Дослідження булевих функцій. Програмна реалізація і обчислення

Мета

Метою цієї роботи є розробка програмного застосунку, який зміг би будувати таблиці істинності для кожного можливого набору значень змінних, виводив би проміжні кроки обчислень, будував досконалі форми подання, був зрозумілим і легким у використанні, зберігав і відтворював історію виконання виразів.

Завдання

За допомогою сучасних інструментів та методів розробки створити програму яка відповідала б усім вимогам зазначеним в меті.

Актуальність

Обрана мною тема є актуальною оскільки булева логіка та булеві функції широко використовуються в криптографії (особливо для розробки алгоритмів з симетричними ключами, псевдовипадкові послідовності бітів), при проектуванні комп'ютерів (реалізація в електросхемах за допомогою логічних гейтів) та для побудови логічних форм функцій вибору та прийняття рішень (теорія соціального вибору для прикладу).

Аналіз подібних програм

Мною було проаналізовано деяку кількість онлайн програм-аналогів, реалізованих на різних мовах програмування. Всі вони виконували поставлене завдання втім, всі по різному. Деякі не мали проміжних кроків, в деяких був досить не зручний ввід і вивід, деякі не підтримували пріоритети операцій (що безсумнівно є досить важливою частиною для вирішення будь-якого виразу) чи можливість групування. Жоден з протестованих калькуляторів не зберігав історію, а один з них мав обмежений функціонал (решту функцій необхідно купляти). Такі результати аналізу надихнули мене на розробку власного додатку, який реалізовував би всі функції одразу.

Перший розділ

Булеві функції

У першому розділі курсової роботи я описав загальні теоретичні відомості про булеві функції, про спеціальні форми подання, про оператори.

В рамках доповіді хочу нагадати вам основні поняття і визначення з предметної області.

1) Булева функція (функція алгебри логіки) – функція виду $f(x_1, \dots, x_n)$, де область її значень $\{0,1\}$. Змінні $x_1, \dots, x_n$ також набувають двох значень – нуль та один.
2) Булеві функції можна подавати як за допомогою формул так і за допомогою таблиць істинності.
3) Будь яку булеву функцію можна єдиним способом подати в досконалій диз'юнктивній та кон'юнктивній формах.

На цьому слайді ви бачите оператори булевої логіки та відповідну таблицю істинності для кожного з них.

Зворотна польська нотацій

Детальніше хотілось би зупинитись на зворотній польській нотації.

В більшості випадків ми звикли використовувати інфіксну форму запису виразів, яка є не однозначною без наявності дужок. Запрограмувати обчислення виразу в такій формі досить складно, оскільки виконати всі розрахунки за один прохід зліва не право фактично неможливо.

Для вирішення цієї проблеми я скористався зворотною польською нотацією. Зліва на слайді ви бачите альтернативні форми запису. Як можете помітити в постфіксній оператор ставиться після операндів.

Для реалізації алгоритму перетворення інфіксної форми в постфіксну і подальшого обчислення використовуються структура даних стек, яка працює за принципом LIFO (last in, first out).

На слайді ви бачите морозиво яке представляє собою стек. Кулька яка була покладена останньою, вийде зі стеку першою.

Також ви можете помітити процес обрахунку виразу в такому записі. Поміщаємо в стек всі змінні до тих пір поки не зустрінемо оператор. Коли зустрінемо, виймаємо два останніх операнди зі стеку і виконуємо операцію, результат повертаємо в стек.

Другий розділ

У другому розділі курсової роботи описано програмну реалізація та інструменти використанні для досягнення мети.

На слайді ми можете бачити зображення, на якому проілюстровано архітектуру .NET Framework 4.6. Для написання програми я використовував найновішу, 5 версію платформи разом з такими компонентами як:

- Середовище розробки Visual Studio 19 – яке має багато інструментів для розробки (для пошуку помилок в коді програми чи проектування користувацького інтерфейсу).
- Об’єктно орієнтовна мова програмування C#, що підтримується платформою
- WPF або Windows Presentation Foundation, компонент .NET Framework, призначений для створення користувацького інтерфейсу програм для настільних систем.
- LINQ або Language Integrated Query, інший компонент, що дозволяє звертатись до масивів чи списків як до бази даних.

Демонстрація

1) Ресайз
2) Допомога
3) Елементарні приклади
4) Константи
5) Виняткові ситуації
6) Історія
7) Вирази що можна перевірити
8) Ввід з клавіатури

На відео ви бачите запущений застосунок в роботі. Перш за все ви можете побачити дизайн і масштабованість інтерфейсу.

Далі я відкриваю вікно додаткової інформації, де описано процес взаємодії з програмою та також відомості про розробника.

На відео ви бачите демонстрацію на елементарних прикладах. Як можна помітити програма працює як з українськими літерами так і з англійськими.

Програма додає кон’юнкції між змінними автоматично. Пізніше ця функція буде продемонстрована на більш складному прикладі.

Кнопка CA очищує поле вводу і таблицю.

Програма підтримує константи 1 і 0. Диз’юнкція хоча б з одною одиницею буде завжди давати 1, кон’юнкція якщо є хоча б один 0 буде давати завжди 0.

Зараз я демонструю обробку виняткових ситуацій, передбачених мною. Окрім них, програма буде перехоплювати всі винятки що виникають під час виконання, завдяки чому програма не буде завершуватись аварійно.

Як ви можете помітити історія виразів наповнюється в зворотному порядку вводу (нові згори, старі внизу). Для демонстрації я підготував декілька більш складних прикладів.

Цей приклад було наведено в підручнику, тому перевірити правильність обчислень не складно. Програма підтримує групування і виводить проміжні кроки обчислень.

Цей приклад також взятий з підручника для демонстрації правильності побудови досконалих форм. Їх ви могли помітити знизу під таблицею.

Всі вирази завжди переписуються з використанням більш зручного і зрозумілого набору операторів і зараз я це продемонструю на прикладі минулої функції.

Також можемо помітити що в історії дублікат функції не створився, через те що така функція вже була в тому списку.

На останок я демонструю роботу програми з більш складним виразом. Хоча програма розуміє пріоритет операцій, коли я будую проміжні кроки з постфіксної форми, я додаю дужки аби усунути неоднозначність.

Кінець

Підводячи підсумок можу сказати що я досяг поставленої мети.

Весь описаний функціонал я зміг реалізувати і показати. Нажаль мені не вдалось динамічно змінювати розмір шрифту в таблиці залежно від розміру екрану через особливості роботи WPF.

Програму можуть використовувати як студенти так і викладачі для демонстрації різних прикладів чи перевірки правильності обчислень. Вже для виразів з 4 змінними існує 16 наборів значень, через що існує досить велика вірогідність помилитись.

В майбутньому можна додати алгоритми мінімізації, перехід до алгебри Жегалкіна, або навіть якісь інші розділи дискретної математики (наприклад теорія графів).

На цьому я завершую свою доповідь. Дякую за увагу, буду радий відповісти на ваші питання.