CSE 332 Data Abstractions, Winter 2013

Homework 6

Due: **Friday, March 1, 2013** at the BEGINNING of lecture. Your work should be readable as well as correct. You should refer to the written **homework** guidelines on the course website for a reminder about what is acceptable pseudocode. You are going to love the fact that homework Six has TWO (really, only 2) terrific questions!! Please write your **section** at the top of your homework.

Problem 1: Forkjoin Parallelism: Longest Series

Consider the problem of finding the longest sequence of some number in an array of numbers: `longest_sequence(i,arr)` returns the longest number of consecutive i in arr. For example, if arr is {2,17,17,8,17,17,17,0,17,1} then `longest_sequence(17,arr)` is 3 and `longest_sequence(9,arr)` is 0.

a) In pseudocode, give a parallel fork-join algorithm for implementing longest sequence. Your algorithm should have work O(n) and span O(log n) where n is the length of the array. Do *not* employ a sequential cut-off: your base case should process an array range containing one element. Hint: Use this definition:

```
class Result {
      int numLeftEdge;
      int numRightEdge;
      int numLongest;
      boolean entireRange;
      Result(int l, int r, int m, boolean a) {
            numLeftEdge=l; numRightEdge=r;
            numLongest=m; entireRange=a;
      }
}
```

For example, `numLeftEdge` should represent the length of the sequence at the *beginning* of the range processed by a subproblem. Think carefully about how to combine results; given left and right 'Result's, how can you compute the merged 'Result'?

b) In English, describe how you would make your answer to part (a) more efficient by using a sequential cut-off. In pseudocode, show the code you would use below this cut-off.

Problem 2: Forkjoin Parallelism: Leftmost Occurrence of Substring

Consider the problem of finding the leftmost occurrence of the sequence of characters `cseRox` in an array of characters, returning the index of the leftmost occurrence or -1 if there is none. For example, the answer for the sequence `cseRhellocseRoxmomcseRox` is 9.

a) In English (though some high-level pseudocode will probably help), describe a fork-join algorithm similar in design to your solution in problem 1. Use a sequential cut-off of at least 12 (that is, problems of size 11 or smaller should be solved sequentially) and explain why this significantly simplifies your solution. Notice you still must deal with the leftmost occurrence being "split" across two recursive subproblems.

b) Give a much simpler fork-join solution to the problem that avoids the possibility of a "split" by using slightly overlapping subproblems. Assume a larger sequential cut-off, for example 100. Give your solution precisely in pseudocode. Watch out for off-by-one errors.