CS 466/666 Fall 2018 due Tuesday September 25, 5 PM
A. Lubiw

ASSIGNMENT 1

DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

1. [5 marks] Suppose we wish to maintain an array of elements subject to the following two operations:

   add(e) – add element e to the next available space in the array. The cost is 1 time unit.

   empty – empty the array by removing all its elements. The cost is $k+1$ time units if the array has k elements in it.

   The goal of this question is to use the potential method to prove that the amortized cost of each operation is $O(1)$ in any sequence of operations starting with an empty array.

   Define the *charge* of the *add* operation to be 2. Define the *charge* of the *empty* operation to be 1. Recall that the definition of the potential after the i^{th} operation is $\Phi_i = \Phi_{i-1} + \mathit{charge}(i) - \mathit{cost}(i)$, with the initial potential $\Phi_0 = 0$.

   (a) [3 marks] What is the potential when the array has k elements?

   (b) [1 mark] Prove that the final potential is $\geq$ initial potential.

   (c) [1 marks] Conclude that the amortized cost of each operation is $O(1)$. (You can quote the relevant theorem from class, you do not need to reprove it.)

   Note that the point of this exercise is to use the potential method. Do not answer the question using a diffferent method.

2. [10 marks] Suppose you want to support Search and Insert on a set of elements from a totally ordered universe. The set will be stored as a collection of sorted arrays. Specifically, to store n elements, let $b_k, \ldots, b_0$ be the binary representation of n, and use a collection of sorted arrays, $A_k, \ldots, A_0$ where A_i is empty if b_i is 0, and A_i has 2^i elements otherwise. It doesn't matter which elements go in which A_i's.

   (a) [3 marks] Give a Search algorithm and analyze its worst case run time.

   (b) [7 marks] Give an Insert algorithm and analyze its worst case and amortized run times. You may use results from class on amortized analysis of a binary counter.

   (It goes without saying that you should aim for efficient algorithms and for tight analysis.)