



USER MANUAL

QUINE-MCCLUSKEY CALCULATOR

DEVELOPED BY:

ACOSTA, ARIANNE JAYNE D.
BONIFACIO, CHRISTIAN JESSE Q.



TABLE OF CONTENTS

I. INTRODUCTION

1.1 The QuineMcCluskeyCalculator	3
--	---

II. GRAPHICAL USER INTERFACE (GUI)

2.1 GUI	4
2.2 Layout.....	5
2.3 Components	6

III. INSTALLATION

3.1 Step 1: Downloading	8
3.2 Step 2: Open the Project	9
3.3 Step 3: Install Libraries	10
3.4 Project	11

IV. RUNNING THE PROGRAM

4.1 Running the Program	12
4.2 Using the Calculator	13

V. TROUBLESHOOTING

5.1 Method in MainGUI.java	15
5.2 Support	16

INTRODUCTION

Introducing the **Quine-McCluskey Calculator**, a sophisticated Java software crafted to aid in comprehending and utilizing the Quine-McCluskey method for reducing Boolean functions. Developed by Arianne Jayne Acosta and Christian Jesse Bonifacio, this program offers a robust solution for simplifying complex Boolean functions.

The Quine-McCluskey method is a crucial technique in digital design and Boolean algebra for minimizing logical functions. By iteratively combining terms and identifying essential prime implicants, the method produces a simplified Boolean expression that retains the same logic as the original but with fewer terms. This simulator aims to demystify the Quine-McCluskey method, making it accessible to students, educators, and professionals in the field of digital logic design.

The software undertakes a **dual-phase approach** to achieve Boolean function simplification. Initially, it utilizes the Quine-McCluskey method to detect prime implicants and iteratively enhance the expression. The subsequent stage entails additional refinement through techniques like row dominance, column dominance, and the utilization of **Petrick's method**.

This software accommodates a wide-ranging audience encompassing individuals engaged in digital logic design, computer science, and associated disciplines. Whether you're a student wrestling with Boolean algebra or a professional in search of a dependable tool for function simplification, this program is tailored to suit your requirements.

GUI

Graphical User Interface (GUI)

The system incorporates a Graphical User Interface (GUI) is designed with JavaFX, providing a user-friendly and intuitive interaction platform.

The GUI maintains a clean and organized layout, enhancing the overall user experience. It simplifies complex operations and presents information in an easily understandable format, making the system accessible to all users.

Quine-McCluskey Calculator

Minterms: Enter minterms... (e.g. 0,1,2 or 0 1 2)

Custom Variables: Yes No

Variables: A, B, C, D, E, F, G, H, I, J

Solve

Solution

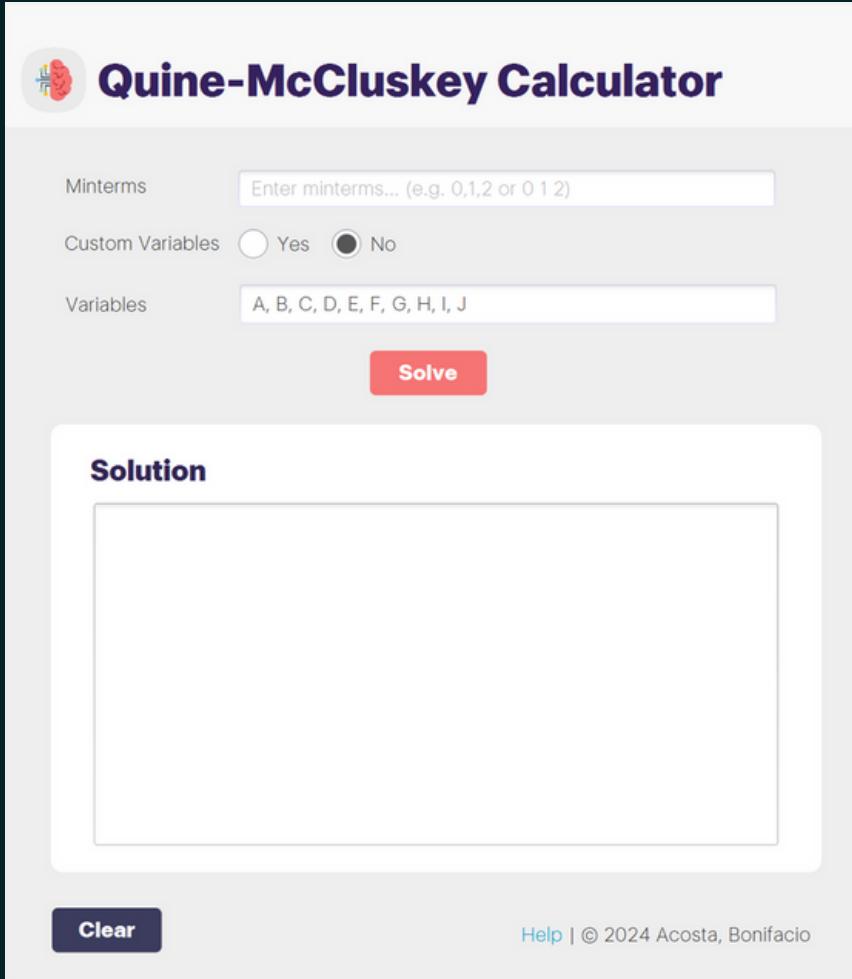
Clear [Help](#) | © 2024 Acosta, Bonifacio

