Alexio Nugroho, Harshavardhan Mishra, Lucy Cui, Peter Zhang

Propositional Logic Learning Tool

Introduction

Through this project, we endeavour to develop a tool to assist high school and college students in their studies of Theory of Computation, with a specific focus on the mastery of logical principles and development of logical-thinking skills. More specifically, our project would entail creating a tool that would allow educators to input complex propositional logic statements and allow them to simplify said logic statements, thus rendering the learning process more accessible to the students. The motivation behind our project stems from the collective experience of our cohort, who encountered difficulties in their pursuit of comprehension of the simplification process. 1.1

We believe that there exists a void wherein there does not exist enough practice questions for students to thoroughly develop their simplification abilities, and this is the void that we aim to fill with our tool, which we hope will serve as a valuable addition to the existing educational resources available.

Our tool will apply the use of Karnaugh Maps otherwise known as K-map. A K-map is a graphical technique used to simplify Boolean functions by eliminating the need for complicated mathematical theorems or manipulations. This will significantly help reduce the complexity of the boolean expressions and can reduce the amount of calculations done either in code or on paper.

It will have two modes: a ‘calculator’ mode where the tool will directly simplify the expression, and an ‘exercise’ mode where you have to manually fill in the karnaugh map and the simplified expression. The tool will tell the user whether the karnaugh map is correctly filled in and whether the simplified expression is valid or not.

This tool is of reasonable scope; its development will put our coding abilities to the test; however, as its implementation draws upon the data structures and coding concepts that we have studied, it should be possible to complete within the given timeframe. This imbues our project with a healthy measure of ambition, but never to the extent of being unrealistic. The tool is also interesting in the way that a project like this has not been undertaken in the past, making it even more relevant to the target audience, and to third-party stakeholders.

Features

1. Allow the educator to input the complex propositional logic statement questions into the tool
    - We aim to have two methods to do this: either a file input method where the teacher uploads a file with questions - as files are a form of persistent storage, these would be saved even when the application is terminated and opened again the next time. The other method of input that we plan on having is 1.2 ectly into the tool upon starting, wherein questions will be temporarily held in a data structure and can be used for the duration that the program is

run for. However, as data structures are not a mode of persistent storage, these questions would be lost once the program is terminated.

2. Complex Propositional Formula Simplification Calculator
    - The purpose of this second feature is to help students quickly check their simplified versions of complex propositional formulas that they might have found elsewhere. This tool can quickly help students ensure that they have reached the correct answer to their question as it will compare their input to the system generated answer and mark it accordingly.

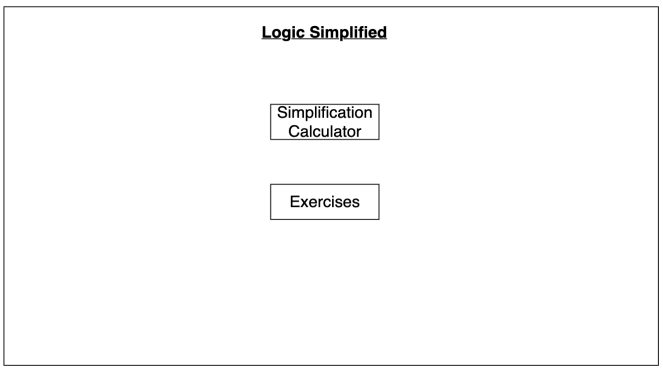
3. Complex Propositional Formula Simplification Exercises
    - In the exercises mode, users will have the opportunity to practise their K-map skills by completing a K-map on their own. The results of their work will be compared with the tool's generated K-map, allowing users to assess their accuracy. Users will also be challenged to provide their own simplified logic statements, which will be verified for validity. Through this hands-on experience, users can develop their own proficiency in simplifying expressions, both with and without the aid of the tool.

4. Virtual keyboard as the method of user input
    - As it is rather difficult to type in the symbols that form propositional logic statements, we will provide a virtual keyboard with all required symbols on it so that students and educators do not have to waste time in finding them on their keyboard. This will help to improve the ease-of-use of the product.

