Feedback — Mergesort

Help Center

You submitted this quiz on **Mon 21 Sep 2015 7:14 PM EDT**. You got a score of **2.60** out of **3.00**. You can attempt again, if you'd like.

```
To specify an array or sequence of values in an answer, separate the value
s in
the sequence by whitespace. For example, if the question asks for the firs
t
ten powers of two (starting at 1), then the following answer is acceptable
:

     1 2 4 8 16 32 64 128 256 512

If you wish to discuss a particular question and answer in the forums, ple
ase
post the entire question and answer, including the seed (which can be used
 by
the course staff to uniquely identify the question) and the explanation (w
hich
contains the correct answer).
```

Question 1

```
(seed = 547624)
Give the array that results immediately after the 7th call (and return)
from merge() when top-down mergesorting the following array of size 12:

    35 79 11 95 78 68 33 83 27 39 28 52

Your answer should be a sequence of 12 integers, separated by whitespace.
```

You entered:

```
11   35   68   78   79   95   27   33   83   39   28   52
```

Your Answer		Score	Explanation
11 35 68 78 79 95 27 33 83 39 28 52	✓	1.00	
Total		1.00 / 1.00	

Question Explanation

```
The correct answer is: 11 35 68 78 79 95 27 33 83 39 28 52

Here is the array immediately after each call to merge():

                   35 79 11 95 78 68 33 83 27 39 28 52
merge(0, 0, 1):  35 79 11 95 78 68 33 83 27 39 28 52
merge(0, 1, 2):  11 35 79 95 78 68 33 83 27 39 28 52
merge(3, 3, 4):  11 35 79 78 95 68 33 83 27 39 28 52
merge(3, 4, 5):  11 35 79 68 78 95 33 83 27 39 28 52
merge(0, 2, 5):  11 35 68 78 79 95 33 83 27 39 28 52
merge(6, 6, 7):  11 35 68 78 79 95 33 83 27 39 28 52
merge(6, 7, 8):  11 35 68 78 79 95 27 33 83 39 28 52
```

Question 2

(seed = 16382)
Give the array that results immediately after the 7th call (and return) from merge() when bottom-up mergesorting the following array:

21 71 52 98 49 18 50 29 11 15

Your answer should be a sequence of 10 integers, separated by whitespace.

You entered:

```
21   52   71   98   18   29   49   50   11   15
```

Your Answer		Score	Explanation
21 52 71 98 18 29 49 50 11 15	✓	1.00	
Total		1.00 / 1.00	

Question Explanation

```
The correct answer is: 21 52 71 98 18 29 49 50 11 15

Here is the array immediately after each call to merge():

                 21 71 52 98 49 18 50 29 11 15
merge(0, 0, 1):  21 71 52 98 49 18 50 29 11 15
merge(2, 2, 3):  21 71 52 98 49 18 50 29 11 15
merge(4, 4, 5):  21 71 52 98 18 49 50 29 11 15
merge(6, 6, 7):  21 71 52 98 18 49 29 50 11 15
merge(8, 8, 9):  21 71 52 98 18 49 29 50 11 15
merge(0, 1, 3):  21 52 71 98 18 49 29 50 11 15
merge(4, 5, 7):  21 52 71 98 18 29 49 50 11 15
```

Question 3

(seed = 526958)
Which of the following statements about mergesort are true? Check all that apply. Unless otherwise specified, assume that mergesort refers to the pure recursive (top-down) version of mergesort (with no optimizations), using the merging subroutine described in lecture.

Your Answer		Score	Explanation
☑ The number of compares to mergesort an array of N/2 1s interleaved with N/2 0s (e.g.	✗	0.00	In general, each merge involves two subarrays of the form 00001111 (i.e., 1/2 N 0s followed by 1/2 N 1s). This takes ~ 3/4 N compares because the left subarray is exhausted with 1/4 N 1s remaining in the right subarray. Thus, the total number of compares satisfies the recurrence T(N) = 2 T(N/2) + 3/4 N, which yields T(N) ~ 3/4 N lg N.

, 1 0 1 0 1 0 1 0 1 0) is ~ 1 N lg N.			
☑ For any array of N distinct keys with N a power of 2, top-down mergesort and bottom-up mergesort compare exactly the same pairs of keys (but possibly in a different order).	✓	0.20	This can be proved by induction - in either version of mergesort, all of the subarray sizes are powers of 2.
□ Suppose we have a sorting algorithm that in addition to regular compares, is also allowed super-compares: take three keys and return those three keys in sorted order. Then, any compare-based sorting algorithm requires at least lg (N!) compares or super-compares (i	✓	0.20	Similar to the lower bound argument with 2-way compares, but now the height of the tree is at least log_6 (N!) since each node has as many as 6 children, corresponding to the 3! possible outcomes for each super-compare.

n the worst case) to sort an array of N items.			
□ It is possible to design a compare-based algorithm to merge two sorted arrays, each of size N, with no more than 3/2 N compares.	✓	0.20	Any merging algorithm requires at least 2N-1 compares in the worst case to merge two sorted arrays of the form x = [0, 2, 4, ..., N-2] and y = [1, 3, 5, ..., N-1] because x[i] must be compared with both y[i-1] and y[i+1]. Alternatively, such a merging algorithm would lead to a compare-based sorting algorithm that guarantees to make no more than ~ 3/4 N lg N compares, which would violate the sorting lower bound.
☑ The number of compares in mergesort depends only on the size of the array N (and not on the items in the array).	✗	0.00	The number of compares ranges from ~ 1/2 N lg N (sorted array) to ~ N lg N (random array).
Total		0.60 / 1.00	

Question Explanation