1. Consider the “dynamic range minimum query problem.” Let $S \subseteq U = \{0, 1, 2, \ldots, u - 1\}$. Each $s \in S$ is labeled with a real number $v(s)$. We need to maintain a data structure on $S$ that supports the following operations:

- **initialize**($S$): Construct and initialize the data structure for $S$.
- **decreaseKey**($s, x$): If $x < v(s)$, change $v(s)$ to $x$; otherwise, do nothing.
- **minimum**($x$): Return $\min\{v(s) | s \in S, s \leq x\}$.

Show how a vEB tree can be used to support these three operations. Analyze the time/space complexity of your algorithms.

2. PhD Qualifying Exam Section Problem 8.

   In practical implementations of vEDB trees, especially on machines with “shift-by-$k$” and “find first zero” instructions, performance can be improved by switching to a bit array once the size of a cluster is nearly equal to the word size, say a small multiple of it, is reached. All operations on a single word (or a few words) are constant time.

   (a) Show that this does not affect the asymptotic performance of vEDB trees.

   (b) Explain carefully why this idea avoids the majority of the pointer storage and pointer dereferences, achieving a significant practical savings in time and space. Estimate the amount of the savings for each of the vEDB tree functions given in CLRS.