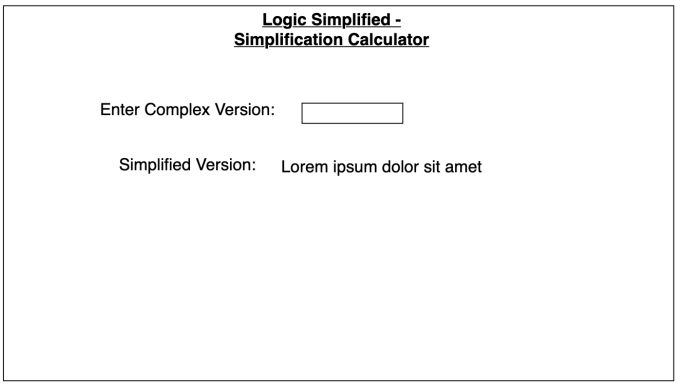
2.1

5. Progress tracker
    - This feature will help the student keep track of their accuracy level, and will be updated over time each time a student gets a question right or wrong.

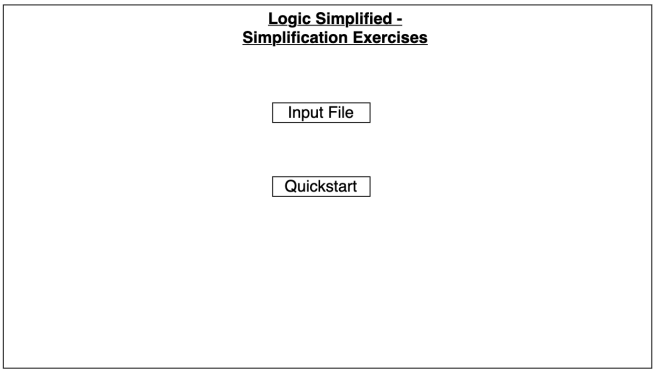
GUI MockUp



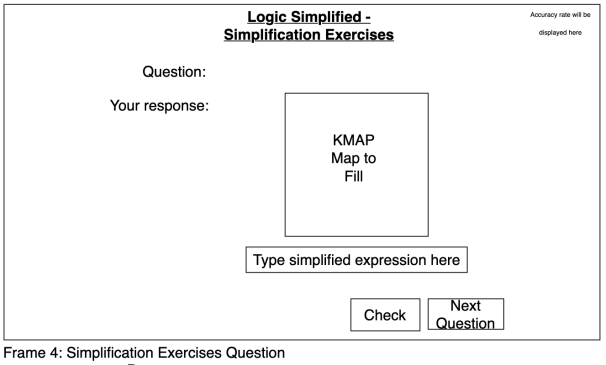
Frame 1: Home Page



Frame 2: Simplification Calculator

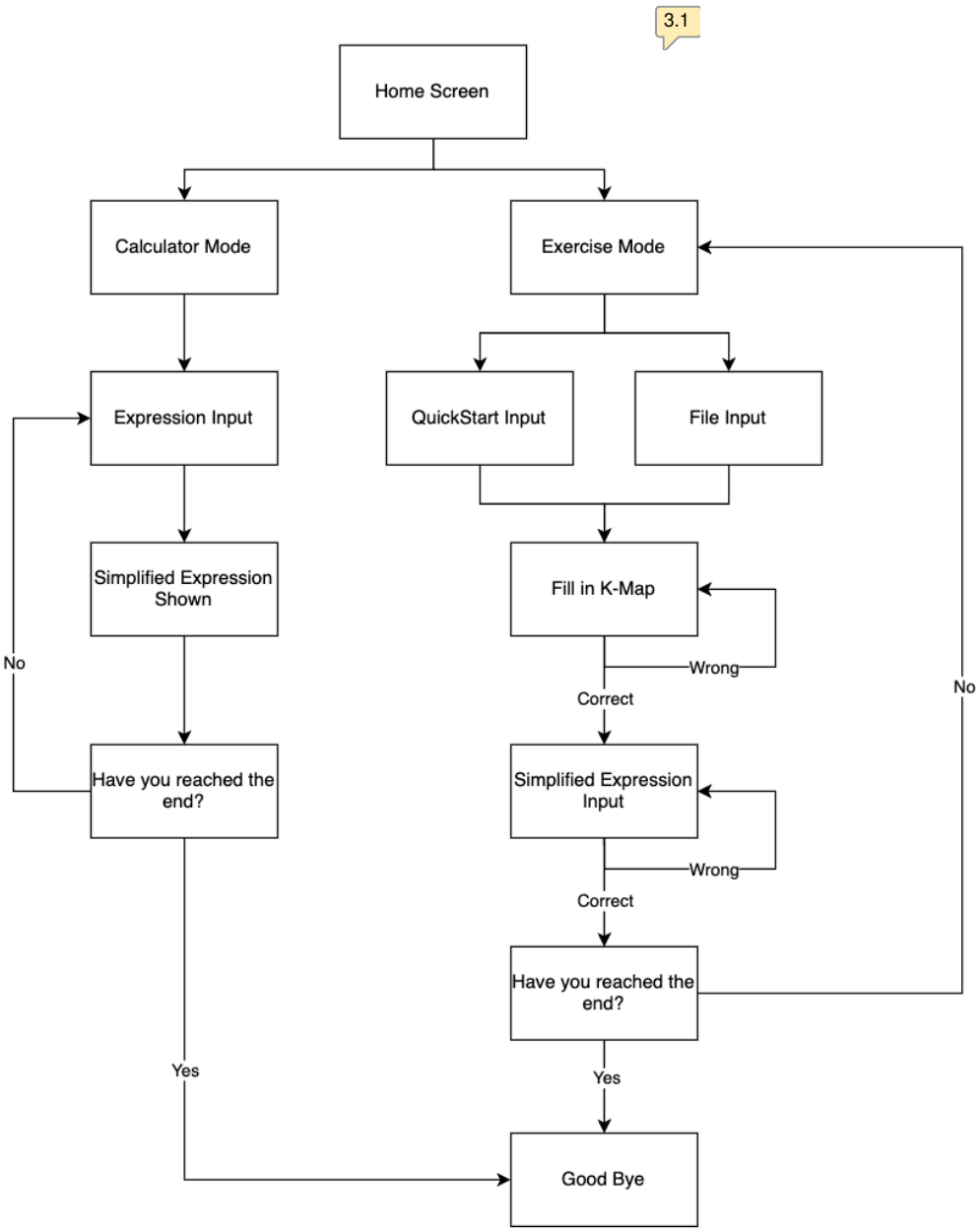


Frame 3: Simplification Exercises Home Page



Frame 4: Simplification Exercises Question Page

Overall Functionality of the System



Components

- **Data Structure**
  A hashmap, with a question as the key and a 2D array as the value. Used to store any questions inputted within the quickstart submode within the exercises mode

- **Data Storage**
  Files : used to store pre-written questions, can be loaded into the system
  String: used to store on the spot written statement

4.1

- **Algorithms**
- For exercise mode:
- An algorithm to check the correctness of manually filled K-map
    - Compare the filled in K-map with the stored answer and point out the differences as mistakes.
- An algorithm to keep track of the accuracy level.
    - Record the total number of questions done and the number of correctly answered questions.

- For calculator mode:
- An algorithm to convert a Boolean Expression to a K-map.
    - For each valuation of letters, judge whether the boolean expression is true, and fill in the K-map accordingly: 0 for false, and 1 for true.
- An algorithm to generate simplified Boolean Expression.
    - Detailedly described below.

The function has 3 main variables: minterms, source, and flag.

4.2

Minterms store terms like '1001' and '10**'. The source keeps track of where a term originates from, for example, the term '1001' comes from itself and its source is only its index in minterms [1]. However, for '10**', it was generated by ['1000', '1001', '1010', '1011'] and its source is [1, 2, 3, 4]. The flag checks if a term has been used to generate a new term.

