1. van Emde Boas trees [2012 Paper 3 Question 1]
   (a) Draw a proto-vEB node, labelling each of its fields and briefly explaining what each field does.
   (b) Draw a vEB node, labelling each of its fields and briefly explaining what each field does.
   (c) On its own page, draw in pencil a complete and legible vEB tree holding the values \{0,2,4,8,9,10,13,14\}. The correctness of the structure and the accuracy of all fields of all nodes is important. Once done, write each of the values in ink under the leaf record in which it is logically stored.
   (d) Consider the task of inserting a value \(v\) into a proto-vEB or vEB tree whose root node is \(r\). Assume the value is in range and not already in the tree. Write two legible pieces of high-level pseudocode for \(\text{insertInProtoVEB}(r, v)\) and \(\text{insertInVEB}(r, v)\) respectively. Clarity (insert comments where appropriate) will count more than perfect low-level accuracy. Derive the computational complexity of your two procedures using the appropriate recurrence formulae (but solving the recurrences is not required). Explain what specific features of the vEB tree make it faster than the proto-vEB tree for this particular task.

2. Amortized and aggregate analysis [2009 Paper 4 Question 1]
   (a) Explain the terms amortized analysis and aggregate analysis, highlighting the difference between them.
   (b) Assume that the following two classes, which implement the standard stack (last-in first-out) and queue (first-in first-out) data structures, are available.

```java
class stack
    void push(Item x)
    Item pop()
    Boolean isEmpty()

class Queue
    void enqueue(Item x)
    Item dequeue()
    Boolean isEmpty()
```
(i) A Multistack class is derived from Stack with the addition of two methods:
   - a \texttt{void multipush(Itemlist l)} that takes a list \(l\) of items and pushes each of the items onto the stack (each action of extracting an item from the list and pushing it onto the stack has constant cost), and
   - a \texttt{void multipop(int m)} that takes an integer \(m\) and pops that many items off the stack (raising an exception if there were fewer, but don’t worry about that).
Is it true or false that, given an arbitrary sequence of \( n \) \texttt{Multistack} operations starting from an empty \texttt{Multistack}, each operation in the sequence has amortized constant cost? Justify your answer in detail.

(ii) Provide an implementation of \texttt{class Queue} using no other data structures than \texttt{Item}, \texttt{Boolean}, \texttt{int} and \texttt{Stack}. The amortized running time of each \texttt{Queue} method must be constant. (Note that you may only use the \texttt{Stack} as a black box: you are not allowed to access its internal implementation.)

(ii) Using the potential method, prove that the amortized running time of all your \texttt{Queue} methods from part (b)(ii) is indeed constant.

3. Linear cost
   Why do we claim that keeping the sorted-array priority queue sorted using bubble sort has linear costs? Wasn’t bubble sort quadratic?

4. proto-vEB tree
   Sketch a picture of a proto-vEB tree of size \( u = 16 \) representing the set \( 2, 3, 4, 5, 7, 14, 15 \).

5. Recurrence relations
   (a) Prove that the recurrence \( T(u) = 2T(\sqrt{u}) + O(1) \) has the solution \( T(u) = O(\log u) \).
   (b) Prove that the recurrence \( T(u) = 2T(\sqrt{u}) + O(\log \sqrt{u}) \) has the solution \( T(u) = O(\log u \log \log u) \).

6. vEB trees
   Sketch a picture of a vEB tree of size \( u = 16 \) representing the set \( 2, 3, 4, 5, 7, 14, 15 \). OK to study the textbook and handout first, but keep them closed while doing this exercise. No matter how good you are, you will almost certainly get some details wrong. That’s OK. Check the textbook after having drawn your solution and mark in red all the items that are different, understanding why. Then the next day do the exercise again, with textbook closed. Iterate until you get no errors. Will take several days (no shame in that). You may think you understand vEB trees but you actually don’t until you successfully complete this exercise.