Problem 1.

(a) What is the best-case number of moves in Insertion Sort? Justify.
(b) What is the worst-case number of moves in Insertion Sort? Justify.
(c) What is the average-case number of moves in Insertion Sort? Justify.

Problem 2.

(a) Is $2^{n+1} = O(2^n)$?
(b) Is $2^{2n} = O(2^n)$?

Problem 3. Use the iteration method to solve the following recurrences. You may assume $n$ is “nice”. Prove your answers using mathematical induction.

(a) $T(n) = 2T(n/2) + n^3$, $T(1) = 1$.
(b) $T(n) = T(\sqrt{n}) + 1$, $T(2) = 1$.
(c) $T(n) = 2T(n/2) + n \lg n$, $T(1) = 1$.
(d) $T(n) = T(n - 3) + 5$, $T(1) = 2$.

Problem 4. Which of the above problems can be solved using the “Master Theorem” derived in class. Solve them exactly using the “Master Theorem”.

Problem 5. Assume you have an array of numbers, where each value occurs at most twice. We consider sums of contiguous numbers in the array. But we only consider such sums whose two endpoints have the same value. The sum includes the two equal values themselves. So if the two equal numbers are at index $i$ and index $j$ ($i < j$) in array $A$, then we sum all the values $A[i], A[i + 1], \ldots, A[j]$.

(a) Give an algorithm that finds the maximum such sum. Make your algorithm as efficient as possible. Describe the algorithm briefly in English and in pseudo code.

(b) Analyze the running time of your algorithm.
Problem 6. Let $A[1, \ldots, n]$ be an array of $n$ numbers (some positive and some negative).

(a) Give an algorithm to find which two numbers have sum closest to zero. Make your algorithm as efficient as possible. Write it in pseudo-code.

(b) Analyze its running time.

Problem 7. Show that

(a) $$\frac{1}{2} \leq \sum_{j=1}^{\infty} \frac{1}{j2^j} \leq 1$$

(b) $$1 \leq \sum_{j=1}^{\infty} \frac{1}{j^2} \leq 2$$