The function is divided into 2 parts:

1) simplifying all the terms until no further simplification is possible (the example's output is: ['10**', '10', '1**0', '*110'] which is equivalent to AB' + AC' + AD' + BCD'),

2) checking the source of all terms. If a term's source lacks a unique source or has a unique term(s) equal to "don't care," it is considered unnecessary and removed to get the final result.

Pseudocode of the algorithm

```
DEFINE str_terms AS STRING LIST
DEFINE terms_not_care AS STRING LIST
DEFINE t_minterms AS STRING LIST
DEFINE not_cares AS STRING LIST
DEFINE minterms AS STRING LIST

DEFINE Class Minterms (Object)
    PROCEDURE __INIT__(minterms, not_cares <- NONE):
    IF minterms is NONE:
      THEN
        minterms <- []
    ENDIF
    IF not_cares is NONE:
      THEN
        not_cares <- []
    ENDIF
    ENDPROCEDURE

    SET Object.minterms <- minterms
    SET Object.not_cares <- not_cares

    PROCEDURE simplify(Object):
    prime_imlicants <- CALL find_prime_implicants(Object.minterms,Object.not_car
es)
    result <- CALL find_essential_prime_implicants(prime_imlicants,Object.minter
ms)
    ENDPROCEDURE


INPUT str_terms, terms_not_care
FOR each elements in str_terms:
    t_minterms ADD (SET elements AS an Object)
ENDFOR
FOR each elements in not_cares:
    not_cares ADD (SET elements AS an Object)
ENDFOR

CALL Minterms(t_minterms)
    THEN CALL PROCEDURE simplify
CALL Minterms(not_cares)
    THEN CALL PROCEDURE simplify
```

Examples of Exercises

1. Input:
   ((P → Q) ∧ (Q → R)) ∧ (R → P)

   Kmap:

	PQ	PQ'	P'Q'	P'Q
R	1	0	0	0
R'	0	0	1	0

   Simplified statement: (P ∧ Q ∧ R) ∨ (¬P ∧ ¬Q ∧ ¬R)

2. Input:
   (¬P ∧ ¬Q ∧ ¬R) ∨ (¬P ∧ Q ∧ ¬R) ∨ (P ∧ ¬Q ∧ ¬R) ∨ (P ∧ ¬Q ∧ R) ∨ (P ∧ Q ∧ ¬R) ∨ (P ∧ Q ∧ R)

   Kmap

	PQ	PQ'	P'Q'	P'Q
R	1	1	0	0
R'	1	1	1	1

   Simplified statement: P ∨ ¬R

3. Input:
   (¬P ∧ ¬Q ∧ ¬R ∧ ¬W) ∨ (¬P ∧ ¬Q ∧ R ∧ ¬W) ∨ (¬P ∧ Q ∧ R ∧ ¬W) ∨ (¬P ∧ Q ∧ R ∧ W) ∨ (P ∧ ¬Q ∧ ¬R ∧ ¬W) ∨ (P ∧ Q ∧ R ∧ ¬W) ∨ (P ∧ Q ∧ R ∧ W)

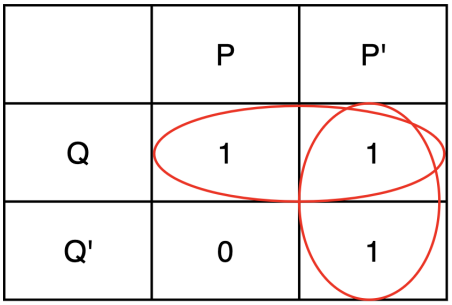
   Kmap:

	PQ	PQ'	P'Q'	P'Q
RW	1	0	0	1
RW'	0	1	1	0
R'W'	0	1	1	0
R'W	1	0	0	1

Simplified statement: (Q ∧ W) ∨ (¬Q ∧ ¬W)

4. Input:
   (¬P ∧ ¬Q) ∨ (¬P ∧ Q) ∨ (P ∧ Q)

   Kmap:



	P	P'
Q	1	1
Q'	0	1

   Simplified statement: ¬P ∨ Q

7.1

Language

Our team reached a conclusion on utilising Python for our project. The reason behind our decision was twofold. Firstly, we are already familiar with the language, which will save us time in getting up to speed with the new project. Secondly, the syntax of Python is easy to read, which will make it easier for us to collaborate and maintain the code.

