CISC320 algorithms, 10S, midterm exam question illustrations

```
procedure Stooge-sort
---------------------
Input: array A[0..n-1] of n numbers
Output: A is sorted in increasing order.

If n = 2 and A[0] > A[1], then swap (A[0], A[1])
If n > 2 then
Stooge-sort(A[0..ceil(2n/3)]) // sort first two-thirds.
Stooge-sort(A[floor(n/3)..n]) // sort last two-thirds.
Stooge-sort(A[0..ceil(2n/3)]) // sort first two-thirds again.
```

1. Let T(n) denote the worst case number of comparisons (A[0] ¿ A[1]) made for an input array of n numbers. Give a recurrence relation for T(n).
2. State the case of the Master theorem that applies to this recurrence and give the values of a, b, and d of the theorem that apply here.
3. Solve the recurrence. That is, give a big-O upper bound for T(n) that results from the Master theorem.
4. Is stooge-sort correct? Would you be inclined to use stooge sort in an application?

Questions like 1, 2, 3, but about either an algorithm we have discussed or a new one stated on the exam (but simpler than stooge-sort) may be expected.
Questions like #4 will not be on the exam.
Some questions may be in multiple choice format.

1. RSA public key encryption requires secret pre-determination of 2 large primes. How is primality tested? [questions to explain a specific algorithm we have studied]
2. About how many comparisons are necessary to find the max and min in an array of n numbers? [questions asking you to recall an analysis we have done.]
3. What makes the RSA public key encryption/decryption scheme hard to break? [questions probing the main idea of an algorithm or system]
4. (a) write a function to merge two sequences into a third. (b) write merge sort. [questions to produce a specific algorithm we have studied]

Vis a vis chapter 1, the main point is that modular arithmetic and primality testing are easy (how easy?) but factoring is hard.

Vis a vis chapter 2, the main point is that divide and conquer is a powerful strategy. The master theorem tells how to solve most divide and conquer problems. Memorize and understand the master theorem. Know which part of it applies to each divide and conquer problem.

Vis a vis chapter 3, the main point is that the times of previsits and postvisits in depth first search can solve a number of graph problems (classification of edges, existence or not of a cycle, linearization of DAGs, strongly connected components, ...others to come later).

Vis a vis chapter 4, Dijkstra's single source shortest path algorithm.