The program starts, and there's a menu where you can choose to calculate or to exit. If you exit, the program terminates. If you calculate, it will ask you for the variables to be used. After correct input, it will ask for integers (minterms that evaluate to 1), and then after correct input, the program displays your inputted variables and minterms, the simplified boolean expression and the menu again. This what you can see, but inside the code, there's much more.

The program is coded in Java. There are two Java classes called UserInterface.java and MoreMethods.java. The MoreMethods class uses methods that will be used for the UserInterface class. The methods include removeDuplicates(int array, int n) and countOccurences(int array, int n, int x). They'll be explained later on in the UserInterface class. In the UserInterface class, this is where everything happens. After inputting your variables and minterms, they get stored in a var array and mint array respectively. The number of minterms and number of variables are also stored in mint_count and var_count respectively. The variable count gets stored in an int called base_two which is 2 ^ var_count and base_two_dim will be minus one of base_two. The variables also get stored in a var_display String array which shows your variables in a string separated by a comma and enclosed by a parenthesis. Your minterms get converted to binary, and they're placed in a mint_group multidimensional array. In the minterm part, the program gets it, and then sorts it for the removeDuplicate() method which is used to remove any duplicates that you inputted in your minterms. The number of minterms changes after passing that method. After that, it prints your inputted minterms and variables.

The next part is a simple if and else statement. The condition for the if statement is it checks if mint_count is equal to base_two. It returns the final simplified boolean function equal to one, and the calculation is finished, and it goes back to the menu.

If it's not, then we move to the else statement which is the longest part of the code. A multidimensional array table is created with a size of mint_count x var_count (row x column). This part is pretty long so I'll try to simplify it. Basically, the table gets grouped first by number of occurrences of

1 in the binary array mint_group in ascending order, and then gets added to a new multi array group. After that, it compares the first pair of numbers. The program iterates through each column and row To be able to iterate through each group, it uses the countOccurences() method. Basically in a series of nested loops, it will find the first occurence (starting at 0), and then the incremented occurence by one. It will then compare those two rows by checking if their difference is a power of 2. If it is, it will add it to group. After it’s done iterating through, it will then continuously compare until it reaches to the second to the last group. The code is almost the same as comparing it the first time, the only difference is that it will keep doing this code until it reaches the second to the last group. For the entire time, the code puts a symbol in the column unaffected by the data that indicates that the row in a group has been used.

The next part is the prime implicant chart. A new multi array is created, and the minterms are displayed on the top row. It will iterate through each row of the group and if the symbol is not present, it will take the minterm of that group row and it add to the prime implicant chart. Basically, it checks if the number in the minterm part of each row is present and puts a symbol indicating it in the chart. Next, it checks each column if there’s a presence of a singular x. If there is, it checks it and it goes back to the row of the minterm(s) and checks it and then checks every column that has the numbers in the minterms. After that for the rest of the remaining implicants, it checks the unchecked columns. If there’s only one unchecked, it will get whatever minterm appears first in the row when iterated going down til the last row. After that, it will check each column and get the first extracted one that completes the most unchecked part, and then so on. After that, the simplified boolean expression is displayed, and then the calculation ends going back to the menu.