Problem 1. Explain how to implement the operation \( x \mod y \) in \( O(1) \) time where \( x \) and \( y \) are positive integers.

Problem 2. For the \( k \)-selection problem, suppose that the input is an array of 12 elements: \( (58, 23, 98, 83, 32, 24, 18, 45, 85, 91, 2, 34) \). Recall that our randomized algorithm first selects a number \( v \) and then recursively solves a subproblem. Suppose that \( v = 34 \) and \( k = 10 \). What is the size of the array for the subproblem?

Problem 3 (Textbook Exercise 9.3-5). The median of a set \( S \) of \( n \) elements is the \( \lfloor n/2 \rfloor \) smallest element in \( S \). Suppose that you are given a deterministic algorithm for finding the median of \( S \) (stored in an array) in \( O(n) \) worst-case time. Using this algorithm as a black box, design another deterministic algorithm for solving the \( k \)-selection problem (for any \( k \in [1, n] \)) in \( O(n) \) worst-case time.

Problem 4. Let \( S \) be a set of \( n \) integers, and \( k_1, k_2 \) be arbitrary integers satisfying \( 1 \leq k_1 \leq k_2 \leq n \). Suppose that \( S \) is given in an array. Give an \( O(n) \) expected time algorithm to report all the integers whose ranks in \( S \) are in the range \( [k_1, k_2] \). Recall that the rank of an integer \( v \) in \( S \) equals the number of integers in \( S \) that are at most \( v \).

Problem 5* (Textbook Exercise 9-2). We are given an array that stores a set \( S \) of \( n \) distinct integers. Set \( W = \sum_{e \in S} e \). Describe an algorithm to find the element \( e^* \in S \) that makes both of the following hold:

- \( \sum_{e < e^*} e < W/2 \)
- \( \sum_{e > e^*} e \leq W/2 \).

Your algorithm should finish in \( O(n) \) time (\( O(n) \) expected time is acceptable).

(Hint: First convince yourself that such \( e^* \) is unique, and then adapt the \( k \)-selection algorithm).