Assignment 1

September 5, 2022

Deadline:-September 12, 2022 5:30 PM IST

1 Coding

For all the following questions devise the most efficient possible algorithm.

1. Write a quicksort algorithm with the following modification. The pivot selection happens using a pseudorandom number generator (you can use any library). You must also track the number and size of the size-problems which are being created. If the input size is n, you are constrained to have atmost $\lceil 2 \lg n \rceil$ subproblems. If the number of subproblems goes beyond this threshold, you may use modifications like

   - Merge few adjacent problems to make them into one subproblem, thereby reducing the number of subproblems, or
   - Use heap-sort on the smallest sub-problem in the set of all subproblems, thereby reducing the number of quicksort subproblems by one.

   Or you may also avoid such a scenario by always using a pivot which splits the problem into two subproblems of roughly equal size. If you find a pivot which does not fit this description, you discard that pivot and find a new pivot.

   Any alternate way to avoid beating this threshold can also be accepted as a viable solution, as long as it is theoretically correct.

   Input:- n on the first line

   Followed by n elements on the next line

   Output:- At each stage, print the number of subproblems.

   At the end, print the sorted output, along with number of times, the threshold was reached.

2. Store a two dimensional 15×20 integer array. The 2D array is organized in such a way that elements in each row are strictly increasing order and

elements in each column are also in strictly increasing order. Prompt the user for an integer (say x) input and return the location of x in the 2D array, if it is present otherwise return nil.

Devise the most efficient possible algorithm. No additional data structures can be used except the 2D array.

You may pre-fill the 2D array with distinct integers in the range of 1 to 1000.

Input:- an integer x.

Output:- location in array where x is present. If x is not present in the array return -1.

Hint:- If you are thinking of applying binary search, you are most likely wrong.

3. Dr Danny Denzongpa likes the idea that one can inspect the integers of a number in base 10 and figure out whether it is divisible by 2(11) or not.

   So he came up with this unusual algorithm, which he claims to be correctly finding the GCD everytime.

   The algorithm is as follows.

   Suppose the algorithm is finding GCD of two integers a and b. $x|y$ denotes that x is divisible by y with zero remainder.

   If $a|11$ and $b|11$, then GCD(a, b)= 11*GCD$(a/11, b/11)$.

   Otherwise, if $a|11$, $a \nmid 2$ and $b \nmid 11$ then GCD(a, b)= GCD$(a/11, 2 * b)$.

   Otherwise, if $a|11$, $a|2$ and $b \nmid 11$ then GCD(a, b)= GCD$(a/11, |a - b|)$.

   Otherwise, if $b|11$, $b \nmid 2$ and $a \nmid 11$ then GCD(a, b)= GCD$(2 * a, b/11)$.

   Otherwise, if $b|11$, $b|2$ and $a \nmid 11$ then GCD(a, b)= GCD$(|a - b|, b/11)$.

   Otherwise, GCD(a, b)= GCD$(|a - b|, min(a, b))$.

   You are also given a base case that GCD$(a, a) = a$.

   If the algorithm is correct, prove its correctness and

   If the algorithm is wrong, give a counter-example to prove its incorrectness.

   Either way, write the code to capture the above mentioned idea in the following format.

   Input:- two integers a and b.

   Output:- GCD(a,b) which maybe wrong if the aforementioned algorithm is wrong.

Hint:- First figure out, if this process always terminates. Give a counter-example if the process does not terminate.

If it terminates, find out how many steps does it need to terminate and figure out its correctness. If it terminates with incorrect answer, then find a pair of integers for which the output is incorrect.

Each question is worth 2 marks, for a total of 6 marks.