ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ ІНФОРМАТИКИ

Допускається до захисту

Завідувач кафедри _________О.О. Ємець
(підпис)

«______»__________________2019 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО ДИПЛОМНОЇ РОБОТИ

на тему
ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ КАЛЬКУЛЯТОРА З ТЕМИ «МІНІМІЗАЦІЯ ЛОГІЧНИХ ФУНКЦІЙ З ДОПОМОГОЮ КАРТ КАРНО» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «МАТЕМАТИЧНА ЛОГІКА»

зі спеціальності 122 «Комп'ютерні науки»

Виконавець роботи Кравцов Валентин Олександрович
___________ «___»______2019р.
(підпис)
Науковий керівник к.ф.-м.н., доц. Черненко Оксана Олексіївна
___________ «___»______2019р.
(підпис)

ПОЛТАВА 2019р.

ЗМІСТ

ВСТУП 3

1. ПОСТАНОВКА ЗАДАЧІ 5

2. ІНФОРМАЦІЙНИЙ ОГЛЯД 7

2.1. Алгебра логіки 7

2.2. Огляд сервісів для розв’язування математичних задач 12

3. ТЕОРЕТИЧНА ЧАСТИНА 21

3.1. Застосування карт Карно для мінімізації логічних функцій 21

3.2. Приклади застосування карт Карно для мінімізації логічних функцій 34

3.3. Алгоритм роботи калькулятора 36

3.4. Обґрунтування вибору програмних засобів 40

4. ПРАКТИЧНА ЧАСТИНА 43

4.1. Опис процесу програмної реалізації 43

4.2. Опис програми 49

4.3. Інструкція по використанню тренажеру 52

ВИСНОВКИ 57

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 59

ДОДАТОК А. КОД ПРОГРАМИ 61

ВСТУП

Останнім часом широкого розповсюдження в мережі Інтернет набули онлайн (Інтернет) калькулятори для розв’язання задач широкого спектру. Все більшої популярності набувають калькулятори, призначені для розв’язання типових задач елементарної математики.

Використовуючи калькулятор, ви зможете розв'язати задачу або перевірити правильність свого розв'язку. Для розв'язання задачі досить ввести данні задачі, програма самостійно виконає всі розрахунки і покаже відповідь з покроковим розв'язком в розгорнутій формі, що допоможе краще розібратися в незрозумілому матеріалі. А також знайти та виправити помилки, або переконатись в правильності свого розв'язку.

Мета роботи – розробка калькулятора з теми «Мінімізація логічних функцій з допомогою карт Карно» дистанційного навчального курсу «Математична логіка».

Об’єктом розробки є процес дистанційного навчання математичним дисциплінам.

Предмет розробки — програмний продукт, що реалізує калькулятор для розв’язування задач мінімізації логічних функцій з допомогою карт Карно.

Методи розробки – методика мінімізації логічних функцій з використанням карт Карно.

Для програмної реалізації використано середовище розробки програм NetBeans IDE та об’єктно-орієнтована мова програмування Java.

В програмі реалізовано можливість переключення між українською та англійською мовами.

Тренажер готовий до використання в дистанційному курсі «Математична логіка».

Робота складається з чотирьох розділів. У першому розділі розглянуто постановку задачі. У другому розділі описано основні поняття алгебри логіки та огляд сервісів для розв’язування математичних задач. У третьому розділі представлено застосування карт Карно для мінімізації логічних функцій, наведено приклади застосування карт Карно, описано алгоритм роботи калькулятора та обґрунтування вибору програмних засобів. У четвертому розділі – описано процес програмної реалізації, саму програму і інструкцію по використанню.

Обсяг пояснювальної записки: 94 стор., в т.ч. основна частина - 55 стор., джерела - 9 назв.

1. ПОСТАНОВКА ЗАДАЧІ

Потрібно скласти алгоритм роботи калькулятора і розробити програмний продукт як складову дистанційного курсу «Математична логіка».

Розглянемо основні завдання роботи:

- розглянути теоретичні відомості про мінімізацію логічних функцій з допомогою карт Карно;
- розглянути специфіку застосування карт Карно;
- розробити алгоритм калькулятора із застосування карт Карно для мінімізації логічних функцій;
- обргунтувати вибір програмного забезпечення.
- розробити калькулятор;
- описати процес прогрманої реалізації;
- надати опис програми та інструкцію користувача.

Калькулятор повинен містити поле для введення логічного виразу, кнопки із змінними і операторами. Слід також розробити можливість введення змінних з клавіатури, а оператори мають вводитися лише натисненням на відповідну кнопку калькулятора.

У розв’язку повинно описуватися всі виконані дії при мінімізації введеної логічної функції. При побудові таблиці істинності функції передбачити можливість виведення таблиць істинності використаних операцій.

Якщо під час обрахунків виникає помилка, то необхідно вивести повідомлення про дану помилку і її причину.

