Assignment
Problems identified by x.y(z) denote the problem “y”, in chapter “x” of the textbook, with part “z”. If “z” is not noted, then the entire problem is required.

Assignment 2a, Due Jan 17
- 2.5(a, c, e) Use the master theorem, show work.
- Solve recurrence relation $T(n) = 2T(n/3) + n$. Use the master theorem, show work.

Assignment 2b, Due Jan 19
- 2.5(b, d) Use the master theorem, show comparison.
- Solve recurrence relation $T(n) = 8T(n/3) + n^2$. Use the master theorem, show work.
- 2.5(g) Use the substitution method. Show the pattern and determination of $k_{max}$.

Assignment 2c, Due Jan 22
- 2.5(f, h) Use the substitution method. Show the pattern and determination of $k_{max}$.
- 2.16 Find an algorithm, give pseudo-code, argue correctness, analyze the runtime, showing it is $O(\log(n))$. The values stored are integers, not necessarily positive. Hint: You should know how to find items in a sorted array in $O(\log(n))$.
- Write the function `unsigned int binary_search( const std::vector< int > &data, int value )`. Verify that the function will correctly find the index of `value` within `data`. You may assume that `value` is present, and `data` is already sorted in ascending order. Submit statement of correctness, and estimated Big-Oh complexity of the algorithms. For Python students, your function will receive a list of numbers, and a number from the list. It will return the index of the number. (Same as the C++ version).

TO BE UPDATED BELOW HERE

Due Feb 9
- 2.5(i, j) Use the substitution method. Show the pattern and determination of $k_{max}$.
- 2.12 Write down the recurrence relation and solve it.
- 2.4(A) Write down the recurrence relation. Solve it.

Due Feb 12
- 2.5(k) Use the substitution method. Show the pattern and determination of $k_{max}$.
- 2.22 Find an algorithm, give pseudo-code, argue correctness, analyze the runtime, showing it is $O(\log(m) + \log(n))$.
- If one algorithm is $O(\log(m+n))$, another is $O(\log(m) + \log(n))$, which is more efficient?
- 2.4(B) Write down the recurrence relation. Solve it.
- Write two functions `unsigned int binary_search( const std::vector< int > &data, int value )` and `unsigned int ternary_search( const std::vector< int > &data, int value )`. Verify that both functions will correctly find the correct index of `value` within `data`. You may assume that `value` is present, and `data` is already sorted in ascending order. Submit statement of correctness, and estimated Big-Oh complexity of both algorithms.

Due Feb 14
- 2.25(a) Fill in the missing code, give a recurrence relation, and solve it.
- 2.14 Find a divide-and-conquer algorithm, write the recurrence relation, solve it.
- 2.4© Write down the recurrence relation. Solve it.
- Time `binary_search` and `ternary_search` on vectors of sizes $2^0, 2^1, ... , 2^{30}$. Be sure to do correct statistical data collection. Submit a statement of data collected, and declaration of which appears to be faster.

Due Feb 16
- 2.25(b) Fill in the missing code, give a recurrence relation, and solve it.
- 2.4 Which would you choose?
2.17 Find an algorithm, prove the runtime is $O(\log(n))$.
Chart the normalized runtimes of $\text{binary_search}$ and $\text{ternary_search}$, along with $N^{\frac{1}{2}}, N^{\frac{1}{3}}, \log_2(N), \log_3(N)$ and 1. Submit the chart, and a statement discussing which algorithm is faster.

Submission
- At the beginning of class on the due dates, submit paper copies of your solutions.