LAYOUT

Main Input
Fields and
Controls

Solution
Panel

Footer



The image shows the Quine-McCluskey Calculator interface. At the top, there is a logo of a brain inside a circle and the title "Quine-McCluskey Calculator". Below the title, there are three input fields: "Minterms" with a placeholder "Enter minterms... (e.g. 0,1,2 or 0 1 2)", "Custom Variables" with radio buttons for "Yes" and "No" where "No" is selected, and "Variables" with a list "A, B, C, D, E, F, G, H, I, J". A red "Solve" button is located below these fields. Below the input area, there is a section titled "Solution" which contains a large empty rectangular box. At the bottom of the interface, there is a "Clear" button on the left and a "Help | © 2024 Acosta, Bonifacio" link on the right.

COMPONENTS

1. Main Input Fields and Controls

Components which allow users to interact with the program and run the Calculator

The screenshot shows a user interface for a Boolean function calculator. It includes:

- A text input field labeled "Minterms" with placeholder text "Enter minterms... (e.g. 0,1,2 or 0 1 2)".
- A section for "Custom Variables" with two radio buttons: "Yes" (unchecked) and "No" (checked).
- A text input field labeled "Variables" containing the default variables "A, B, C, D, E, F, G, H, I, J".
- A red "Solve" button at the bottom.

Enter Minterms Text Field

Text field where users enter the list of minterms of the Boolean function to be simplified. It is required that the minterms be integers which are delimited by commas or spaces. Throws error windows* if format is not followed.

Custom Variables Radio Buttons

The “yes” and “no” buttons determine if the user prefers to use their own variables for the program.

Depending on which button is clicked, the Variables Text Field is updated.

Variables Text Field

Displays default 10 variables (A,B,C,D,E,F,G,H,I,J) or allows for custom variables if user clicks Yes.

Yes selected

The screenshot shows the interface with "Yes" selected for Custom Variables. The "Variables" field contains "Enter 10 variables... (e.g. Q,W,E,R,T,Y,U,I,O,P)".

No selected

The screenshot shows the interface with "No" selected for Custom Variables. The "Variables" field contains "A, B, C, D, E, F, G, H, I, J".

Solve Button

Runs the program. Error windows* are displayed if minterm and/or variable input is invalid.

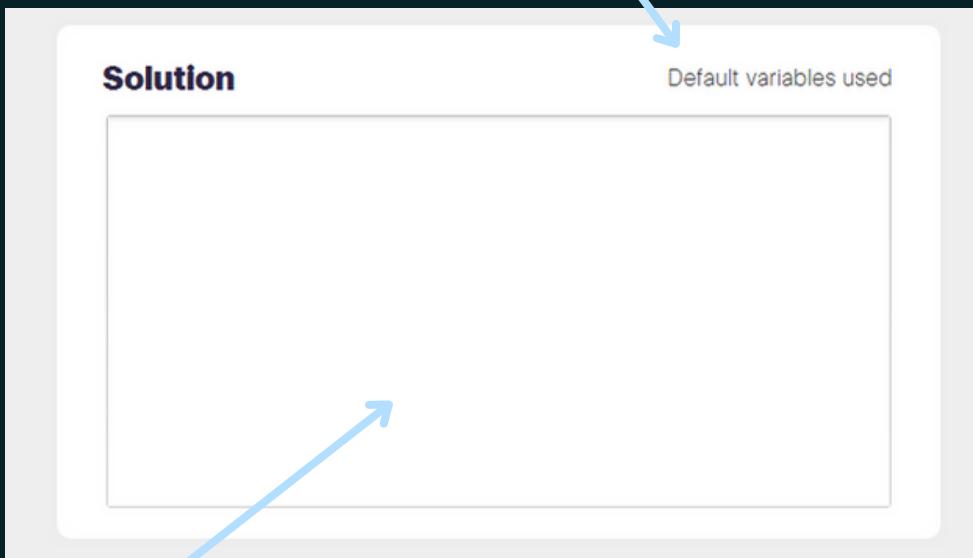
*See page 15 for Troubleshooting.