GUI Tool: PyQt and Tkinter are two popular GUI (Graphical User Interface) frameworks for Python, and we will be using one of these for our project. Each framework has its own set of advantages:

Advantages of PyQt:

- Wide range of functionality: PyQt offers a comprehensive set of tools for GUI development, including a wide range of widgets, event handling, and signal-slot mechanisms.
- Robustness: PyQt has been used to develop large, complex applications and is known for its stability and reliability.
- Cross-platform compatibility: PyQt applications can be run on multiple platforms, including Windows, macOS, and Linux.

Advantages of Tkinter:

- Simple and easy to use: Tkinter is easy to use and provides a simple interface for GUI development.
- Built-in library: Tkinter is part of the standard Python library, which means it is available out-of-the-box and does not require any additional installation.
- Lightweight: Tkinter is a lightweight library, making it suitable for small applications or prototypes.

In summary, PyQt is a more robust and feature-rich library, while Tkinter is simpler and easier to use. The choice between the two largely depends on the specific requirements and constraints of a project. For now, we plan on using PyQt due to its wide range of functionality but we may switch gears to Tkinter upon starting development given that more of us have used this tool in the past.

Project Management

For project management, we will use GitHub, for the following reasons:

1. Version Control: GitHub provides Git-based version control, which will allow us to track changes to your code, revert to previous versions if needed, and collaborate with each other in real-time.
2. Collaboration: GitHub allows work to continue regardless of location. One can easily share your code, track changes, and merge contributions from other team members.
3. Bug Tracking and Project Management: GitHub provides issue tracking and project management tools which will help us to keep the project organised and on track.

References

Introduction to karnaugh maps - combinational logic circuits, functions, & truth tables (2019) *YouTube*. YouTube. Available at: https://www.youtube.com/watch?v=RO5alU6PpSU (Accessed: February 6, 2023).

Karnaugh map (2023) *Wikipedia*. Wikimedia Foundation. Available at: https://en.wikipedia.org/wiki/Karnaugh_map (Accessed: February 6, 2023).

Karnaugh maps – introduction (2016) *YouTube*. YouTube. Available at: https://www.youtube.com/watch?v=3vkMgTmieZI (Accessed: February 6, 2023).

Karnaugh maps – simplify boolean expressions (2016) *YouTube*. YouTube. Available at: https://www.youtube.com/watch?v=UfZKvPQku8w (Accessed: February 6, 2023).

Python for beginners – full course [programming tutorial] (2022) *YouTube*. YouTube. Available at: https://www.youtube.com/watch?v=eWRfhZUzrAc&t=1964s (Accessed: February 6, 2023).

Storr, W. (2022) *Boolean algebra simplification*, *Basic Electronics Tutorials*. Available at: https://www.electronics-tutorials.ws/boolean/boolean-algebra-simplification.html (Accessed: February 6, 2023).

Index of comments

1.1	Introduction: provides a clear explanation of the project's purpose and relevance to the target audience.
1.2	is by directly inputting
2.1	Would be helpful to include visuals for virtual keyboard and progress tracker as specified.
3.1	Some of these boxes represent screens, and some others represent user actions. It is worth putting a key/legend to guide readers and differentiate different visual elements.
4.1	May be helpful to provide a more straightforward explanation or a flowchart to accompany the technical description.
4.2	Please cite github or appropriate source here.
7.1	Subheadings and numbering can be used to guide readers better and provide clarity
8.1	Overall, a good project proposal. Some additional material/information could be included. Approach in terms of tools used and project management seems reasonable although is light on detail. Minor issues in terms of report content/clarity. I like how you've put thought into the K-map algorithm.