За допомогою карт Карно легко вирішуються задачі мінімізації функцій з кількістю змінних до шести включно. При більшій кількості змінних пошук мінімальних форм запису функцій значно ускладнюється і наочність карт Карно втрачається.

Важливим завданням є реалізація зміни мови калькулятора на українську або англійську.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1. Алгебра логіки

Проектування і аналіз логічних схем ЕОМ ведеться за допомогою спеціального розділу математики - алгебри логіки. В алгебрі логіки можна виділити три основні логічні функції: "НЕ" (заперечення), "І" (сполучення), "АБО" (диз'юнкція).

Для створення будь-якого логічного пристрою необхідно визначити залежність кожної з вихідних змінних від діючих вхідних змінних така залежність називається перемикальної функцією або функцією алгебри логіки.

Функція алгебри логіки називається повністю визначеною, якщо задані всі 2^n її значення, де n - число вихідних змінних.

Якщо визначені не всі значення, функція називається частково визначеною.

Пристрій називається логічним, якщо його стан описується за допомогою функції алгебри логіки.

Для представлення функції алгебри логіки (ФАЛ) використовується наступні способи:

- словесний опис - це форма, яка використовується на початковому етапі проектування і має умовне уявлення;
- опис функції алгебри логіки у вигляді таблиці істинності;
- опис функції алгебри логіки у вигляді алгебраїчного виразу – використовується дві алгебраїчні форми ФАЛ: ДНФ і КНФ.

ДНФ - диз'юнктивна нормальна форма. ДНФ виходить з таблиці істинності за наступним алгоритмом або правилом:

1) в таблиці вибираються ті рядки змінних для яких функція на виході дорівнює 1;

2) для кожного рядка змінних записується логічний вираз; причому змінні, які дорівнюють 0 записуються з інверсією.
3) отриманий вираз логічно підсумовується.

ДНФ називається досконалою, якщо всі змінні мають однаковий ранг або порядок, тобто в кожен вираз обов'язково повинні включатися всі змінні в прямому або інверсному вигляді.

КНФ - кон'юнктивна нормальна форма. КНФ може бути отримана з таблиці істинності за наступним алгоритмом:

1) вибираємо набори змінних для яких функція на виході дорівнює 0;
2) для кожного набору змінних записується елементарна логічна сума, причому змінні, що дорівнюють 1 записуються з інверсією;
3) логічно перемножуються отримані суми.

КНФ називається досконалою, якщо всі змінні мають однаковий ранг.

Всі операції алгебри логіки визначаються таблицями істинності значень. Таблиця істинності визначає результат виконання операції для всіх можливих логічних значень вихідних висловлювань. Кількість варіантів, що відображають результат застосування операцій, буде залежати від кількості висловлювань в логічному вираженні. Якщо число висловлювань в логічному вираженні N, то таблиця істинності буде містити 2^N рядків, так як існує 2^N різних комбінацій можливих значень аргументів.

Операція НЕ - логічне заперечення (інверсія).

Логічна операція НЕ застосовується до одного аргументу, в якості якого може бути і просте, і складне логічне вираз. Результатом операції НЕ є наступне:

- якщо вихідний вираз істинний, то результат його заперечення буде хибним;

- якщо вихідний вираз помилковий, то результат його заперечення буде істинним.

Для операції заперечення НЕ прийнято такі умовні позначення: НЕ А, Ā, not A, ¬A,! A

Результат операції заперечення НЕ визначається наступною таблицею істинності:

Таблиця 2.1 – Таблиця істинності операції «НЕ»

A	не A
0	1
1	0

Результат операції заперечення правдивий, коли вихідне висловлення помилкове, і навпаки.

Операція АБО - логічне додавання (диз'юнкція, об'єднання).

Логічна операція АБО виконує функцію об'єднання двох висловлювань, в якості яких може бути і простий, і складний логічний вираз. Висловлювання, що є вихідними для логічної операції, називають аргументами. Результатом операції АБО є вираз, який буде істинним тоді і тільки тоді, коли істинно буде хоча б одне з вихідних виразів.

Застосовувані позначення: A або B, A V B, A or B, A || B.

Результат операції АБО визначається наступною таблицею істинності:

Таблиця 2.2 – Таблиця істинності операції «АБО»

A	B	A або B
0	0	0
0	1	1
1	0	1
1	1	1

Результат операції АБО правдивий, коли істинно А, або істинно В, або істинно і А і В одночасно, і хибна тоді, коли аргументи А і В - помилкові.

Операція I - логічне множення (кон'юнкція).

Логічна операція I виконує функцію перетину двох висловлювань (аргументів), в якості яких може бути і простий, і складний логічний вираз. Результатом операції I є вираз, який буде істинним тоді і тільки тоді, коли істинні обидва вихідних вираження.

Застосовувані позначення: А і В, А Λ В, А & В, А and В.