COMPONENTS

2. Solution Panel

Variables Information Label

Once Solve is clicked, it appears and displays information based on the variables chosen for the simplification.



Solution Text Area

Displays simplified Boolean functions. May display more than one solution depending on inputted minterms.

3. Footer

Clear

[Help](#) | © 2024 Acosta, Bonifacio

Clear Button

Clears all input fields and solution window. Resets radio button to default variables (no).

Help Hyperlink

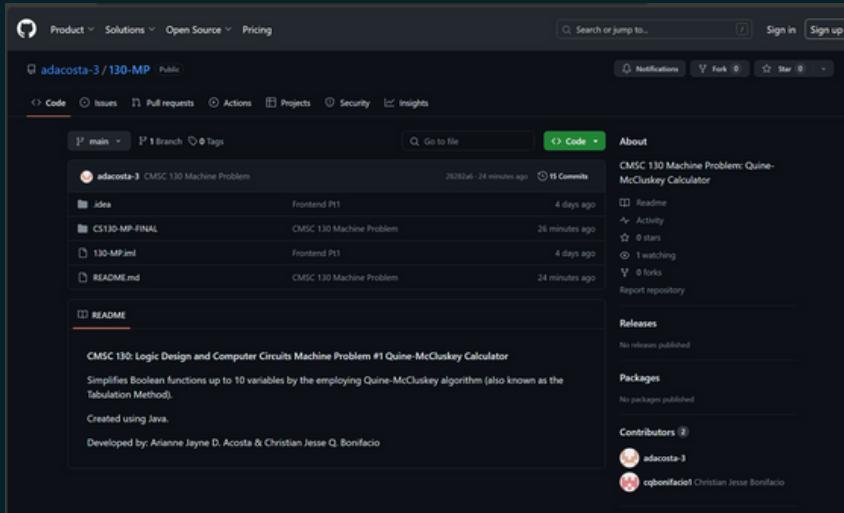
Redirects user to user manual.

INSTALLATION

STEP 1: DOWNLOADING

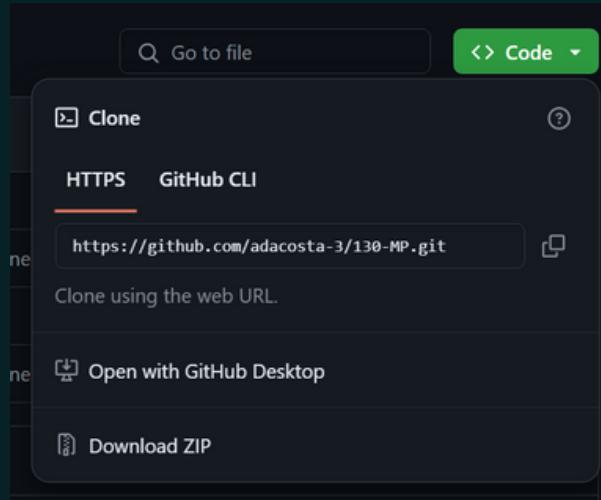
1. Open Github link

Click the icon above to be redirected to the 130-MP GitHub directory.



2. Download ZIP file

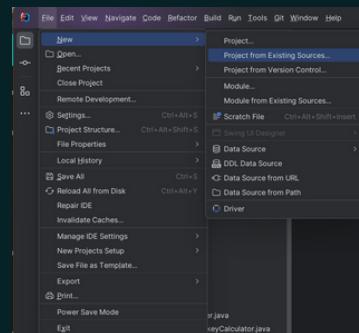
Hover over the `<> Code` button, and click it to reveal Clone options. Click Download ZIP. Unzip the file in your chosen directory.



STEP 2: OPEN PROJECT

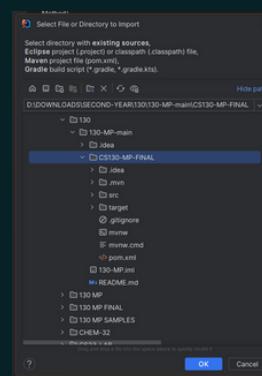
1. Create Project

Open your chosen IDE. To run this program, we recommend using IntelliJ by JetBrains. Under File -> New, create a Project from Existing Sources.



