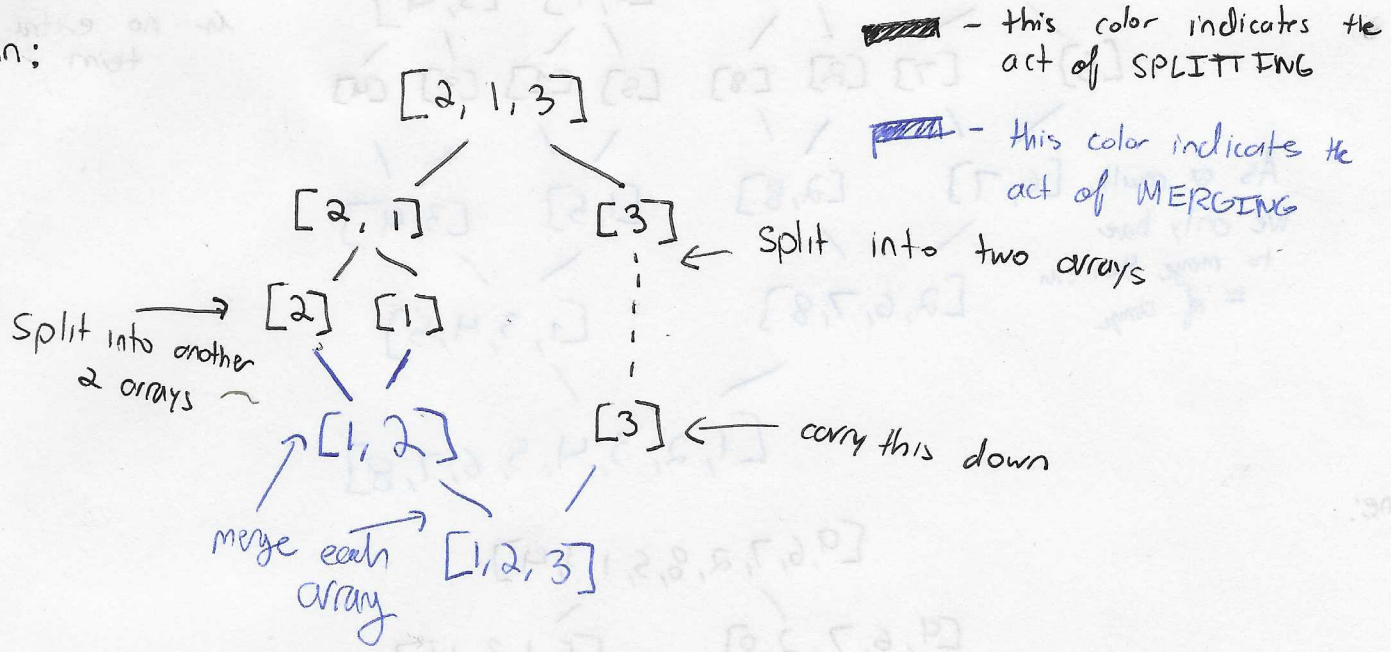
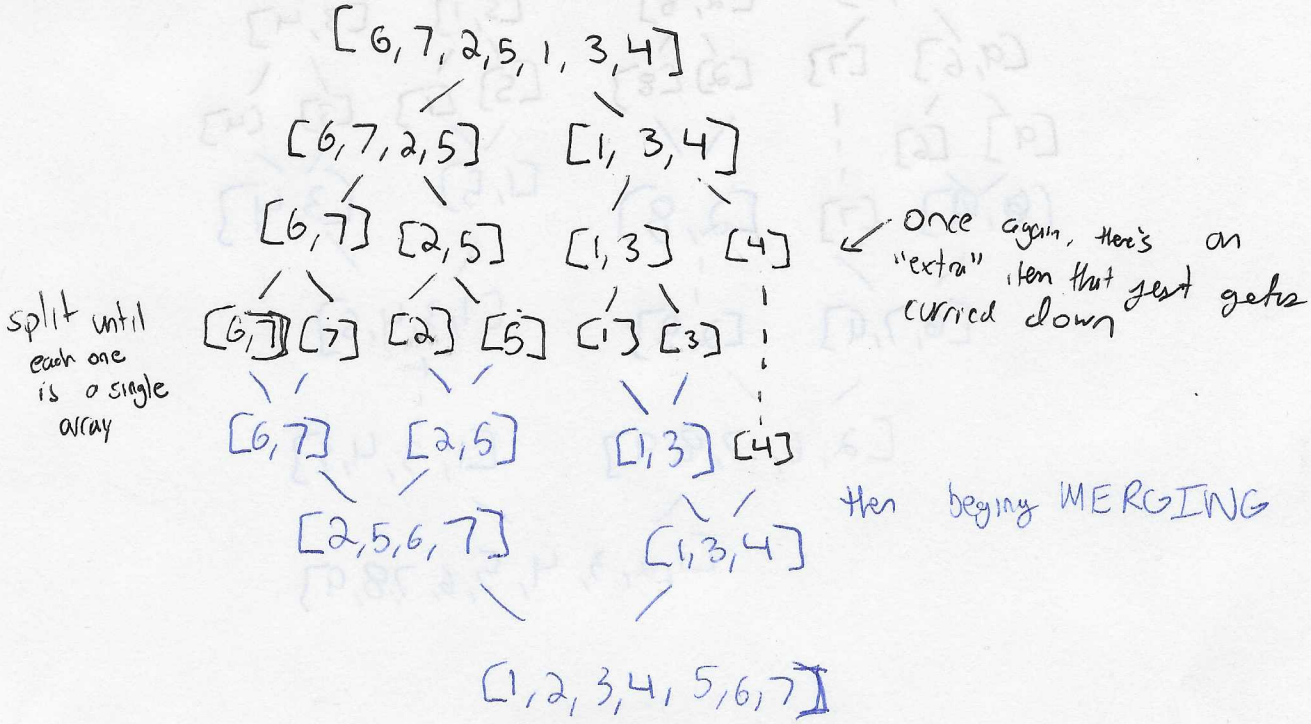
Rubin Peci
APCS-2 Pd.2
HW #06 - Nyoom
2018-02-12

