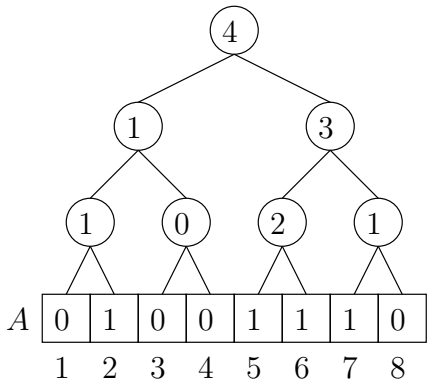
Homework Assignment #5
Due: November 4, 2022 at 11:59 p.m.

1. In class, we described a data structure to store a set that consisted of an array A of bits with a tree structure built on top of it. Elements of the set come from the domain $D = \{1, 2, \ldots, N\}$. For simplicity, assume throughout this question that N is a power of 2.

The entries of the array are the leaves of the tree. Each internal node x in the tree stores the number of non-zero bits in the leaves of the subtree rooted at x. For example, the structure that represents the set $\{2, 5, 6, 7\}$ is shown below. This data structure supports insertions, deletions, and a variety of queries in $O(\log N)$ time each.



Since this tree has a fixed size and shape, it is nice to implement it using a single array $B[1..2N-1]$ in the same way that we represent a binary heap (see Chapter 6.1 of the textbook, including Figure 6.1). Each entry of B stores a node of the tree, ordered as they would be in a breadth-first search of the tree. $B[1]$ is the root. The left and right children of $B[i]$ are $B[2i]$ and $B[2i+1]$. The parent of $B[i]$ is $B[\lfloor i/2 \rfloor]$. The value of $A[i]$ appears in $B[i+N-1]$. The tree above would be represented as follows.

B	4	1	3	1	0	2	1	0	1	0	0	1	1	1	0
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Now consider the following problem. We wish to store a *multiset* of elements drawn from the domain $D = \{1, 2, \ldots, N\}$ using a similar kind of data structure. Recall that a multiset is like a set, except that we can have more than one copy of the same element. Like sets, the order of elements in the multiset is not important. Assume the multiset is initially empty. We want to support the following operations (where $1 \le i \le j \le N$).

- INSERT(i) adds a copy of the element i to the multiset.
- COUNT(i) returns the number of copies of i that are currently in the multiset.
- COUNTRANGE(i, j) returns the total number of elements whose values are between i and j (inclusive) in the multiset.

For example, if the multiset is $\{3, 1, 7, 1, 3, 2, 5, 7, 7\}$ then COUNT(7) would return 3 since there are 3 copies of 7 in the multiset and COUNTRANGE(3, 7) would return 6 since there are 6 elements whose values are between 3 and 7 (namely, $3, 7, 3, 5, 7, 7$). An INSERT(1) applied to this multiset would change it to $\{3, 1, 7, 1, 3, 2, 5, 7, 7, 1\}$.

Your data structure should implement the three operations efficiently. The space usage should be $O(N)$ words of memory. Assume the number of elements in the multiset fits in a single word and that arithmetic on words of memory can be done in constant time.

[2] **(a)** Briefly describe you represent the multiset in your data structure.

[2] **(b)** Draw a picture of your data structure for the multiset $\{3, 1, 7, 1, 3, 2, 5, 7, 7\}$, assuming $N = 8$.

[6] **(c)** Write pseudocode for each of the three operations.

Hint: to implement CountRange, you might want to design a helper function similar to the Rank query that was discussed in class.

[3] **(d)** What is the worst-case running time of each of your operations? Use Θ notation to express your answer in terms of N and/or n, where n is the number of elements in the data structure.