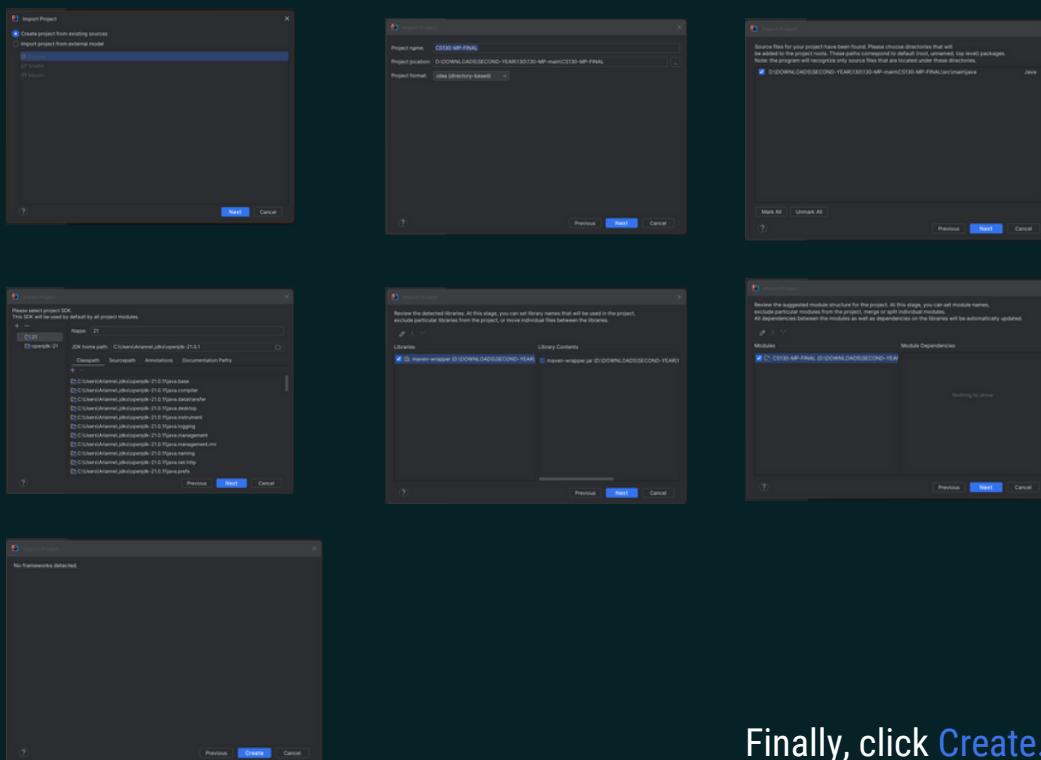
2. Select File

Navigate to the directory you saved the 130-MP-main folder in, click the folder “CS130-MP-FINAL”* then press **OK**. The folder you selected must have the files seen in the image.



3. Import Project

Follow the images below in importing the project. You may choose to set your own name for the project.



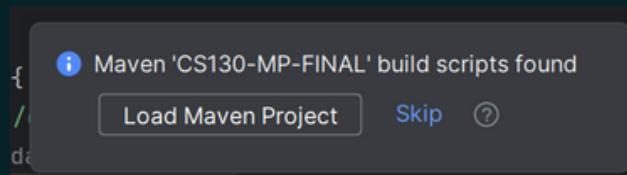
Finally, click **Create**.

INSTALLATION

STEP 3: INSTALL LIBRARIES

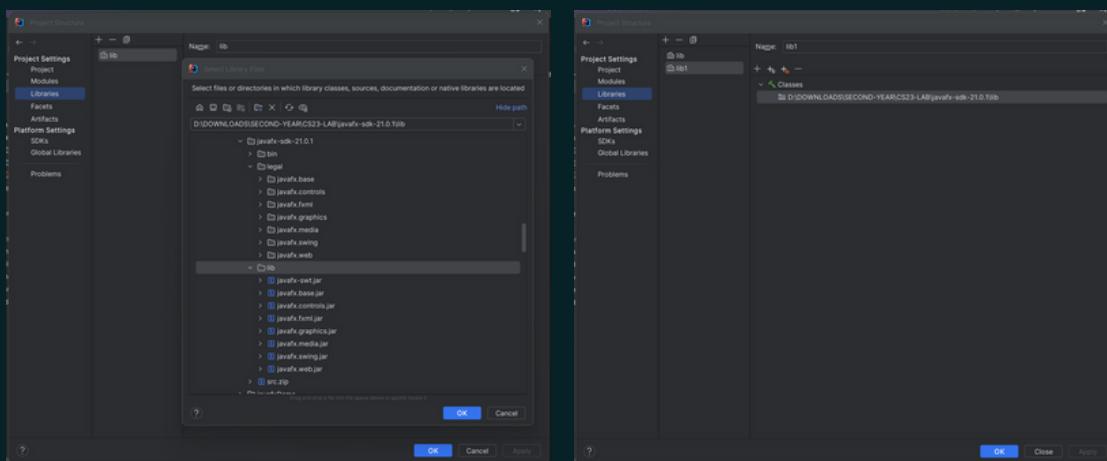
1. Load Maven

Upon opening the project, you will see a small window on the bottom-right of the screen indicating that Maven build scripts have been found. Make sure to click **Load Maven Project** to ensure that the application will run.



