North South University

Department of Electrical and Computer Engineering

CSE 373: Design and Analysis of Algorithms

Spring 2018

Assignment 01

Problem description:

Assume you have an array ***A[1..n]*** of ***n*** elements. A majority element of ***A*** is any element occurring in more than ***n/2*** positions (so if n = 6 or n = 7, any majority element will occur in at least 4 positions). Assume that elements cannot be ordered or sorted, but can be compared for equality. Design an efficient divide and conquer algorithm to find a majority element in ***A*** (or determine that no majority element exists). Your algorithm must run in no more than ***O(nlgn)*** time.

Solution guideline: The algorithm begins by splitting the array in half repeatedly and calling itself on each half. This is similar to what is done in the merge sort algorithm. When we get down to single elements, that single element is returned as the majority of its (1-element) array. At every other level, it will get return values from its two recursive calls. The key to this algorithm is the fact that if there is a majority element in the combined array, then that element must be the majority element in either the left half of the array, or in the right half of the array. There are 4 scenarios.

a. Both return “no majority.” Then neither half of the array has a majority element, and the combined array cannot have a majority element. Therefore, the call returns “no majority.”

b. The right side is a majority, and the left isn’t. The only possible majority for this level is with the value that formed a majority on the right half, therefore, just compare every element in the combined array and count the number of elements that are equal to this value. If it is a majority element then return that element, else return “no majority.”

c. Same as above, but with the left returning a majority, and the right returning “no majority.”

d. Both sub-calls return a majority element. Count the number of elements equal to both of the candidates for majority element. If either is a majority element in the combined array, then return it. Otherwise, return “no majority.”

The top level simply returns either a majority element or that no majority element exists in the same way.

Input and output specification:

The input is a sequence of integers. The first integer ***n*** is the number of elements in the array. The next ***n*** integers are the elements themselves. The output of your program is a single integer which represents the majority element in the array. If there is no majority element, your program should print “None”. Your program must not print any extra text whatsoever.

```
Sample input: 12  3  2  1  2  4  2  2  9  2  2  1  2

Sample output: 2

Sample input: 12  3  1  1  3  4  2  2  9  6  2  1  2

Sample output: None
```

Submission instructions:

Please read carefully the following instructions on how to submit your assignment. If you make any mistake at all in the submission process, your assignment will not be marked.

Suppose your NSU student ID is 1234567890. After you complete the assignment, rename your source file (let's say main.c for example) as "1234567890.c" and upload this file on Google Classroom in assignment section. Do not send assignments as message attachment. Do not upload any additional file.

Any form of cheating will be penalized heavily. Duplicate codes (no matter if full or partial) will not be marked regardless of which one the original is.