There are 5 problems, accounting for 100% in total.

Problem 1 (20%)**. (Problem 1-1, Comparison of running times).** For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes $f(n)$ microseconds.

	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
$\log n$							
$\sqrt{n}$							
n							
$n \log n$							
n^2							
n^3							
2^n							
$n!$							

Problem 2 (20%)**.** Implement the BucketSort algorithm. Conduct an experiment on "Insertion-Sort", "Merge-Sort", and "Bucket-Sort" algorithms and record the running times of these algorithms for n randomly generated integers for different n. Use a table and a figure to summarize your result.

Problem 3 (20%)**.** Using Figure 2.4 (in the textbook) as a model, illustrate the operation of merge sort on the array $A = \{ 3, 41, 52, 26, 38, 57, 9, 49 \}$.

Problem 4 (20%)**.** Describe a $\Theta(n \log n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exists two elements in S whose sum is exactly x.

Problem 5 (20%)**.** Prove the following statements.

1. If $f(n) = O(g(n))$ and $g(n) = O(h(n))$, then $f(n) = O(h(n))$.
2. $f(n) = O(g(n))$ if and only if
$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = O(1).$$
3. $f(n) = o(g(n))$ if and only if $g(n) = \omega(f(n))$.
4. $f(n) = o(g(n))$ if and only if
$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = o(1).$$