Practice Problems - Canonical Forms

1. Find the Disjunctive Normal Form for the following Boolean expressions, assuming the DNF has the number of variables given.

   a. \( f_a = A + B' \) (assume two variables, A and B)
   b. \( f_b = A + B' \) (assume three variables, A, B, and C)
   c. \( f_c = AB'C' \) (assume three variables, A, B, and C)
   d. \( f_d = 1 \) (Assume three variables, A, B, and C)
   e. \( f_e = AB' + A'C + BC' \) (assume three variables, A, B, and C)
   f. \( f_f = (A + B)(A' + C) \) (assume three variables, A, B, and C)

2. Find the Conjunctive Normal Form for the following Boolean expressions, assuming the CNF has the number of variables given.

   a. \( g_a = A + B \) (assume two variables, A and B)
   b. \( g_b = A + B \) (assume three variables, A, B, and C)
   c. \( g_c = ABC + ABC' + AB'C + A'B'C' \) (assume three variables, A, B, and C)
   d. \( g_d = (A + B')(B + C') \) (assume three variables, A, B, and C)
   e. \( g_e = 1 \) (assume two variables, A, and B)

Karnaugh Maps

Karnaugh Maps are the preferred mechanism for the simplification of Boolean expressions. They constitute a graphical means for this purpose, but are, for practical reasons, limited to expressions with relatively few variables, usually less then or equal to four, although they can conceptually be used with as many as desired.

A Karnaugh map is a chart or table of rows and columns. Each row and column is labeled with appropriate Boolean terms. The cells which constitute the intersection of rows and columns contain 1's and 0's, depending on whether the corresponding terms appear in the final function. In fact, a Karnaugh Map is nothing more than a Truth Table.
where each cell in the map corresponds to a row in the equivalent Truth Table.

It is probably easiest, given any arbitrary Boolean expression \( f \), to rewrite \( f \) in a sum-of-products form first. Then the cells in the truth table can be filled in the same manner as the output column of a truth table.

Let’s take a simple two-variable function \( f = A + B' \). This is already in sum-of-products form.

Here is an empty two-variable Karnaugh Map.

\[
\begin{array}{c|c|c|c|c|c}
 & B' & B \\
\hline
A' &   &   \\
A   &   &   \\
\end{array}
\]

The four empty cells correspond to the four possible minterms which might be combined to form any 2-variable Boolean Function: \( A'B' \), \( A'B \), \( AB' \), and \( AB \). For \( f = A + B' \) we fill in all the cells as follows - All the cells where \( A = 1 \) are filled (the bottom row) and all the cells where \( B = 0 \) are filled (the left column).

\[
\begin{array}{c|c|c|c|c|c}
 & B' & B \\
\hline
A' & 1 & 0 \\
A   & 1 & 1 \\
\end{array}
\]

Now, to simplify the function we ‘loop’ the largest possible rectangular areas containing \( 2 \times 1 \)'s within the table. In this case, we can form two loops, one containing the two vertical cells under the \( B' \) column, and one containing the two horizontal cells in the \( A \) row. Note that loops may overlap - this will usually be the case if we insist on choosing the largest possible size for each loop.

\[
\begin{array}{c|c|c|c|c|c}
 & B' & B \\
\hline
A' & 1 & 0 \\
A   & 1 & 1 \\
\end{array}
\]

It now just remains to write the function. Each loop corresponds to a term in the final function. **Wherever a loop includes both a variable and its complement, then that variable is eliminated.** Basically, this mechanical method implements the complement
rule. In this example, the two loops are

\[ B' \quad \text{[from } A'B' + AB' = (A' + A) B' = 1 \cdot B' = B'] \quad \text{and} \]

\[ A \quad \text{[from } AB' + AB = A(B' + B) = A \cdot 1 = A] \]

so the simplified function is

\[ f = A + B' \]

When drawing karnaugh maps of three or more variables, care must be taken in labeling the rows and columns to ensure that **as you travel across rows or down columns, only one variable changes at a time.** Also, be aware that **loops may wrap from top to bottom and left to right.** Consider the following three variable map. Three loops will be required to cover all the 1’s, but notice that the simplified result is not unique. Two different loopings are shown

<table>
<thead>
<tr>
<th>B'C'</th>
<th>B'C</th>
<th>BC</th>
<th>BC'</th>
</tr>
</thead>
<tbody>
<tr>
<td>A'</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

In both tables, the two loops which appear to go off the edges of the table are actually the same loop which wraps around to get the two 1’s in the top row (giving A'C'). The second loop encompasses the two middle 1’s in the bottom row, giving AC. We need one more loop to pick up the one in the lower right hand corner. We could use a loop that encircles just that one cell, but you must always loop as many cells as possible to ensure a minimally simplified expression. In this case, then, we have a choice for the third loop as shown in the two tables, giving either BC' or AB. Thus, there are two possible solutions for the function:
The student should verify that these are equivalent using any of the methods (truth tables, DNF, or algebraic) already presented.

Remember, as many 1's as possible should be included in each loop. Loops may overlap (that is, terms often appear in more than one loop).

**Practice problems-Karnaugh Maps**

1. Write the simplified function given by the following Karnaugh maps:
   a. 
   
<table>
<thead>
<tr>
<th>B'C'</th>
<th>B'C</th>
<th>BC</th>
<th>BC'</th>
</tr>
</thead>
<tbody>
<tr>
<td>A'</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
   
   b. 
   
<table>
<thead>
<tr>
<th>B'C'</th>
<th>B'C</th>
<th>BC</th>
<th>BC'</th>
</tr>
</thead>
<tbody>
<tr>
<td>A'</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

2. Show the Karnaugh Maps and simplify the following Boolean expressions:
   a. \( A'(B' + C) + BC' \)
   b. \( AB + B'(C + A'D) + (B + CD) \)
   c. \( AB + AB' A'B + A'B' \)
   d. \( ABC + ABC' + AB'C + AB'C' + A'BC + A'BC' + A'B'C \)

Note that if all possible minterms (or maxterms) appear in an expression, then the value of that expression is 1, regardless of the assigned values of the individual variables. Thus, if there are eight unique minterms in an expression (DNF), or all the rows of a truth table have a 1 in the output (f) column, or all the cells in a Karnaugh Map contain 1, the value of the Boolean expression is 1.