Christian Rodriguez

Algorithms HW 1

1. Prove that the following algorithm sorts its input data; i.e., that data [1] _
data [2] _ : : : _ data [n] when the algorithm terminates. You may assume
that data contains at least one element. Also, bxc represents the
 function, which returns the largest integer less than or equal to the given
value (e.g., b 3: 1415c = 3).

    Input : data : an array of integers
    Input : n : the length of data
    Output : a reordering of data in (ascending) sorted order
    1 Algorithm: ThirdSort
    2 if n = 1 then
        3 return data
    4 else if n = 2 then
        5 if data [1] > data [2] then
        6 Swap data [1] and data [2]
        7 end
        8 return data
    9 else
        10 third = bn= 3c
        11 Call ThirdSort on data [1::n -third]
        12 Call ThirdSort on data [third +1::n]
        13 Call ThirdSort on data [1::n -third]
        14 return data
    15 end

Answer:
    Base Case(s):
- n = 1, data has one element, so it is already sorted and ThreeSort returns array
- n = 2, data has two elements which are compared with each other and properly swapped in the right order

    Inductive Hypothesis:
- We have a data array with elements 1, 2, 3, …, k+1 and we split this array into three "thirds" A, B, and C where AB represents the first recursion, BC represents the second recursion, and AB represents the third recursion (which is the same implementation as the first recursion).

    Inductive Step:
- ThirdSort performs a recursive call on A & B. Because k+1 >= 3, third >= 1, so n-third <= k (A & B has length n-third).
  - Because length |AB| <= k, ThirdSort sorts AB by Induction Hypothesis.
- After the first recursion each element in B >= A, and by the same arguments as AB ThirdSort correctly sorts BC.
  - … Therefore data[1] <= data[2] <= … <= data[n]
- Finally, by the same arguments as the first recursion the final recursion of AB again sorts the elements by Inductive Hypothesis.
  - B <= C and A <= C, therefore A <= C.