2. Import JavaFX Library

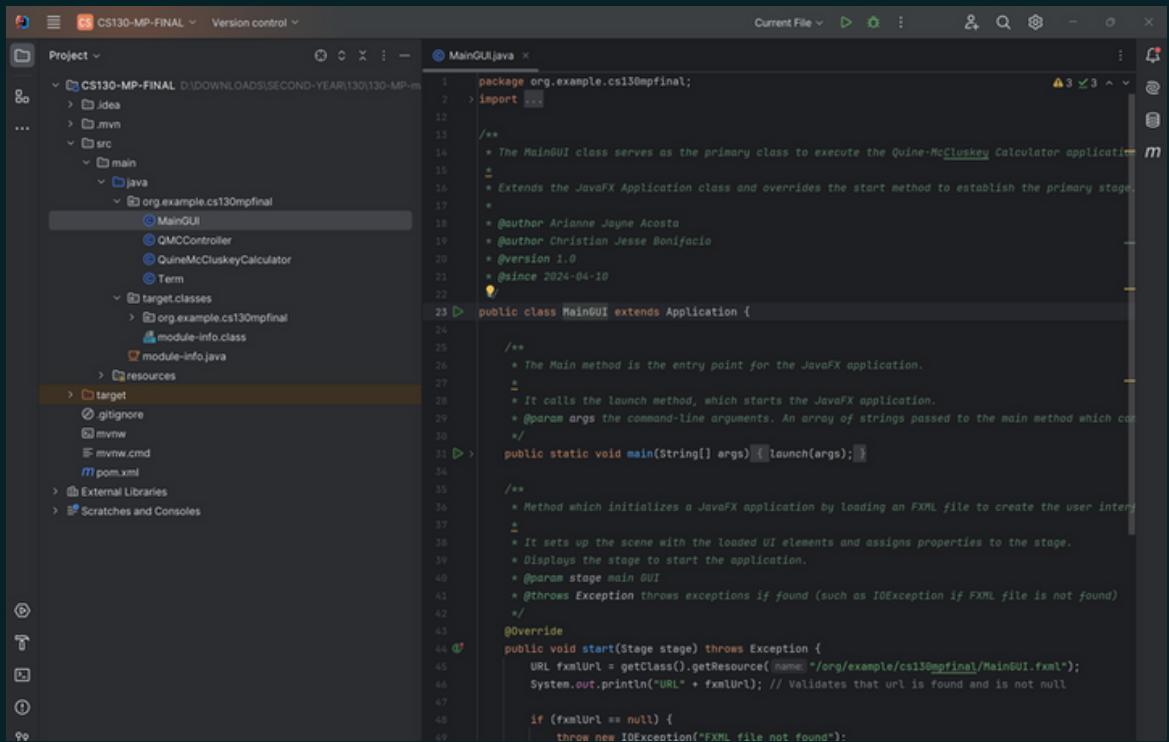
If JavaFX is not installed on your IDE, navigate to Project Structure and import the **lib** folder of your downloaded [JavaFX SDK](#) from Gluon. Click **OK**, then **Apply**.



PROJECT

Installed Project

After following all the steps, your Project must look like this.