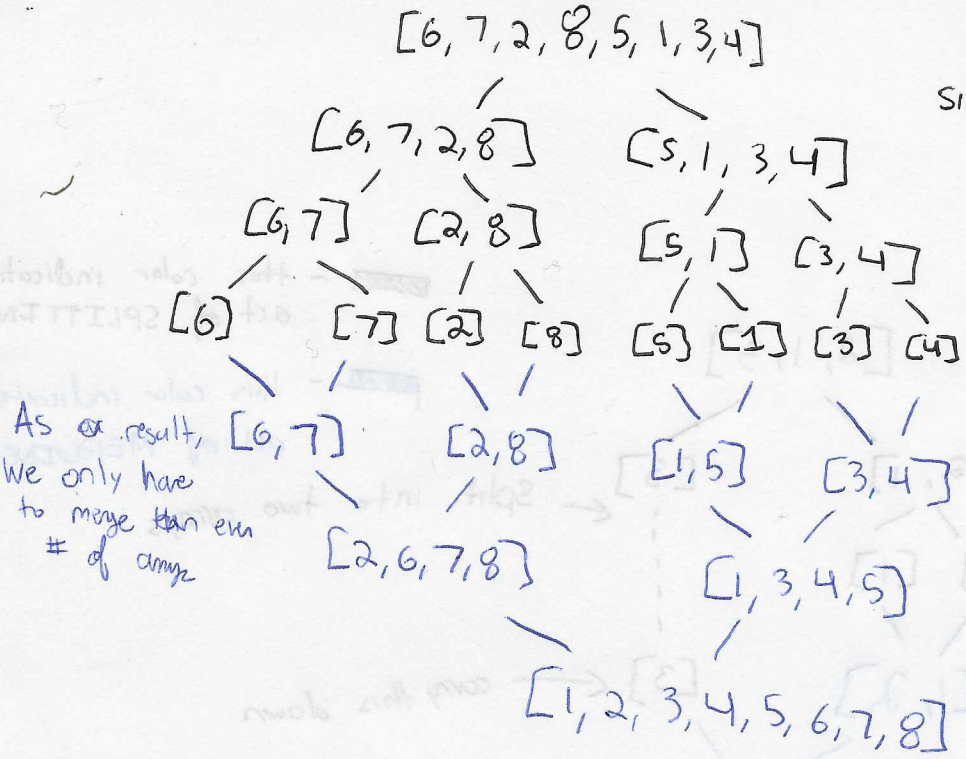
1. Win:



