Daniel Tran

CSE 417

2 Feb. 2017

**Homework 4: Divide and Conquer Questions**

1. **What is the base case for this recursion? How do you solve the problem in that case in O(1) time?**
2. **As described in the**[**next section**](https://courses.cs.washington.edu/courses/cse417/18wi/hws/hw4/hw4.html#combine)**, our combine step will use a "two finger" algorithm with each finger pointing to a range in one of the two lists and maintaining the invariant that the range under the right finger points to the longest one that can be concatenated with the range under the left finger to produce a sideways trend.**

**On the next iteration, when we advance the left finger to the next range in the left list, the right finger will always to move in one particular direction (or not move at all). What direction is that (to lower indexes or higher ones), and why do we know this is the case?**

1. **Why is the total number of finger moves O(n)? Furthermore, if we implement this with the two fingers stored as integer indexes in the code, why does all the work of moving the indexes, concatenating ranges, and checking whether they define sideways trends take only O(n) time?**
2. **Why do we know that the longest sideways trend crossing the midpoint is one that we get by concatenating a range under the left finger and a range under the right finger satisfying the invariant above? (I.e., why is this algorithm correct when we're not actually considering every combination of gold and purple ranges?)**
3. **Extra Credit: As usual when designing a divide and conquer algorithms, now that we are done, we should step back and consider whether the problem is truly easier to solve with divide and conquer than it is to solve directly. Would you say that that true in this case? Make a convincing argument for why that is or is not the case here.**