1. Give an example of a situation that motivates amortized analysis, and argue for its usefulness.

2. If the set of stack operations included a MULTI_PUSH operation, which pushes \( k \) items onto the stack, would the \( O(1) \) amortized cost of stack operations continue to hold?

3. Suppose we perform a sequence of stack operations on a stack whose size never exceeds \( k \). After every \( k \) operations, we make a copy of the entire stack for backup purposes. Show that the cost of \( n \) stack operations, including copying the stack, is \( O(n) \) by assigning suitable amortized costs to the various stack operations.

4. Suppose we perform a sequence of \( n \) operations on a data structure in which the \( i \)th operations costs \( i \) if it is an exact power of 2, and 1 otherwise. Use aggregate analysis to determine the amortized cost per operation.