Exercises on implementing data structures

1. Consider a sequence of n bits $b_0, b_1, \ldots, b_{n-1}$. Assume that initially all bits are false. The operations to be performed on the sequence are

```
a) set(i) : set ith bit to true.
b) reset(i) : set ith bit to false.
c) complement(i) : complement ith bit.
c) get(i) : get the value of ith bit.
d) setall() : set all n bits to true.
e) resetall() : reset all bits to false.
f) complementall() : complement all n bits.
```

   Describe a data structure that can do all operations in $O(1)$ time, independent of n. It may be easier to first come up with a data structure that takes $O(m)$ time for any sequence of m operations. You can take $O(n)$ time for initialization.

2. Consider a sequence $a_0, a_1, \ldots, a_{n-1}$ of integers. The sequence is said to be uniform if all elements in the sequence have the same value. A data structure is required that supports the following operations.

```
a) set(i,x) : set the value of a_i to x.
b) get(i)   : return the value of a_i.
c) is_uniform() : return true iff the current sequence is uniform.
```

   Describe a data structure that can perform all 3 operations in $O(1)$ time. Suppose a sequence is near uniform if the elements take at most two distinct values. Show how an operation `near_uniform` can be implemented in $O(1)$ time. Can this be generalized to at most k values for any fixed constant k? Again initialization can take $O(n)$ time.

3. A stack is a vector in which elements can only be added or deleted from the end, and only the last element can be accessed. Another operation `max` is to be implemented along with other stack operations, that returns the maximum value in the stack. Show how to implement all operations in $O(1)$ time. Assuming the stack contains integers, can you implement it so that only $O(1)$ additional memory, apart from that required for storing the sequence, is used. A queue is a sequence in which elements are added to the end but deleted from the front. Show how to implement the `max` operation for a queue, along with the other operations, so that any sequence of m operations takes $O(m)$ time. Can all operations be done in $O(1)$ time?

4. Suppose you have n counters $c_0, c_1, \ldots, c_{n-1}$, all initially 0. The following operations are to be performed on these counters.

```
a) increment(i) : increment the ith counter.
b) decrement(i) : decrement the ith counter.
c) findmax(i) : return the smallest counter number with maximum value.
```

Show how you can implement all operations is $O(1)$ time.

5. Consider a data structure that represents a non-negative integer with arbitrary number of bits. The initial value is 0. The only operations to be performed are incrementing the integer and printing its value. Show how the increment operation can be performed in $O(1)$ time, while printing can take time proportional to the number of bits in the value printed.