Problem 1. In a min-heap every node $i$, other than the root, holds the property, $A[parent(i)] \leq A[i]$, such that, the smallest element in a min-heap is at the root. Suppose we have a min-heap containing $n = 2^j$ elements in an array of size $n$, and suppose that we repeatedly extract the minimum element, $n$ times. To make the heap space efficient, we move the heap over to an array of size $2^j$ whenever an extraction decreases the number of elements to $2^j$ for any integer $j$. Suppose the cost of each such move is $2^j$. What is the total movement cost caused by $n$ extract-mins starting from the heap of $n$ elements? (You may ignore the $\theta(n\lg n)$ cost from the heapify operations themselves.)

Problem 2. Let $A$ be an arbitrary (un-sorted) array of distinct (very large) positive integers. Describe a strategy in high level pseudo-code or structured English for an efficient algorithm that finds the two largest and the two smallest numbers in $A$ with an exact worst-case number of comparisons to be, $\frac{3n}{2} + 2\lg n - 4$. Show the comparison for each step, you present.

Problem 3. Suppose there is an array of length $n$. It contains values in the range of 1, $\ldots$, $n+1$. However, exactly one value out of $\{1, \ldots, n+1\}$ is missing from this array. Find this missing number as efficiently as possible in the following two cases:

(a) The numbers in the array are stored in a random order.
(b) The array is sorted such that the value stored at index 1 < value at index 2 < $\cdots$ < value at index $n$

Write pseudocode to find the missing number for the two cases. Analyze the algorithms exactly.