	9	10	11
83	42	5	73	91	62	29	0	33	48	11	12

5. `moveDown()` for
 $i = 2$, 2 swaps.

0	1	2	3	4	5	6	7	8	9	10	11
83	42	62	73	91	12	29	0	33	48	11	5

6. `moveDown()` for
 $i = 1$, 2 swaps.

0	1	2	3	4	5	6	7	8	9	10	11
83	91	62	73	48	12	29	0	33	42	11	5

7. `moveDown()` for
 $i = 0$, 1 swap.

0	1	2	3	4	5	6	7	8	9	10	11
91	83	62	73	48	12	29	0	33	42	11	5

8. The array is now
a heap.

0	1	2	3	4	5	6	7	8	9	10	11
91	83	62	73	48	12	29	0	33	42	11	5

Exercise 6

1. Starting with the heap from step 8 of Ex. 5., initialize $i = 11$

0	1	2	3	4	5	6	7	8	9	10	11
91	83	62	73	48	12	29	0	33	42	11	5

2a. Delete root of heap by swapping with index i ; decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
5	83	62	73	48	12	29	0	33	42	11	91

2b.
moveDown(data, 0, i=10).

0	1	2	3	4	5	6	7	8	9	10	11
83	73	62	33	48	12	29	0	5	42	11	91

3a. Delete root of heap by swapping with index $i = 10$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
11	73	62	33	48	12	29	0	5	42	83	91

3b.
moveDown(data, 0, i=9).

0	1	2	3	4	5	6	7	8	9	10	11
73	48	62	33	42	12	29	0	5	11	83	91

4a. Delete root of heap by swapping with index $i = 9$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
11	48	62	33	42	12	29	0	5	73	83	91

4b.
moveDown(data, 0, i=8).

0	1	2	3	4	5	6	7	8	9	10	11
62	48	29	33	42	12	11	0	5	73	83	91

5a. Delete root of heap by swapping with index $i = 8$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
5	48	29	33	42	12	11	0	62	73	83	91

5b.
moveDown(data, 0, i=7).

0	1	2	3	4	5	6	7	8	9	10	11
48	42	29	33	5	12	11	0	62	73	83	91

6a. Delete root of heap by swapping with index $i = 7$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
0	42	29	33	5	12	11	48	62	73	83	91

6b.
moveDown(data, 0, i=6).

0	1	2	3	4	5	6	7	8	9	10	11
42	33	29	0	5	12	11	48	62	73	83	91

7a. Delete root of heap by swapping with index $i = 6$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
11	33	29	0	5	12	42	48	62	73	83	91

7b.
moveDown(data, 0, i=5).

0	1	2	3	4	5	6	7	8	9	10	11
33	11	29	0	5	12	42	48	62	73	83	91

8a. Delete root of heap by swapping with index $i = 5$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
12	11	29	0	5	33	42	48	62	73	83	91

8b.
moveDown(data, 0, i=4).

0	1	2	3	4	5	6	7	8	9	10	11
29	11	12	0	5	33	42	48	62	73	83	91

9a. Delete root of heap by swapping with index $i = 4$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
5	11	12	0	29	33	42	48	62	73	83	91

9b.
moveDown(data, 0, i=3).

0	1	2	3	4	5	6	7	8	9	10	11
12	11	5	0	29	33	42	48	62	73	83	91

10a. Delete root of heap by swapping with index $i = 3$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
0	11	5	12	29	33	42	48	62	73	83	91

10b.
moveDown(data, 0, i=2).

0	1	2	3	4	5	6	7	8	9	10	11
11	0	5	12	29	33	42	48	62	73	83	91

11a. Delete root of heap by swapping with index $i = 2$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
5	0	11	12	29	33	42	48	62	73	83	91

11b.
moveDown(data,
0, i=1).

0	1	2	3	4	5	6	7	8	9	10	11
5	0	11	12	29	33	42	48	62	73	83	91

12a. Delete root of heap by swapping with index $i = 1$. Decrement i .

0	1	2	3	4	5	6	7	8	9	10	11
0	5	11	12	29	33	42	48	62	73	83	91

12b.
moveDown(data,
0, i=0).

0	1	2	3	4	5	6	7	8	9	10	11
0	5	11	12	29	33	42	48	62	73	83	91

3. $i = 0$ so the loop stops and the array is sorted.

0	1	2	3	4	5	6	7	8	9	10	11
0	5	11	12	29	33	42	48	62	73	83	91