Результат операції I визначається наступною таблицею істинності:

Таблиця 2.3 – Таблиця істинності операції «I»

A	B	A i B
0	0	0
0	1	0
1	0	0
1	1	1

Результат операції I правдивий тоді і тільки тоді, коли істинні одночасно висловлювання А і В, і хибна у всіх інших випадках.

Операція «ЯКЩО-ТО» - логічне слідування (імплікація).

Ця операція пов'язує два простих логічних вирази, з яких перше є умовою, а друге - наслідком з цього умови.

Застосовувані позначення: якщо А, то В; А тягне В; if A then В; А → В.

Таблиця істинності:

Таблиця 2.4 – Таблиця істинності операції «ЯКЩО-ТО»

A	B	$A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Результат операції слідування (імплікації) хибний тільки тоді, коли передумова A істинна, а висновок B (наслідок) помилковий.

Операція «A тоді і тільки тоді, коли B» (еквівалентність, рівнозначність).

Застосовується позначення: $A \leftrightarrow B$, $A \sim B$.

Таблиця істинності:

Таблиця 2.5 – Таблиця істинності операції «A тоді і тільки тоді, коли B»

A	B	$A \leftrightarrow B$
0	0	1
0	1	0
1	0	0
1	1	1

Результат операції еквівалентність правдивий тільки тоді, коли A і B одночасно істинні або одночасно хибні.

Операція «Додавання за модулем 2» (XOR, "який виключає чи", сувора диз'юнкція)

Застосовується позначення: A XOR B, $A \oplus B$.

Таблиця істинності:

Таблиця 2.6 – Таблиця істинності операції «Додавання за модулем 2»

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Результат операції еквівалентність правдивий тільки тоді, коли А істинне, а В хибне і навпаки [4].

Пріоритет логічних операцій.

Пріоритет логічних операцій:

1) Дії в дужках;
2) Інверсія;
3) Кон'юнкція (&);
4) Диз'юнкція (V), виключає Або (XOR), сума по модулю 2;
5) Імплікація (→);
6) Еквівалентність (↔).

2.2. Огляд сервісів для розв’язування математичних задач

Розглянемо доступні онлайн-сервіси для розв’язування математичних задача, зокрема, мінімізації логічних функцій із застосуванням карт Карно.

Сервіс «Онлайн инструменты по математической логике» має вигляд (рис. 2.1) [5]:

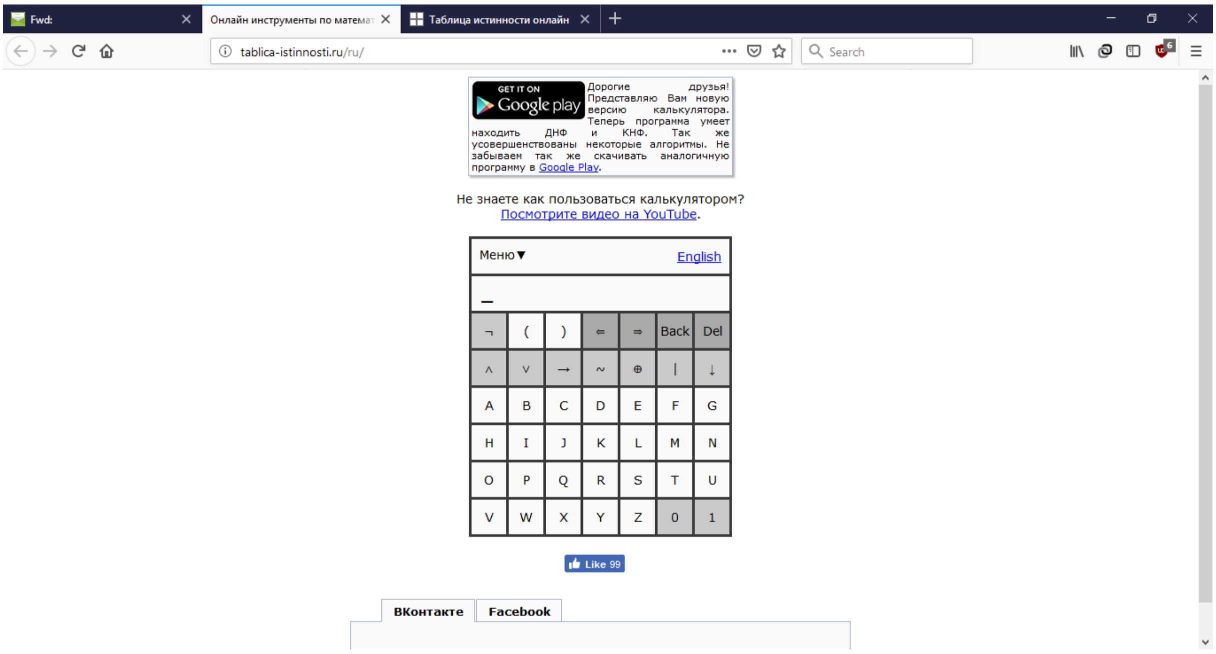


Рисунок 2.1 – Онлайн-сервіс «Онлайн инструменты по математической логике»

Даний калькулятор представляє користувачу наступні методи:

- таблиця істинності;
- поліном Жегалкіна;
- ДНФ, КНФ;
- ДДНФ, ДКНФ;
- карти Карно;
- логічна схема;
- мінімізувати функцію;
- метод Мак-Класкі.