The screenshot shows a Java project named "CS130-MP-FINAL" in an IDE. The project structure on the left includes a "src" folder containing "main" and "java" subfolders, which further contain "org.example.cs130mpfinal" and "MainGUI" respectively. Other files in "src/main/java" include "QMCController", "QuineMcCluskeyCalculator", and "Term". Below "src" are "target", "External Libraries", and "Scratches and Consoles". The right side of the interface is a code editor for "MainGUI.java". The code is as follows:

```
package org.example.cs130mpfinal;
import ...

/**
 * The MainGUI class serves as the primary class to execute the Quine-McCluskey Calculator application.
 */
public class MainGUI extends Application {

    /**
     * The Main method is the entry point for the JavaFX application.
     *
     * It calls the launch method, which starts the JavaFX application.
     * @param args the command-line arguments. An array of strings passed to the main method which contains the application's arguments.
     */
    public static void main(String[] args) {
        launch(args);
    }

    /**
     * Method which initializes a JavaFX application by loading an FXML file to create the user interface.
     *
     * It sets up the scene with the loaded UI elements and assigns properties to the stage.
     * Displays the stage to start the application.
     * @param stage main GUI
     * @throws Exception throws exceptions if found (such as IOException if FXML file is not found)
     */
    @Override
    public void start(Stage stage) throws Exception {
        URL fxmlUrl = getClass().getResource("/org/example/cs130mpfinal/MainGUI.fxml");
        System.out.println("URL" + fxmlUrl); // Validates that url is found and is not null

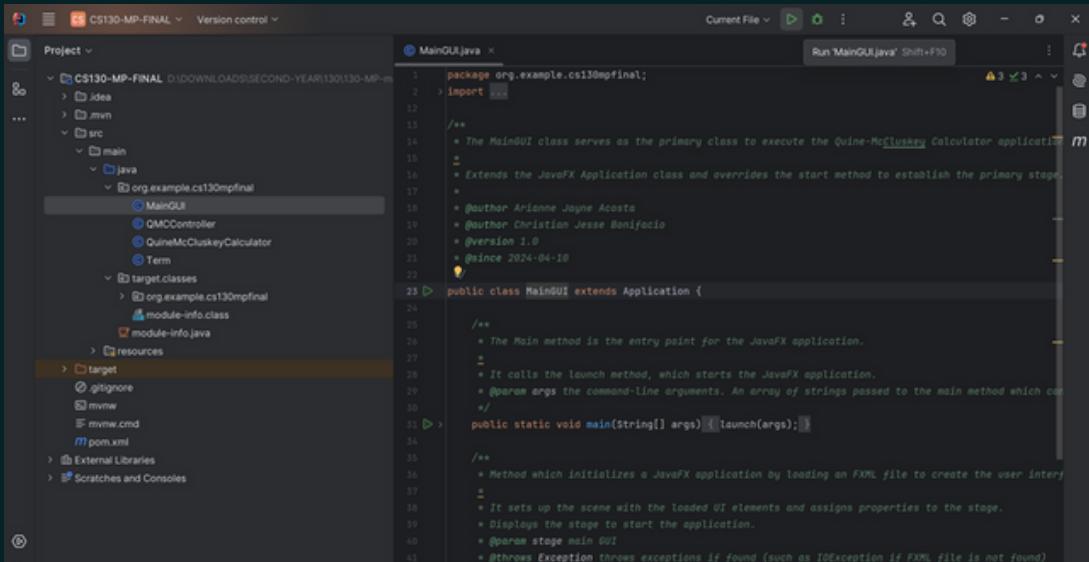
        if (fxmlUrl == null) {
            throw new IOException("FXML file not found");
        }
    }
}
```

INSTALLATION

RUNNING THE PROGRAM

1. MainGUI

After properly installing the project, navigate to the **MainGUI** class to run the program. On the upper right-hand corner of the screen, click the ➤ symbol to run the program.



The screenshot shows the IntelliJ IDEA IDE interface. The left sidebar displays the project structure for 'CS130-MP-FINAL'. The 'src' folder contains 'main' and 'org.example.cs130mpfinal' packages. 'MainGUI.java' is selected in the org.example.cs130mpfinal package. The main editor window shows the code for 'MainGUI.java'. The code is annotated with Javadoc-style comments explaining its purpose and methods. The 'Run' button in the top right corner is highlighted in yellow.

```
package org.example.cs130mpfinal;
import ...;

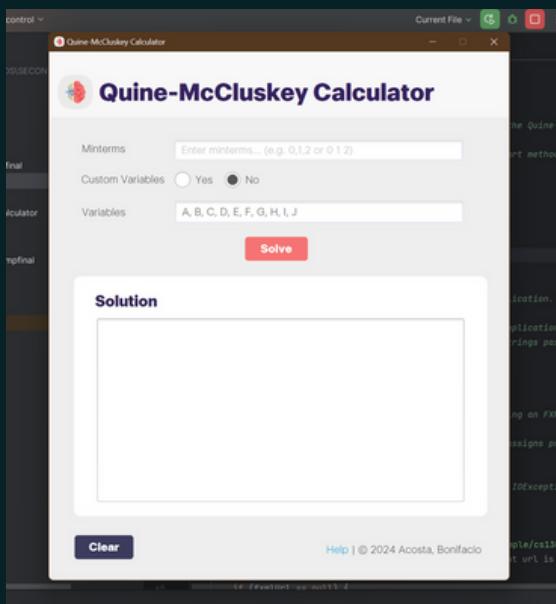
/**
 * The MainGUI class serves as the primary class to execute the Quine-McCluskey Calculator application.
 * Extends the JavaFX Application class and overrides the start method to establish the primary stage.
 *
 * @author Arianne Jayne Acosta
 * @author Christian Jesse Bonifacio
 * @version 1.0
 * @since 2024-04-10
 */
public class MainGUI extends Application {

    /**
     * The Main method is the entry point for the JavaFX application.
     *
     * It calls the launch method, which starts the JavaFX application.
     * @param args the command-line arguments. An array of strings passed to the main method which can
     *             be used to pass parameters to the application.
     */
    public static void main(String[] args) { launch(args); }

    /**
     * Method which initializes a JavaFX application by loading an FXML file to create the user interface.
     *
     * It sets up the scene with the loaded UI elements and assigns properties to the stage.
     * Displays the stage to start the application.
     * @param stage main GUI
     * @throws Exception throws exceptions if found (such as IOException if FXML file is not found)
     */
}
```