2. Tim:



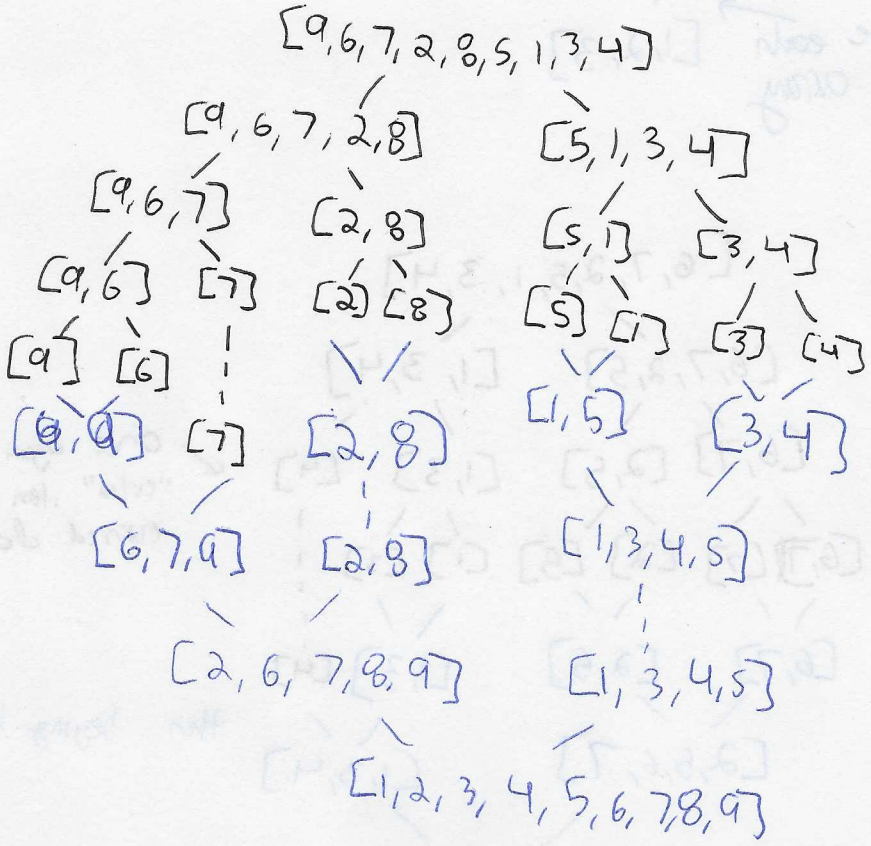
3. Jeremy:

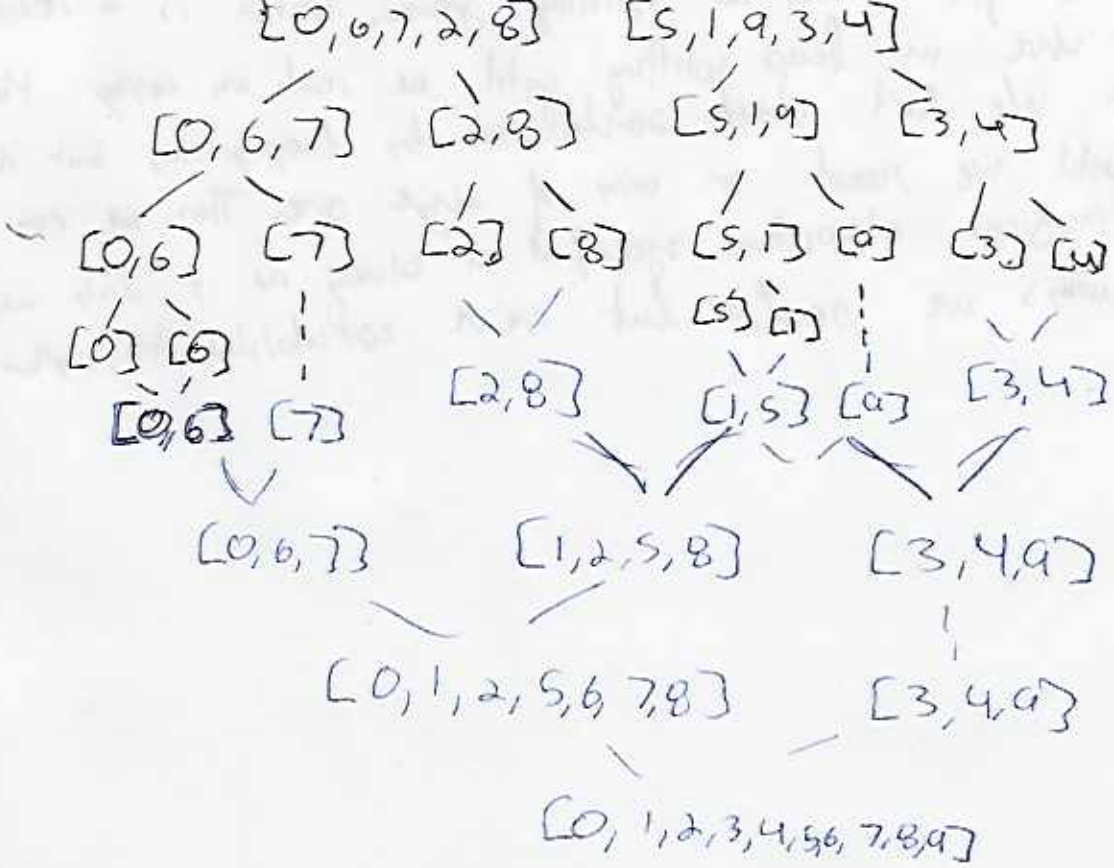


As a result, We only have to merge two even # of arrays

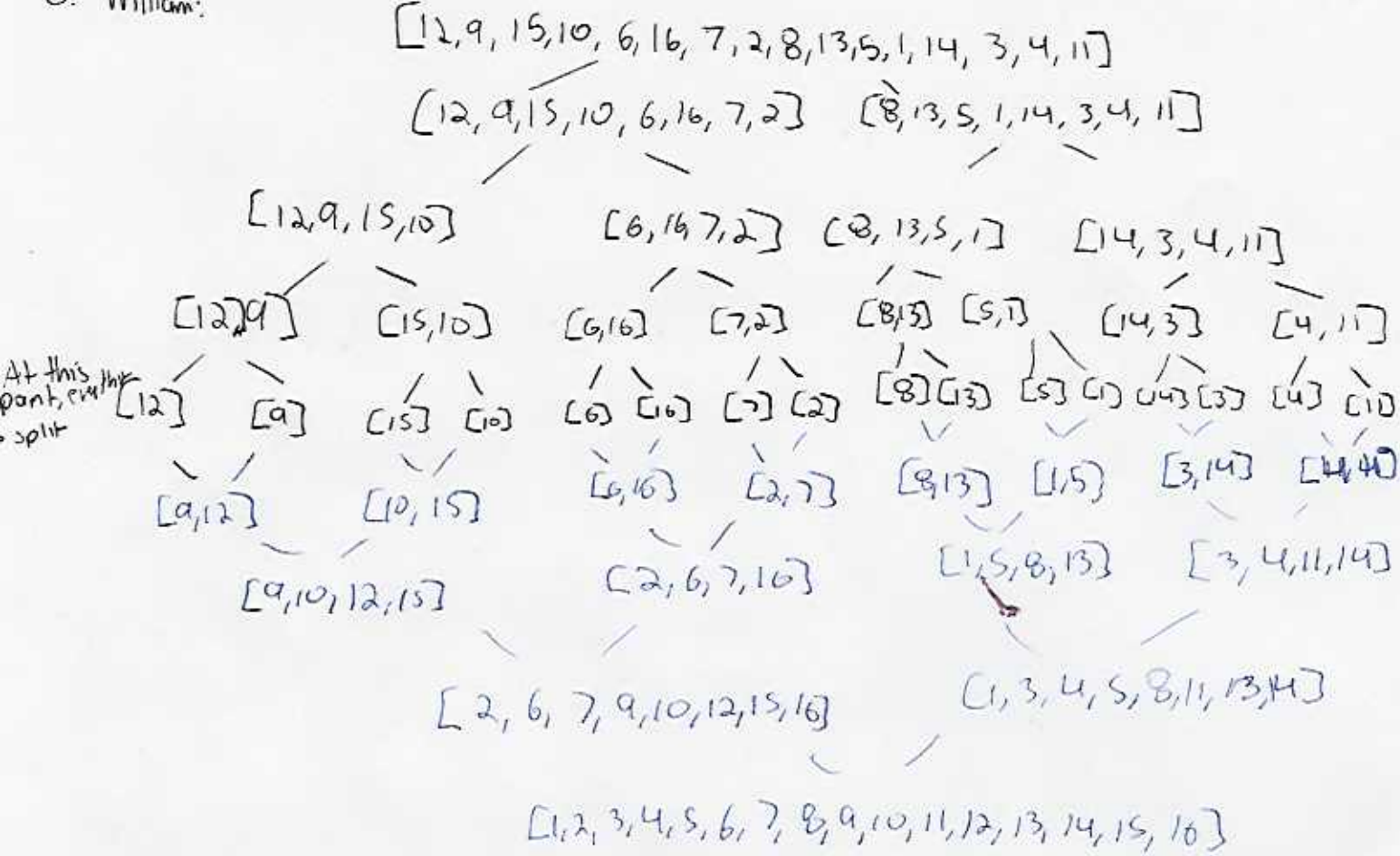
Since this one has an even-n, it's a bit "cleaner", in the sense that there is no extra term

4. Regine:




[0,6,7,2,8]
[5,1,9,3,4]
[0,6,7]
[2,8]
[5,1,9]
[3,4]
[0,6]
[7]
[2]
[8]
[5,1]
[9]
[3]
[4]
[0]
[6]
[5]
[1]
[0,6]
[7]
[2,8]
[1,5]
[9]
[3,4]
[0,6,7]
[1,2,5,8]
[3,4,9]
[0,1,2,5,6,7,8]
[3,4,9]
[0,1,2,3,4,5,6,7,8,9]

6. William:


[12,9,15,10,6,16,7,2,8,13,5,1,14,3,4,11]
[12,9,15,10,6,16,7,2]
[8,13,5,1,14,3,4,11]
[12,9,15,10]
[6,16,7,2]
[8,13,5,1]
[14,3,4,11]
[12,9]
[15,10]
[6,16]
[7,2]
[8,13]
[5,1]
[14,3]
[4,11]
At this point, every thing split
[12]
[9]
[15]
[10]
[6]
[16]
[7]
[2]
[8]
[13]
[5]
[1]
[14]
[3]
[4]
[11]
[9,12]
[10,15]
[6,16]
[2,7]
[8,13]
[1,5]
[3,14]
[4,11]
[9,10,12,15]
[2,6,7,16]
[1,5,8,13]
[3,4,11,14]
[2,6,7,9,10,12,15,16]
[1,3,4,5,8,11,13,14]
[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]

The way our mergesort algo works is in two main phases. The first phase, the "splitting" phase, which is a recursive phase where we keep splitting until we reach an array that is "sorted." We don't check sortedness by looping thru, but rather we split until we reach an array of size one. Then, we can use our merging algorithm, specified in class, as it only works when arrays are sorted but were established they are.