Змінні і «дужки» вводяться в самому калькуляторі або з клавіатури. Оператори ж можна ввести лише натискаючи відповідні кнопки калькулятора (рис. 2.2).

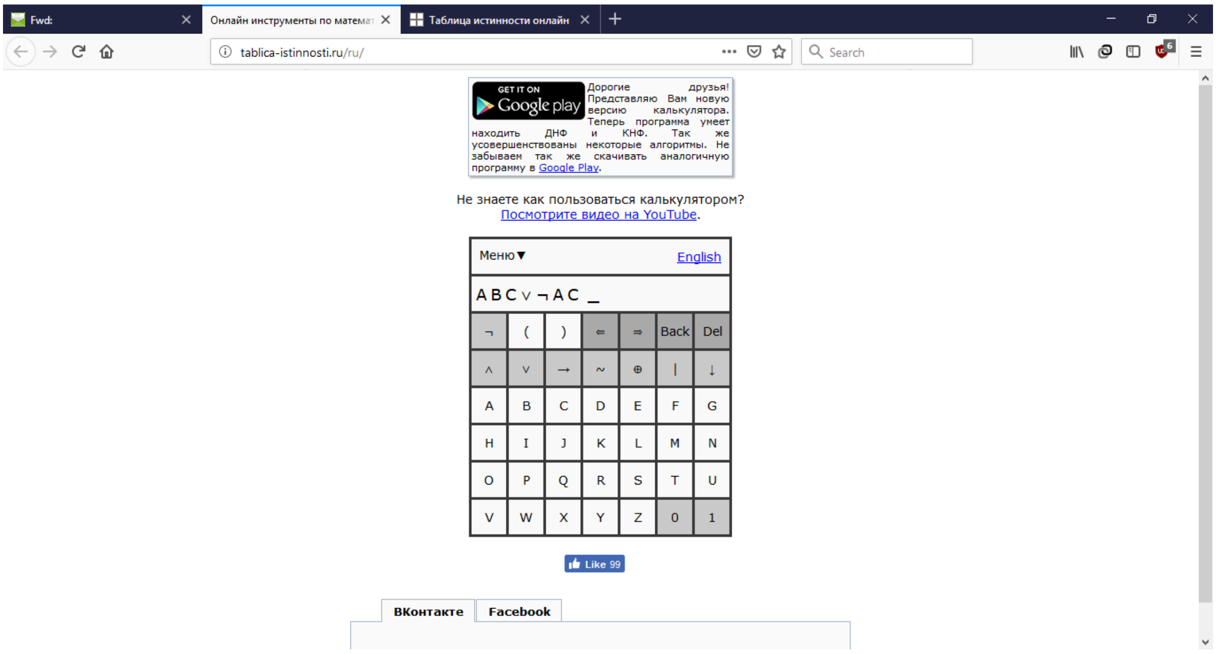
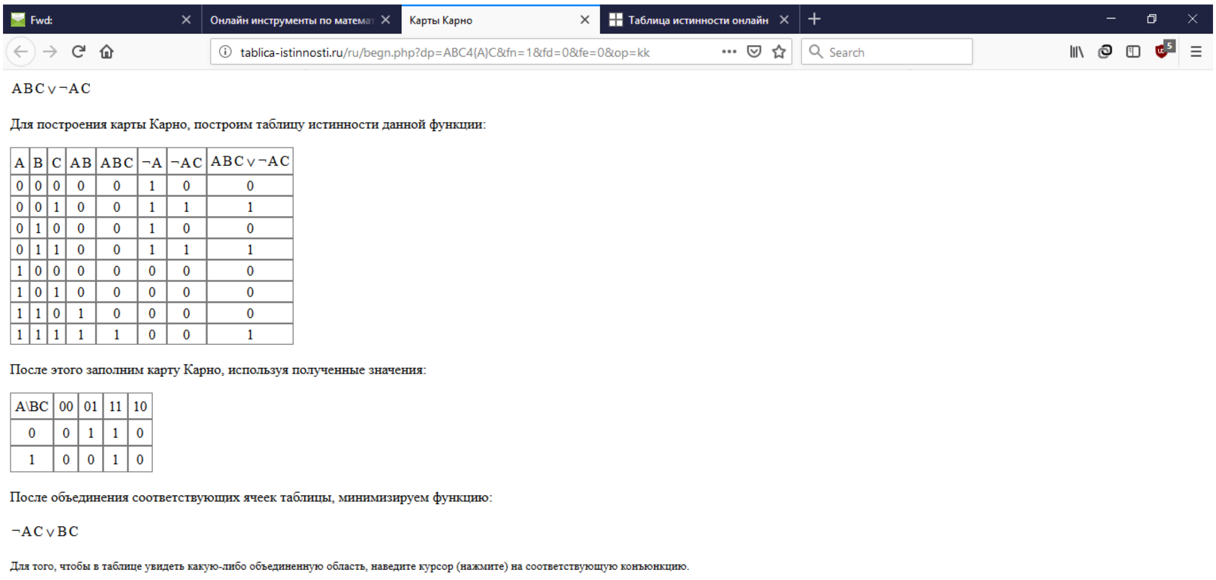


