Exercise 1: asymptotic behavior (20 points).

1. (10 points) Assume you have two computers, $C_A$ and $C_B$ capable of performing $10^6$ and $10^8$ operations per second, respectively. Both computers run a set of algorithms whose precise complexities $f(n)$ are given below. Determine the size $n^*$ of the biggest input that can be processed in 1 second for each computer, as in the example.

<table>
<thead>
<tr>
<th>$f(n)$</th>
<th>$n^*$ for $C_A$</th>
<th>$n^*$ for $C_B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sqrt{n}$</td>
<td>$10^{12}$</td>
<td></td>
</tr>
<tr>
<td>$14n + 4$</td>
<td>$10^{10}$</td>
<td></td>
</tr>
<tr>
<td>$2 + \lg (n^{10})$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$-\sqrt{n}\log_3 n + n$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10n\log_2 n - 1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$n!$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(2n)^{[n/5+1]}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The precise complexity tells you how many operations are performed to solve an instance of size $n$. Assume each operation takes the same time and that the input sizes are natural numbers $1, 2, 3, \ldots$

2. (10 points) Prove formally that $f(n) = an^3 + bn + c$ where $a > 0$ is $\Theta(n^3)$. *Hint:* find values for the constants $c_1, c_2, n_0$ in the definition of $\Theta(\cdot)$ and show it holds.

Exercise 2: running time (38 points). Consider the following function, where $A$ is an array containing integers, $u$ and $v$ are integers with $u < v$, and the initial call is $\text{Fcn-X}(A, u, v, 1, A.\text{length})$.

\begin{verbatim}
Fcn-X(A, u, v, p, r)
    1  if $p > r$
    2      return 0
    3  if $p == r$
    4      if $A[p] > v$
    5          return 1
    6      elseif $A[p] < u$
    7          return -1
    8      else
    9          return 0
10  $q = [p + (r - p)/2]$
11  return (Fcn-X(A, u, v, p, q) + Fcn-X(A, u, v, q + 1, r))
\end{verbatim}

Answer the following questions concisely:

1. (5 points) What does the function do?

2. (10 points) Give a recurrence for its running time $T(n)$ using asymptotic notation and solve it. What can you say about its worst, average and best case?
3. (10 points) Write the function using an incremental (as opposed to divide-and-conquer) approach.

4. (8 points) Give its running time $T(n)$ in asymptotic notation.

5. (5 points) How much did $T(n)$ improve using the incremental approach? Can you think of a way to reduce the running time order?

**Exercise 3: recurrent equations (42 points).** (This is based on textbook problems 4.1 and 4.3.) Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for $n \leq 2$.

1. (6 points) $T(n) = T(2n/3) + \sqrt{n}$
2. (6 points) $T(n) = 4T(\sqrt{n}) + 1$
3. (6 points) $T(n) = 27T(n/3) + n^3$
4. (6 points) $T(n) = 13T(n/2) + n^3 + 3n\sqrt{n} + 1$
5. (6 points) $T(n) = 7T(n/3) + n^2$
6. (6 points) $T(n) = 2T(n/8) + n^{1/3}$
7. (6 points) $T(n) = T(n-1) + n$

Justify your answers. If you use the master theorem, specify which case and show that its hypotheses are satisfied. If you use recursion trees to find a good guess, verify the guess with the substitution method.

**Bonus exercise: recurrent equations (10 points).** Consider the recurrence

$T(n) = aT(n/b) + f(n)$.

Case 3 of the master theorem requires that $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$ and that $f(n)$ satisfies the regularity condition $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large $n$. Prove that the regularity condition is always satisfied if $f(n) = n^k$. (This means we don’t need to check it when $f$ is a polynomial.)

**Bonus exercise: recurrent equations (30 points).** Consider the following particular type of recurrence:

$T(n) = \begin{cases} 
1, & n = 1 \\
aT(n/b) + n^c, & n > 1 
\end{cases}$

where $a \geq 1$ and $b > 1$ are integers, and $k = \log_bn$ is integer (that is, we can only pick sizes $n$ of the form $n = b^k$ where $k \geq 0$ is an integer). Prove the master theorem for this recurrence, that is, prove that

$T(n) = \begin{cases} 
\Theta(n^{\log_b a}), & c < \log_b a \\
\Theta(n^c \lg n), & c = \log_b a \\
\Theta(n^c), & c > \log_b a. 
\end{cases}$

*Hint:* draw the recursion tree, indicating the tree levels, subproblem sizes and subproblem costs, and then sum all the costs.