Computer Science C73 — September 27, 2023
Scarborough Campus — University of Toronto

Homework Assignment #4
Due: October 4, 2023, by 11:59 pm

• ***You must submit your assignment through the Crowdmark system.*** *You will receive by email an invitation through which you can submit your work in the form of separate PDF docucments with your answers to each question of the assignment. To work with a partner, you and your partner must form a group on Crowdmark. Crowdmark does not enforce a limit on the size of groups.* ***The course policy that limits the size of each group to at most two remains in effect:*** *submissions by groups of more than two persons will not be graded.*
• *It is your responsibility to ensure that the PDF files you submit are legible. To this end, I encourage you to learn and use the LaTex typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You are not required to produce the PDF files you submit using LaTex; you may produce it any way you wish, as long as the resulting document is legible.*
• *By virtue of submitting this assignment you (and your partner, if you have one) acknowledge that you are aware of the policy on homework collaboration for this course.*[a]
• *For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class, or is in the assigned sections of the official course textbooks, by referring to it.*
• *Unless we explicitly state otherwise, you may describe algorithms in high-level pseudocode or point-form English, whichever leads to a clearer and simpler description. Do not provide executable code.*
• *Unless we explicitly state otherwise, you should justify your answers. Your paper will be graded based on* ***the correctness and efficiency of your answers, and the clarity, precision, and conciseness of your presentation****.*

[a] "In each homework assignment you may collaborate with at most one other student who is currently taking CSCC73. If you collaborate with another student on an assignment, you and your partner must submit only one copy of your solution, with both of your names. The solution will be graded in the usual way and both partners will receive the same mark. Collaboration involving more than two students is not allowed. **For help with your homework you may consult only the instructor, TAs, your homework partner (if you have one), your textbook, and your class notes. You may not consult any other source.**"

Question 1. (25 marks) You are given an array $A[1..n, 1..n]$ of ***distinct*** integers, where n is a positive integer; for simplicity assume that n is a power of 2. A ***local minimum*** of this array is an element that is smaller than all its neighbours, where the neighbours of element $A[i, j]$ are those of the elements above it, below it, to its left, and to its right that exist; in other words, those of elements $A[i-1, j]$, $A[i+1, j]$, $A[i, j-1]$, and $A[i, j+1]$ that have indices in the range $[1..n]$. (Thus, if $n = 1$, the unique element of A has no neighbours and is therefore trivially a local minimum; and if $n > 1$, the four "corners" of the array have two neighbours each, the elements in rows 1 and n and in columns 1 and n other than the corner elements have three neighbours each, and all other elements have four neighbours each.) The minimum element of the array is obviously a local minimum, but the array may have other elements that are local minima.

Give a divide-and-conquer $O(n)$ algorithm to find a local minimum in A. Justify the correctness of your algorithm and analyze its running time. Note that the size of your input is $\Theta(n^2)$ so you don't have enough time to look at all elements of the array. Finding the minimum element of A requires $\Omega(n^2)$ time, but a ***local*** minimum can be found faster.

Hint: In $O(n)$ time narrow down the search of a local minimum of the $n \times n$ array into the search of a local minimum of an $n/2 \times n/2$ array.

Question 2. (10 marks) Describe an algorithm that takes as input an array $A[1..n]$ of (not necessarily distinct) integers. Your algorithm returns an integer x that appears in more than $n/5$ positions of A (i.e., $|\{i : 1 \le i \le n \text{ and } A[i] = x\}| > n/5$), if such an x exists; and returns the special value NIL if no such x exists. Note that it is possible for several such elements to exist, in which case we can return any one of them. Your algorithm should run in $O(n)$ time in the worst case. (It is trivial to solve the problem in $O(n \log n)$ time by sorting A.)

Explain why your algorithm is correct, and analyze its running time. Note that the integers in A may be enormous, and so you cannot use a direct-address table.