Assignment 4

due by 9:30 am on April 7

To permit the prompt distribution of solutions and return of marked assignments, late assignments will not be accepted. Include your name, student number, and email address at the top of the first page on all submitted material, as well as the names of people with whom you have discussed your assignment solution. Cite any sources to which you refer, as you should do when presenting any scientific document. You must submit your solution electronically using UMLearn. Only pdf files will be accepted.

1. This question considers multidimensional range counting and range sum queries over a binary universe. 

   **Range Sum Query on an Array.** In lecture we discussed range sum queries on an array. The problem is to preprocess a given integer array \(A[0 : n-1]\) to efficiently answer subsequent range sum queries, i.e., to return the sum of all values in \(A[i : j]\) for arbitrary indices \(i\) and \(j\) given a query time, where \(0 \leq i \leq j \leq n-1\).

   **Bit Arrays and Range Counting Query.** Suppose the universe is restricted to \(\{0, 1\}\). In this case, a range sum query corresponds to returning the number of non-zero bits in the query range. This is also known as a range counting query (i.e., count the number of non-zero bits).

   **Range Counting on a Two-Dimensional Bit Array.** Suppose we have an \(O(nm)\)-size two-dimensional bit array \(A[0 : n-1][0 : m-1]\) and a query range corresponds to an arbitrary subarray \(A[i_x : j_x][i_y : j_y]\); i.e., count the number of non-zero bits in \(A[i_x : j_x][i_y : j_y]\) for arbitrary \(i_x, j_x, i_y, j_y\), and \(j_y\) given a query time, where \(0 \leq i_x \leq j_x \leq n-1\) and \(0 \leq i_y \leq j_y \leq m-1\).

   **Two-Dimensional Dominance Queries.** If at least one of \(i_x = 0\) or \(j_x = n-1\) and at least one of \(i_y = 0\) or \(j_y = m-1\), then the corresponding range counting query is known as a range dominance query. This is because in every dimension, the query range is bounded on one side and unbounded in the other. See Figure 1.

![Figure 1](image-url)

Figure 1: **Left:** This range counting query returns 2. **Right:** This range dominance query returns 6.

Given an arbitrary \(n \times m\) binary array \(A\), describe a static data structure that supports efficient range counting queries on \(A\) and uses no more than \(O(nm)\) space. Describe the corresponding range counting algorithm and its running time.

**Hint:** How can you use a constant number of range dominance queries to answer a range counting query?
2. The following algorithm takes an array \( A[0 : n - 1] \) of integers as input and constructs a data structure.

1. \( \text{root} \leftarrow \text{new BinaryTreeNode} \)
2. \( \text{root.key} \leftarrow \infty \)
3. \( \text{st} \leftarrow \text{new Stack} \)
4. \( \text{st.push(root)} \)
5. \( \text{for } i = 0 \text{ to } n - 1 \)
6. \( x \leftarrow \text{new BinaryTreeNode} \)
7. \( x.key \leftarrow A[i] \)
8. \( \text{if } A[i] \leq \text{st.top.key} \)
9. \( \text{st.top.right} \leftarrow x \)
10. \( \text{else} \)
11. \( \text{while } A[i] > \text{st.top.key} \)
12. \( \text{st.pop} \)
13. \( y \leftarrow \text{st.top} \)
14. \( x.left \leftarrow y.right \)
15. \( y.right \leftarrow x \)
16. \( \text{st.push}(x) \)

(a) Describe the data structure constructed. A few sentences should suffice.

(b) What is the key of the right child of the root at the completion of the \( i \)th iteration of the for loop?

(c) The code does not check whether the stack is empty before calling top or pop. Can this result in an error (empty stack exception)? Explain briefly.

(d) Assuming each push or pop operation takes \( O(1) \) time, what is the algorithm’s worst-case running time expressed in terms of \( n \) using asymptotic (\( \Theta \) or \( O \)) notation? Justify your answer.

(e) What is the output of a call to \textit{someFunction} after having constructed the data structure?

\textit{someFunction}
1. \textit{myFunction(root.right)}

\textit{myFunction(BinaryTreeNode t)}
1. \text{if } t \neq \emptyset
2. \text{myFunction(t.left)}
3. \text{print t.key}
4. \text{myFunction(t.right)}