2. The GUI

The GUI should now appear on your screen.



RUNNING THE PROGRAM

USING THE CALCULATOR

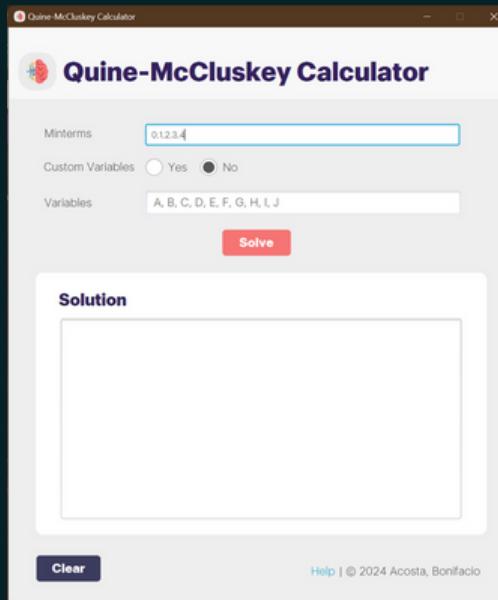
1. Enter minterms

Enter comma-separated or space-separated integer minterms in the given text-field. The inputted minterms must not exceed 1023, the maximum number of minterms for 10 variables. Below are examples of permissible inputs:

- ✓ 0,1,2,3,4
- ✓ 0, 1, 2, 3, 4
- ✓ 0 1 2 3 4

Any other inputs are not allowed and will prompt the appropriate *error windows*.

- ✗ 0.1.2.3,4
- ✗ 0,1,2,b,4



2. Choose your variables

Choose to use the default variables (A,B,C,D,E,F,G,H,I,J) or use your own custom variables by toggling between the radio buttons. Information about the variables used will appear in the Solution Panel once the solve button is clicked.

When inputting custom variables, the **variables must be comma or space separated** as well. 10 variables must be inputted.

<10: Program pads variables with default
>10: Program chooses only first 10 inputted.