Рисунок 2.2 – Введення логічного виразу

Після цього слід вибрати потрібний метод (рис. 2.3) і виведеться результат обрахунків (рис. 2.4):

- таблиця істинності введеної функції;
- побудована карта Карно;
- мінімізована функція.

Рисунок 2.3 – Вибір методу



ABC∨¬AC

Для построения карты Карно, построим таблицу истинности данной функции:

A	B	C	AB	ABC	¬A	¬AC	ABC∨¬AC
0	0	0	0	0	1	0	0
0	0	1	0	0	1	1	1
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	1	0	0	0	0
1	1	1	1	1	0	0	1

После этого заполним карту Карно, используя полученные значения:

A\BC	00	01	11	10
0	0	1	1	0
1	0	0	1	0

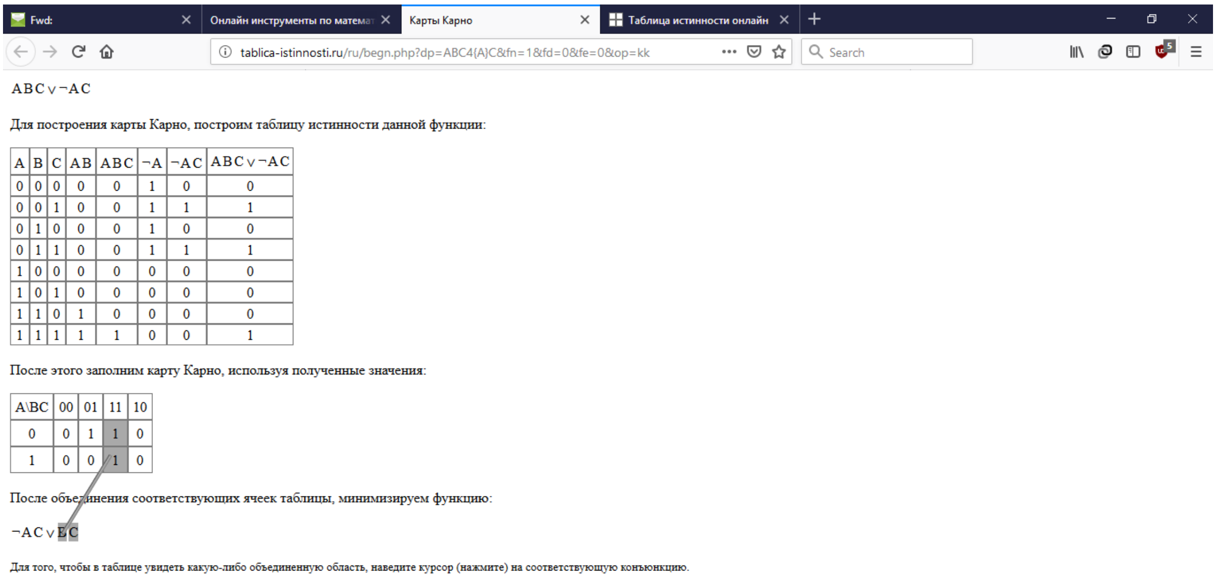
После объединения соответствующих ячеек таблицы, минимизируем функцию:

¬AC∨BC

Для того, чтобы в таблице увидеть какую-либо объединенную область, наведите курсор (нажмите) на соответствующую коньюнкцию.

Рисунок 2.4 – Отриманий результат

Щоб подивитися як отримано результат, необхідно навести курсор на одну з кон’юнкцій (рис. 2.5).



ABC∨¬AC

Для построения карты Карно, построим таблицу истинности данной функции:

A	B	C	AB	ABC	¬A	¬AC	ABC∨¬AC
0	0	0	0	0	1	0	0
0	0	1	0	0	1	1	1
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	1	0	0	0	0
1	1	1	1	1	0	0	1

После этого заполним карту Карно, используя полученные значения:

A\BC	00	01	11	10
0	0	1	1	0
1	0	0	1	0

После объединения соответствующих ячеек таблицы, минимизируем функцию:

¬AC∨BC

Для того, чтобы в таблице увидеть какую-либо объединенную область, наведите курсор (нажмите) на соответствующую коньюнкцию.

