CSCI 3104: Algorithms **Instructor Buxton**
Example Proofs **Summer 2019, CU-Boulder**

1. Insertion Sort
   (a) Loop Invariant
   (b) Resource proves that Insertion sort will correctly sort a given array of elements (A). The Loop Invariant (i.e. the assertion we make that remains true after every iteration) is that the elements in the subarray $A[0 - i - 1]$ are sorted.
   (c) Reference: University of British Columbia
2. Merge Sort
   (a) Loop Invariant followed by a Proof by Induction
   (b) Resource proves first that the subroutine Merge correctly takes two sorted subarrays and merges them into a single sorted array. It then uses a Proof by Strong Induction to show that the outer, MergeSort function properly sorts a given array (A).
   (c) Reference: McGill University
3. Quick Sort
   (a) Loop Invariant followed by a Proof by Weak Induction
   (b) First resource (Quicksort_Partition) proves using a loop invariant that Partition correctly orders a subarray into elements less than the pivot located at indices less than the index of the pivot, the pivot located at it's final index in the entire sorted array, and elements greater than the pivot located at indices greater than the index of the pivot. After proving this, the second resource (Quicksort_Correctness) uses a proof by induction and the knowledge that Partition is correct in order to prove the entire correctness of the algorithm.
   (c) Reference 1: George Mason University
   (d) Reference 2: McGill University
4. Huffman Encoding
   (a) Direct Proof, followed by a Proof by Contradiction, followed by a Proof by Weak Induction
   (b) Resource first proves the a Lemma (Lemma 1) that will be used in the next proof. It's goal is to show that an equation holds when the operation of swapping two nodes in the tree (representing our compression scheme) is performed. Next,

a proof by contradiction is performed to show that the problem has optimal substructure (i.e. showing that an optimal tree contains the two lowest frequent symbols as siblings at the greatest depth in the tree). Finally, a proof by weak induction (claim holds for a tree of $n-1$ nodes) is shown to prove that the algorithm has the greedy choice property and leads to the generation of an optimal tree.

(c) Reference: University of Toronto

(d) Video Reference: Coursera: Stanford University (1 of 4)

5. 0-1 Knapsack

   (a) Proof by Contradiction, followed by a Direct Proof, should be followed by a Proof by Strong Induction

   (b) Resource first proves that the problem exhibits optimal substructure using a proof by contradiction. Much later in the resource, a direct proof is given to prove that the generated recursive definition holds. To be as formal as possible, a proof by strong induction should've followed using the Lemma proven in the previous section to prove the correctness of the entire algorithm.

   (c) Reference: Santa Fe Institute(CU Boulder)

6. Longest Common Subsequence

   (a) Proof by Contradictions followed by a Proof by Strong Induction

   (b) Resource 1 (LCS_Optimal_Substructure) first proves using proof by contradictions to show that this problem exhibits optimal substructure. Resource 2 (LCS_Correctness) then uses a proof by strong induction, using the knowledge that this problem has optimal substructure in order to prove the correctness of this algorithm.

   (c) Reference 1: Columbia University

   (d) Reference 2: University of Chicago

7. Sequence Alignment

   (a) Should contain Proof by Contradictions followed by a Proof by Strong Induction

   (b) The reference does not first prove that the problem exhibits optimal substructure which should be performed. It implicitly uses this knowledge to prove using strong induction over the contents of the two dimensional that this algorithm is correct.

   (c) Reference: CU Boulder

8. BFS and DFS
   (a) Proof by induction
   (b) Resource proves that BFS finds the shortest path from a single source node to all other nodes in an undirected, simple graph.
   (c) Reference: New York CS
9. Dijkstra's (SSSP)
   (a) Proof by strong induction
   (b) Resource proves that Dijkstra's finds the shortest path from a single source node to all other nodes in a directed, simple graph. Assumes no negative edge weights are present in the edge set
   (c) Reference 1: University of California San Diego
   (d) Reference 2: Oregon State University