Default Used

Opting for Custom

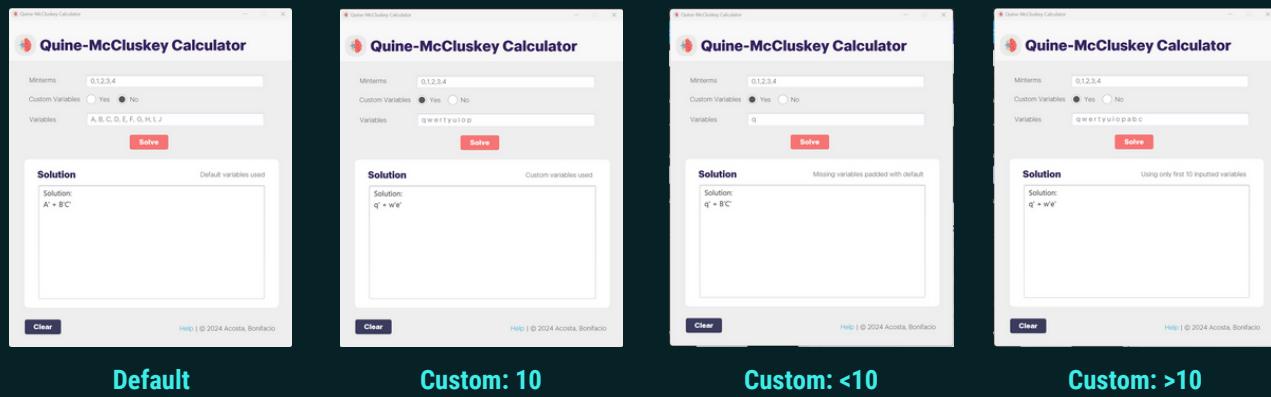
RUNNING THE PROGRAM

USING THE CALCULATOR

3. Solve

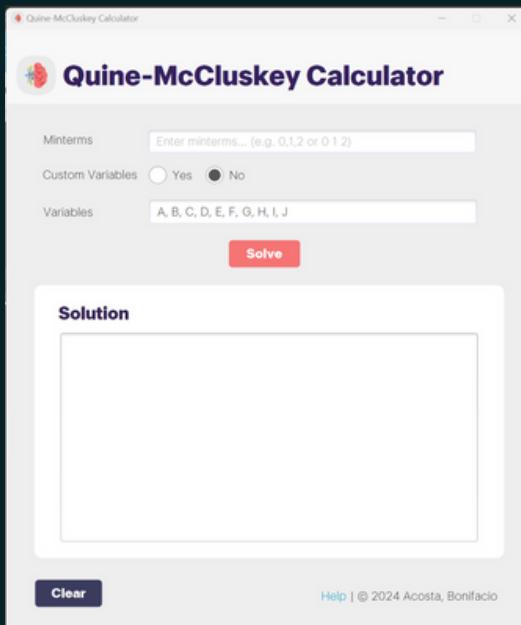
Click **Solve** and see the **simplified Boolean function** in the Solution text area!

Information about the variables used appear on the Solution Panel.



4. Clear

Solve for a different simplified Boolean function by clicking the **Clear** button to restore the Calculator to its default.



5. Exit

Exit by closing the Quine-McCluskey Calculator window.

RUNNING THE PROGRAM

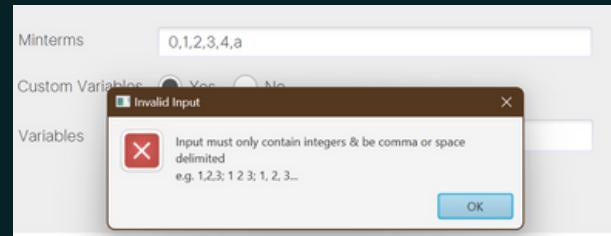
ALERTS

Invalid Inputs

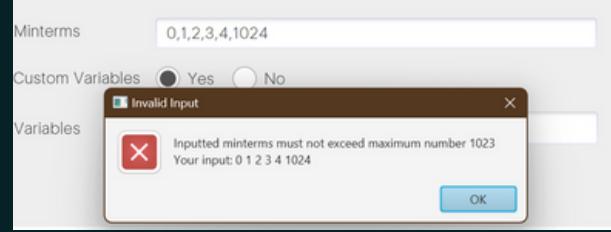
Invalid inputs passed in the text fields, when the **Solve** button is pressed, will prompt error windows, known as **Alerts**, to appear to inform the user. To address this issue, simply close the error window and retype the input, making sure it follows the prescribed format (comma or space delimited).

Below are examples of invalid input error windows.

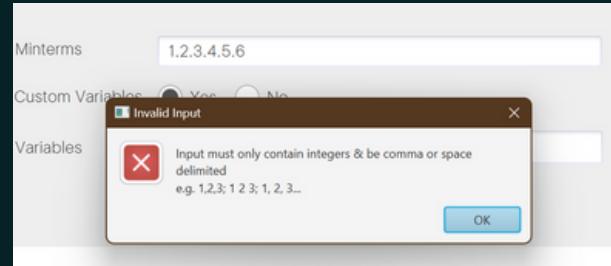
Non-integer input



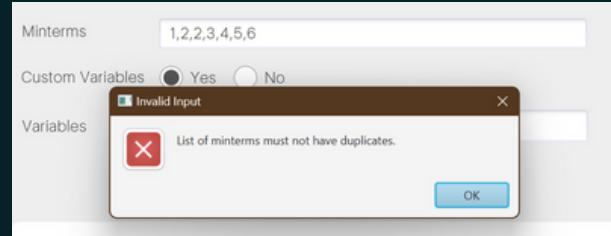
Input above maximum



Not comma or space separated



Duplicates found



SUPPORT

TROUBLESHOOTING

For further assistance or inquiries, please do not hesitate to reach out.

Arianne Jayne D. Acosta

EMAIL: adacosta3@up.edu.ph

FACEBOOK: <https://www.facebook.com/acostarianne>

Christian Jesse Q. Bonifacio

EMAIL: cqbonifacio1@up.edu.ph

FACEBOOK: <https://www.facebook.com/cj.bonifacioo?mibextid=LQQJ4d>

MEET OUR TEAM



ARIANNE JAYNE ACOSTA

CHRISTIAN JESSE
BONIFACIO

ALL RIGHTS RESERVED