1. Consider the array with numbers that is input to the max subarray problem

6 / 6 points

[1, 19, 5, -4, 7, 18, 15, -10]

Select all true facts from the list below making sure that no incorrect choices are selected.

- [x] The output .to the max subarray .problem should be 18 - (-4) = 22

  Correct
  Correct.

- [ ] The max subarray problem can be solved in linear time by simply taking the difference between the largest and smallest elements in the array.

- [x] The divide and conquer algorithm will compute the result of max subarray problem on the first half of the array, which in this instance yields the value 18

  Correct
  Correct: 19 - 1 = 18

- [x] The divide and conquer algorithm will compute the result of max subarray problem on the second half of the array, which in this instance yields the value 11

  Correct

- [ ] For solving the max subarray problem, it is sufficient to recursively solve the problem for left and right halves of the given array and take the maximum among the two.

- [x] The minimum element of the first half of the array is -4 and maximum element of the second half of the array is 18. These in turn form the result for the max subarray problem which is 22.

  Correct
  Correct.

2. Consider the recurrence that represents the running time for the max subarray problem: **1 / 1 point**

$$T(n) = \begin{cases} \Theta(1) & n \leq 2 \\ 2T(\frac{n}{2}) + \Theta(n) & \text{otherwise} \end{cases}$$

- [x] The case when $n \leq 2$ represents the constant amount of work needed to find the max subarray for input arrays of size 1 or 2.

  Correct
  Correct: as explained in the lecture.

- [x] The recurrence assumes that n is a power of two, since repeated division by 2 can yield fractional results otherwise.

  Correct
  Correct: we can always pad an array with dummy elements of $-\infty$ so that its size is a power of two and the result of max-subarray does not change. Doing so will not more than double the number of elements in the worst case.

- [x] The $\Theta(n)$ term in the recurrence for $n > 2$ represents the work to find minimum of first half and maximum of second half.

  Correct
  Correct

- [ ] The recurrence is identical to that obtained for binary search algorithm.

- [x] The recurrence and the running time are identical to that obtained for the mergesort algorithm covered earlier in course 1 of this specialization.

  Correct
  Correct