Before you start, ***write your name at the top of each page****. Enough space should be given for each solution, but if not then indicate this and continue on the back.*

I suggest that you read the entire exam before you start. If you find a problem with the exam, please note it in your answer and answer as best you can. Please show as much of your work as you reasonably can.

1. Consider the following algorithm. Is it guaranteed to terminate? If so, what is its asymptotic big-O worst-case running time on an array of n elements?

   BAD ALL-MIN

   To set all elements of an array a of n integers to the minimum value:
   do
       $changed \leftarrow$ **false**
       $m \leftarrow a[n]$
       for i **in** $1..n$
           $m \leftarrow \mathbf{min}(a[i], m)$
           if $m < a[i]$
               $a[i] \leftarrow m$
               $changed \leftarrow$ **true**
   while $changed$

   The algorithm will always terminate, and will run in time $O(n)$. After the first pass over the array $a[n]$ will contain a minimum element. After the second pass, all the elements will be minimum. After the third pass, nothing will change.

2. Suppose that you want to extract the item at position i of a Heap of n items (the i^{th} element of the array) rather than extracting a best value?

   (a) Give pseudocode for the extraction in terms of UpHeap and DownHeap.

   $e \leftarrow H[i]$
   $H[i] \leftarrow H[n]$
   shrink H **by** 1
   if $H[i] < H[\text{parent}(i)]$
       upheap $H[i]$
   else if $H[i] > H[\text{left}(i)]$ or $H[i] > H[\text{right}(i)]$
       downheap $H[i]$
   return e

   (b) What is the worst-case big-O asymptotic complexity of your algorithm? Justify your answer.

   $O(\lg n)$ since in the worst case the heap is traversed from bottom-to-top or top-to-bottom..

3. Consider the following (terribad—do not use) shuffling algorithm:

Bad Shuffle

To *shuffle* an array a of n elements:
    if $n \leq 1$
        return
    $m \leftarrow (n + 1)$ **div** 2
    $p \leftarrow (m + 1)$ **div** 2
    for i **from** 1 **to** m
        exchange $a[p + \mathrm{i}]$ **with** $a[p + random(1..m)]$
    shuffle $a[1..m]$
    shuffle $a[p..p{+}m]$
    shuffle $a[m..n]$

Assuming constant-time *random*, exchanges and arithmetic, what is the asymptotic big-O worst-case complexity of this algorithm? Justify your answer.

$O(n^{\log_3 2})$. This is Case 3 of the Master Theorem as handed out with the exam, since $a = 3$, $b = 2$ and $c = 1$.