Рисунок 2.5 – Об’єднання клітин у таблиці

Інший калькулятор «Таблица истинности» онлайн-сервісу «math.semestr.ru» передбачає введення виразу вручну, тому наводиться список операторів (рис. 2.6), поле для вводу і вибір методів (рис. 2.7) [6]:

- виведення проміжних таблиць для таблиці істинності;
- побудова ДДНФ;
- побудова ДКНФ;
- побудова полінома Жегалкіна;
- побудова карти Вейча-Карно;
- мінімізація булевої функції.

Також можна перевірити правила введення логічної функції (рис. 2.8).

ИНСТРУКЦИЯ. При вводе с клавиатуры используйте следующие обозначения:

Клавиша	Оператор	
!	¬	Отрицание (НЕ)
|	|	Штрих Шеффера (И-НЕ)
#	↓	Стрелка Пирса (ИЛИ-НЕ)
*	&	Конъюнкция (И)
+	v	Дизъюнкция (ИЛИ)
^	⊕	Исключающее ИЛИ, сумма по модулю 2 (XOR)
@	→	Импликация (ЕСЛИ-ТО)
%	←	Обратная импликация
=	≡ (~, ↔)	Эквивалентность (РАВНО)

Рисунок 2.6 – Список операторів

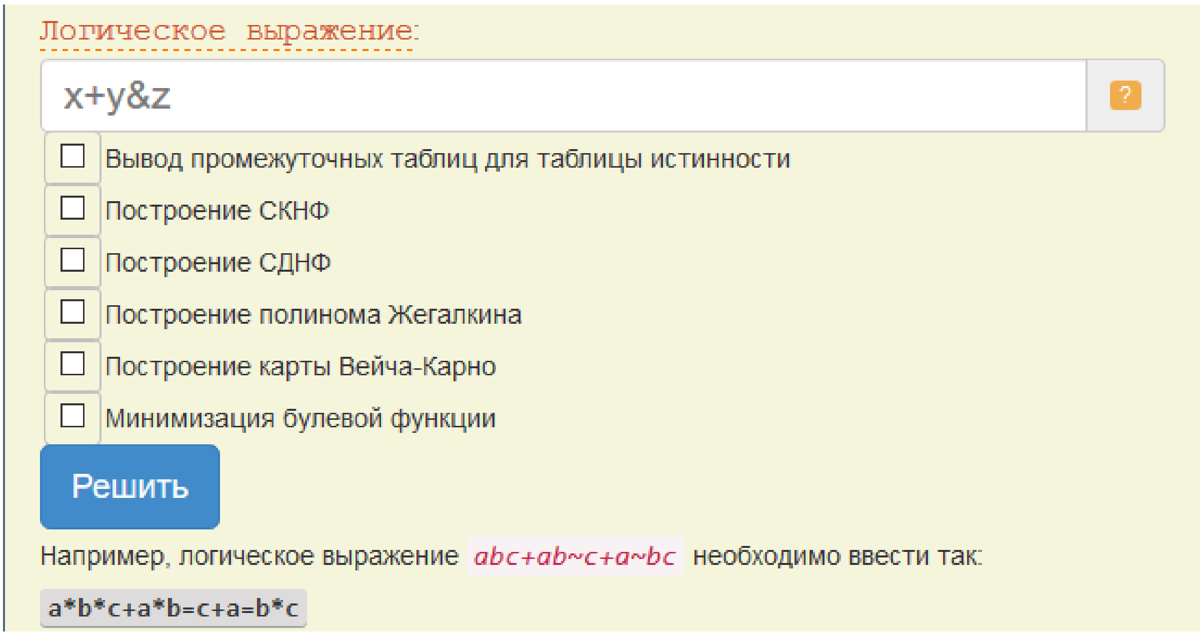


Рисунок 2.7 – Поле для введення виразу

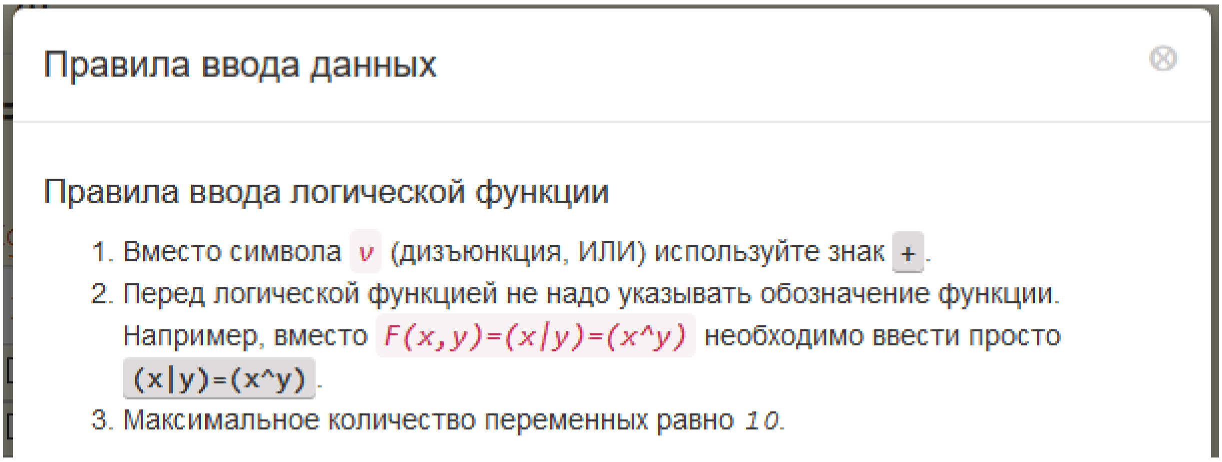


Рисунок 2.8 – Правила введення даних

