The reading for this week is Chapter 13, pages 273–287.

The first exam will be a takehome exam given out in class on Thursday, February 27 and will be due back in class on Tuesday, March 4. The exam will be closed-book and closed-notes, although you may prepare your own 8.5 x 11 sheet (both sides) with any information on it that you like. The exam will be designed to take approximately 75 minutes, but you will be given 2 hours to take it in the safety of your own home.

1. **[20 Points] Extendible Arrays Revisited.** In class we examined extendible arrays. We showed that by doubling the length of the array each time we need to perform an extension, a sequence of $n$ insertions takes time $O(n)$. Now imagine that we allow deletions in addition to insertions. Just as an insertion always inserted an element at the end of the array, a deletion always deletes the last element in the array. You can imagine that an array may become quite large but later, due to deletions, most of the array becomes empty. In this kind of scenario, it would be nice to contract the size of the array to give the memory back to the operating system.

   Professor I. Lai of the Pasadena Institute of Technology has suggested the following rule: Use the regular doubling rule to extend arrays. However, when an array becomes less than half full (because of deletions, presumably), allocate a new array of half the length of the current array and copy the elements into the new array. (At that point the old array is released and its memory is recovered by the operating system.)

   (a) Play the role of the “malicious adversary” and describe a sequence of $n$ insert and delete operations for which Professor Lai’s rule would incur a cost of $\Theta(n^2)$. Explain your analysis.

   (b) Professor Lai was not granted tenure and he was replaced by Professor Anna Litik. Professor Litik had a much better idea: Use the regular doubling rule to extend arrays. However, only contract an array when it becomes less than or equal to $1/4$ full. At that point, allocate a new array of half the length of the current array and copy the elements into the new array. Use an amortization argument to explain why under this scheme any sequence of $n$ inserts and deletes costs a total of $O(n)$ time. Still use 3 rubles for the insertions but now explain how much the deletions should pay and why this all works! (Recall that allocating a new block of memory can be done in constant time. However, copying one element from one array to another takes constant time. Thus, copying $k$ elements takes $k$ time.)

   (c) Professor Litik’s colleague, Professor Polly Nomial, has proposed another variant of extendible arrays: Still just double the array when it becomes full. However, we contract the array (due to deletions, presumably) only when it becomes less
than or equal to 1/3 full. At that point, allocate a new array of 2/3 the length of the current array and copy the elements into that new array. Does a sequence of $n$ inserts and deletes still cost a total of $O(n)$ time in this scheme? If so, give a clear amortization argument to explain why. If not, describe a sequence of $n$ inserts and deletes which would cause this scheme to use more than $O(n)$ time.