Laboration 2

Due 2025-12-02

For each algorithm designed, you should

- give a complete and unambiguous high-level description (step-wise) of your algorithm in plain English/Swedish; and
- implement your algorithm (using Python) as *one* recursive function. Built-in functions or methods for strings or lists must not be used.
- No experimental analysis of the algorithm is needed.

1. Given an array A of n ($n \geq 4$) distinct real numbers, the problem is to determine whether or not A contains three distinct elements x, y, and z such that $x + y = z$. You may assume that $n = 2^k$ for some positive integer $k \geq 2$.

   - Design a worst-case $\Theta(n^3)$-time recursive algorithm using an incremental approach to solve the above problem.
   - Design a worst-case $\Theta(n^2)$-time recursive algorithm using an incremental approach to solve the above problem.

2. Given an array $A = \langle a_1, a_2, \cdots, a_n \rangle$ of non-zero real numbers, the problem is to find a subarray $\langle a_i, a_{i+1}, \cdots, a_j \rangle$ (of consecutive elements) such that the sum of all the numbers in this subarray is maximum over all possible consecutive subarrays. Design a divide and conquer algorithm to compute such a maximum sum. You do not need to actually output such a subarray; only returning the maximum sum. Your algorithm should run in $O(n)$ time in the worst case. You may assume that n is a power of 2.

   - Give a complete and unambiguous high-level description (step-wise) of your algorithm in plain English/Swedish; and
   - Implement your algorithm (using Python) as *one* recursive function. Built-in functions or methods for strings or lists must not be used.

3. Given an array $A = \langle a_1, a_2, \cdots, a_n \rangle$ of n elements, consider the following comparison-based sorting algorithm:

   def *sortR*($\langle a_1, a_2, \cdots, a_n \rangle$):

   0) If $n \leq 4$, then sort the input with insertion sort and return the sorted array.
   1) Sort $< a_1, \cdots, a_{\frac{3n}{4}} >$ recursively and let $< b_1, \cdots, b_{\frac{3n}{4}} >$ be the sorted output.
   2) Sort $< b_{\frac{n}{4}+1}, \cdots, b_{\frac{3n}{4}},\ a_{\frac{3n}{4}+1}, \cdots, a_n >$ recursively and let $< c_{\frac{n}{4}+1}, \cdots, c_n >$ be the sorted output.
   3) Sort $< b_1, \cdots, b_{\frac{n}{4}},\ c_{\frac{n}{4}+1}, \cdots, c_{\frac{3n}{4}} >$ recursively and let $< d_1, \cdots, d_{\frac{3n}{4}} >$ be the sorted output.
   4) return $< d_1, \cdots, d_{\frac{3n}{4}},\ c_{\frac{3n}{4}+1}, \cdots, c_n >$

   You may assume that all the problem sizes during recursions are integers.

   - Show that the above algorithm is correct. *Hint: One can use the induction technique to prove the correctness of recursive algorithms.*
   - Derive a *recurrence (equation)* that describes the worst-case running time of this algorithm.
   - Solve the *recurrence* obtained. *Hint: One can use the master method for solving recurrence equations (Theorem 4.1 Master theorem in the textbook).*

Report

Each group submits **one** report. The report can be written in either Swedish or English and should not be handwritten. *Before submitting your report, you should discuss your solution to the laboration (design, implementation, and report) with your lab-assistant.*