Matthew Soicher
z5160737

Question 1

Our goal is to calculate the number of ways we can put parenthesis into the expression so that it will evaluate to true

Let us denote $T(i,j)$ as the number of ways to parenthesise symbols i to j so that the subexpression evaluates to true. Conversely, $F(i,j)$ for the subexpression to evaluate to false.

Our solution is to consider from the bottom-up, substrings of our expression between the i^{th} and j^{th} symbol such that they evaluate to true

Our base cases are simply $T(i,i)$ and $F(i,i)$; if i is $true$ our result is 1 and if i is $false$ our result is 0. The reverse applies for $F(i,i)$

For cases whereby $i \neq i$ we split the expression from i to j by parenthesising two subexpressions, namely from i to k and from $k+1$ to j. Each side is grouped in its own bracket and calculated within based upon the appropriate operand,

Then, we evaluate each subexpression and combine the two together dependant on our operand and by what answer we require.

So,

$$T(i,k,j) = \sum_{i \le k \le j-1} \begin{cases} T(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) + F(i,k) * F(k+1,j) \; (NAND) \\ F(i,k) * F(k+1,j) \; (NOR) \\ T(i,k) * T(k+1,j) \; (AND) \\ T(i,k) * F(k+1,j) + T(i,k) * T(k+1,j) + F(i,k) * T(k+1,j) \; (OR) \end{cases}$$

$$F(i,k,j) = \sum_{i \le k \le j-1} \begin{cases} T(i,k) * T(k+1,j) \; (NAND) \\ T(i,k) * T(k+1,j) + T(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) \; (NOR) \\ T(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) + F(i,k) * F(k+1,j) \; (AND) \\ F(i,k) * F(k+1,j) \; (OR) \end{cases}$$

The complexity of the algorithm is $O(n^3)$