1. Acknowledge all sources and collaborations. If you do not give an acknowledgement statement, your assignment may not be graded.

2. Use the divide and conquer multiplication algorithm to multiply binary 10010101 and 10110011.

3. Solve each of the following recurrence relations. Give a Θ bound for each.

   (i) $T(n) = 3T(n/2) + 5$

   (ii) $T(n) = 5T(n/6) + \sqrt{n}$

   (iii) $T(n) = 8T(n/8) + n$

   (iv) $T(n) = 16T(n/4) + n^2$.

   (v) $T(n) = 16T(n/4) + n^{2.5}$

   (vi) $T(n) = 16T(n/4) + n^{1.9}$

   (vii) $T(n) = 2T(n-2) + 2$

   (viii) $T(n) = T(n^{1/3}) + 3$.

4. (i) The proof of correctness of the matrix multiplication algorithm on page 57 of the text reduces to four cases. For the case of the upper right quadrant, prove that using $P_1 + P_2$ gives the correct result.

   (ii) Prof. Lake claims to have found a better matrix multiplication algorithm. His algorithm decomposes an $n \times n$ matrix in the usual way, and then computes the matrix product using 3 recursive calls (each of size $n/2 \times n/2$) and 3691 $n/2 \times n/2$ submatrix additions or subtractions. Give the runtime of Lake's algorithm.

   (iii) Prove that Lake's algorithm cannot be correct. Hint: give a lower bound for the time to multiply two $n \times n$ matrices.

5. A sorting algorithm is *perfect* if it requires the minimum possible number of key comparisons in the worst case. E.g., a perfect sorting algorithm requires 0,1,3,5 key comparisons respectively to sort lists of size 1,2,3,4.

   (i) Design a perfect sorting algorithm for a list with 4 keys.

   (ii) Prove that an algorithm that uses only key comparisons needs at least 5 key comparisons in the worst case to sort 4 elements.

6. Recall that merging sorted lists of size $x \geq 1$ and $y \geq 1$ takes $x + y - 1$ key comparisons in the worst case. How many key comparisons does it take to merge these lists in the best case? Justify briefly.

7. M3sort takes an unsorted list of $n \geq 1$ elements and works as follows. If $n \leq 5$, use a perfect sorting algorithm and return. Otherwise, recursively M3sort the first $n/3$ elements, copy into sublist A. Then recursively M3sort the second $n/3$ elements, copy into sublist B. Then recursively M3sort the remaining elements, copy into sublist C. Then merge lists A and B, copy into list D. Then merge lists D and C, return this list.

   (i) Argue briefly that M3sort is correct.

   (ii) Give a recurrence relation describing the worst case number of key comparisons performed by a M3sort of n keys.

   (iii) Using Θ notation, give the runtime of M3sort.

   (iv) Write a program that computes the worst-case number of key comparisons performed by M3sort, and also the worst-case number of key comparisons of mergesort (assuming that mergesort also uses a perfect sort for $n \leq 5$), and print these numbers for n up to 36.

   (v) What do you think of M3sort?

8. Consider the graph on page 86 of the text. Assume that the list of vertices, and each vertex's list of neighbours, is stored in reverse alphabetic order.

   (i) Draw the BFS search forest for the graph.

   (ii) Draw the DFS search forest for the graph.