Ale1

* Assignment 1: Creates the infix string by using a stack to process a list of all nodes in-order from postfix.
* Assignment 2: Creates the table using a multidimensional array which is being filled in a nested loop which uses the remainder of the indeces to assign T or F values.
* Assignment 3: Applies the Quine-McCluskey Algorithm to simplify the truth table. <https://en.wikipedia.org/wiki/Quine–McCluskey_algorithm>
* Assigment 4:
* Disjunctive normal form (DNF): Step 1: Iterate through any given row of the truth table and then find formulas/conjuncts. Step 2: Combine all formulas into the complete DNF.
* Simplified disjunctive normal form (SDNF): The SDNF creation behave similar to the DNF's, however, don't care symbols are omitted.
* Assignment 5: The NAND proposition is created with a stack and the given nodes of the proposition. Hereby the in order algorithm is used to create a NAND proposition.

Side notes:

* The application makes use of the MVVM paradigm pattern which greatly enables testability, maintainability, and scalability. Furthermore, it allows workflows, such as the designer-developer workflow
* Certain errors are handled by try/catch clauses. The error message is logged through Debug.Writeline() in the console. For instance, entering an invalid proposition.
* Consider a tautology, then a DNF or SDNF could be represented as "true". In my program the DNF is still being generated as in any other proposition, however, the SDNF is the given prefix input.  
  On the other hand, if the proposition is a contradiction the DNF or SDNF do not exist. Therefore,the given prefix input is displayed.
* The program makes use of graphviz. The installation path is C:\Program Files (x86)\Graphviz2.38\. If the installation varies from the default installation then the file GraphVizService.cs needs to be touched by editing line 20 and 21 accordingly.
* When entering larger propositions it is recommend to maximize the application window.