Для мінімізації логічної функції із використанням карт Карно потрібно ввести вираз, встановити необхідні перемикачі та натиснути кнопку «Решить» (рис. 2.9).

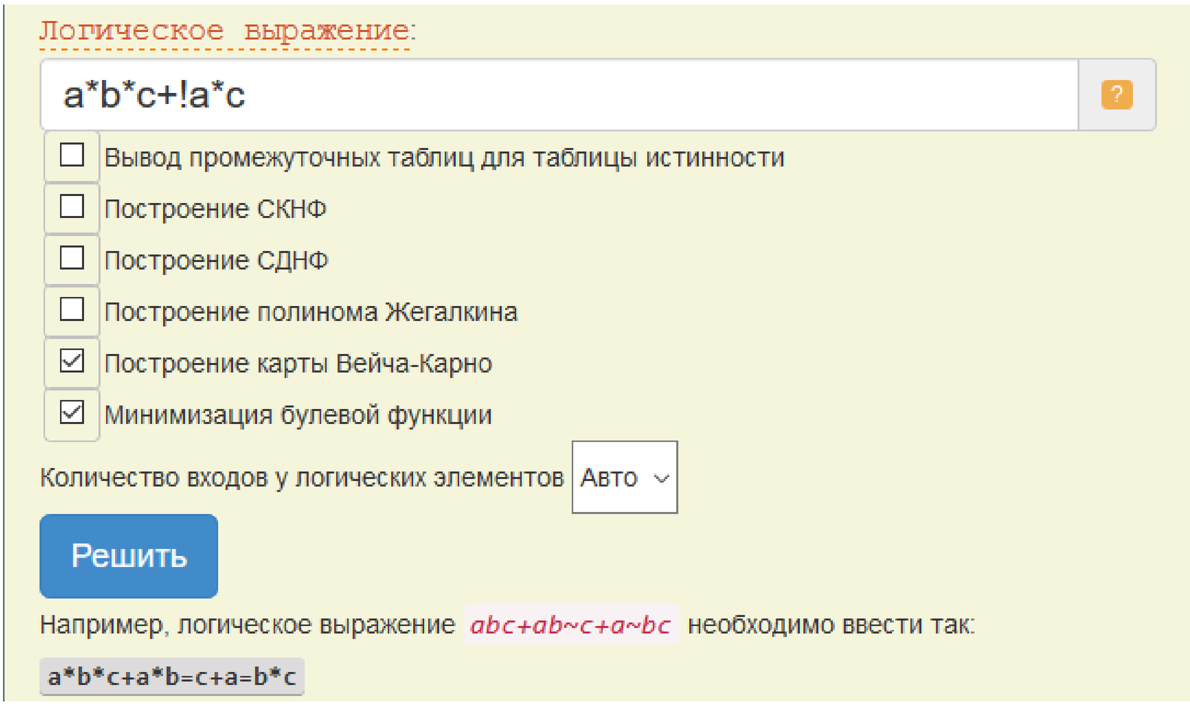


Рисунок 2.9 – Приклад заповнення даних

В результаті виводиться

- таблиця істинності функції (рис. 2.10);
- таблиці істинності використаних логічних операцій (рис. 2.11);
- побудована карта Карно (рис. 2.12);
- мінімізована функція (рис. 2.13).

a*b*cv¬a*c

a	b	c	a&b	(a&b)&c	¬a	(¬a)&c	((a&b)&c)v((¬a)&c)
0	0	0	0	0	1	0	0
0	0	1	0	0	1	1	1
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	1	0	0	0	0
1	1	1	1	1	0	0	1

Рисунок 2.10 – Таблиця істинності введеної функції

При решении были использованы таблицы истинности следующих операций.

Операция ИЛИ — логическое сложение (дизъюнкция, объединение)

x	y	x v y
0	0	0
0	1	1
1	0	1
1	1	1

Операция И — логическое умножение (конъюнкция)

x	y	x & y
0	0	0
0	1	0
1	0	0
1	1	1

Рисунок 2.11 – Таблиці істинності використаних логічних операцій

Метод диаграмм Вейча.

