CS 146: PROJECT V WRITE-UP

RIADIANI MARCELITA

1.
   a. The program will take in an array of integers, and we will have the sequential cutoff as a set number, in our case, 500. We will then divide the array in two, the left and the right side. Using fork-join, we will then check the elements at the left and right side at the same time, treating them as our threads. If the element is smaller than 7, we increment a counter variable. We will then keep halving the array until we reach the sequential cutoff. Our answer for each thread would be our counter variable, and after the process is done, the answer will be the sum of all of the numbers in our counter variable.
   b. We generally do the same process as number 1A, but with an array containing strings instead. Instead of checking if the array elements are bigger or smaller than 7, we check if the elements are equal the word “PARALLEL” exactly. If it is, we increment the counter variable. If it is not, we keep halving the array until we reach the sequential cutoff. The answer would be the sum of the results of the counter variables.
   c. The program will take in an array of integers, and we will have the sequential cutoff as a set number. We will then divide the array in two, the left and the right side. Using fork-join, we will check the elements at the left and the right side at the same time, treating them as our threads. We divide each element of the array with two, and if the remainder is 0, which means it is an even integer, we will increment a counter variable. We will keep halving the array until we reach the sequential cutoff. Our answer or each thread would be our counter variable, and after the process is done, the answer will be the sum of all of the numbers in our counter variable. We will then check the value of our counter variable, and divide it by two. If the remainder is 0, and it is even, then we return a boolean value of true. We will return false otherwise.

2. The program will take in an array of integers. We will take each element of the array and make them into a new thread. We will then have a thread for each unique number, and each

of those threads will have a boolean value. If the array contains a duplicate, then it will return true. Else, it would return false.

3. Our program will start at the root of our tree, then will divide into two threads, left and right. We will then create new threads which each contain the node of the tree and a boolean value. We will then traverse through the tree recursively. If there is a duplicate in the tree, we will return true. Else, it would return false.