1. Assume that you have a special routine that, given an array of size n, uses no comparisons to return the index of a random element whose value is in the middle third of the array. So the value of this element is between the $n/3+1$th and $2n/3$rd smallest, inclusive. Assume that you execute quicksort with the help of this routine.
   (a) Estimate the number of comparisons that quicksort now uses assuming that the $5n/12$th smallest element is always picked by this routine. Just get the high order term.
   (b) Find the high order term for the number of comparisons that quicksort now uses. To simplify the computation, you may assume that the size of a sublist is always a multiple of 3, and you may modify your summation bounds a small constant amount, but state when and how you do so.
   (c) Compare your estimate from Part (a) with the actual value for the high order term from Part (b).

2. Assume that you have a routine, `random(a,b)`, that inputs two integers a, b and returns a uninformly distributed random integer between a and b, inclusive.
   (a) Write a routine that inputs an array and randomly permutes the elements (so that each permutation is equally likely). You may look this up if you like (but try it yourself first).
   (b) Exactly how many moves and/or exchanges does your algorithm use (in the worst case)?