Метод позволяет быстро получать минимальные ДНФ булевой функции f небольшого числа переменных. В основе метода лежит задание булевых функций диаграммами некоторого специального вида, получившими название диаграмм Вейча.

	b	b	!b	!b
	!c	c	c	!c
a	0	1	0	0
!a	0	1	1	0

Рисунок 2.12 – Побудована карта Карно

$f = \overline{a}c \vee bc$

Решение было получено и оформлено с помощью сервиса:

Таблица истинности

Вместе с этой задачей решают также:

Умножение двоичных чисел

Обратный и дополнительный коды двоичных чисел

Формат числа с плавающей запятой

Перевод чисел онлайн

Сложение двоичных чисел

Рисунок 2.13 – Мінімізована функція

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1. Застосування карт Карно для мінімізації логічних функцій

Карта Карно – один з графічних способів подання логічних функцій (рис. 3.1).

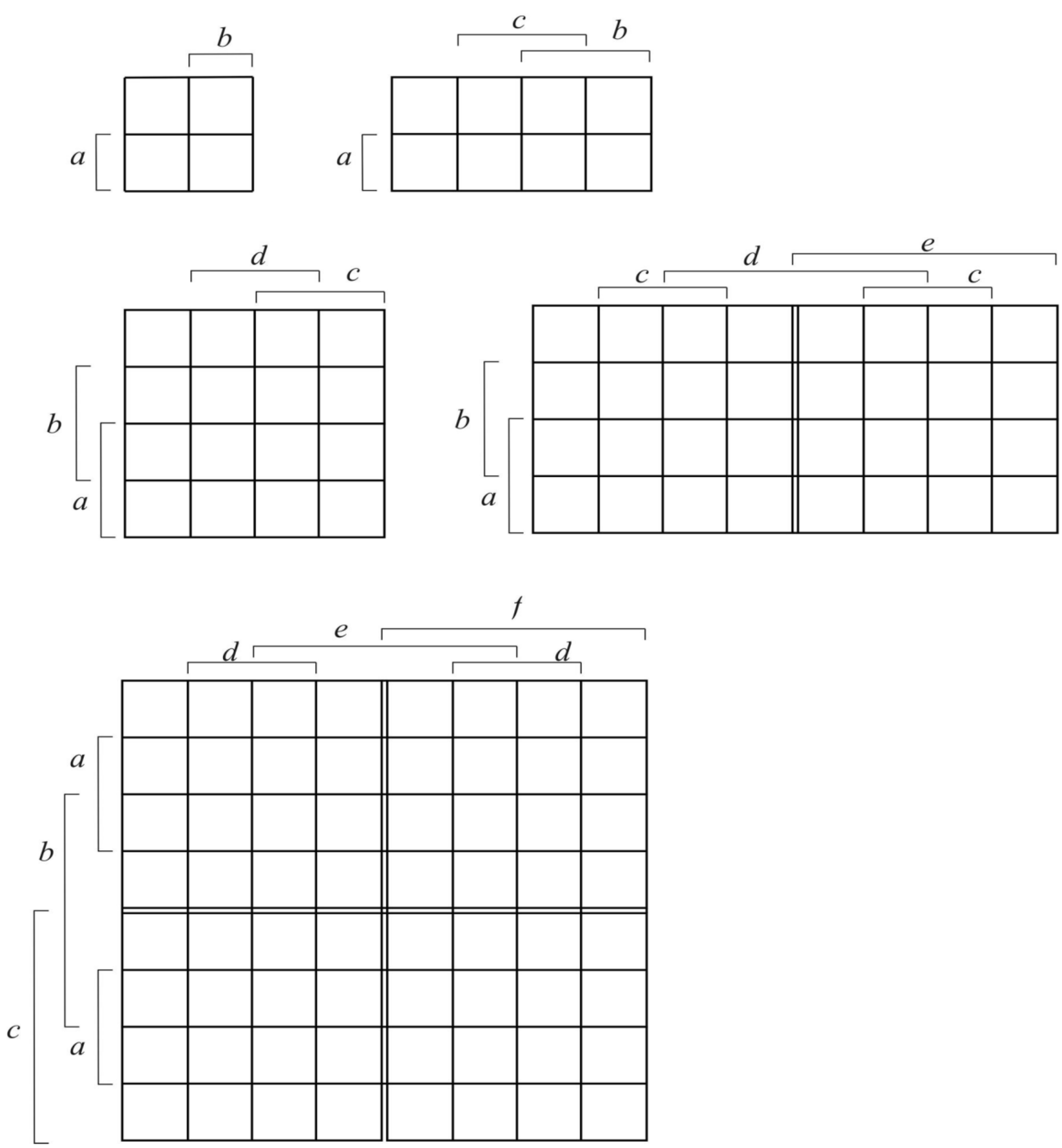


Рисунок 3.1 – Карти Карно для функцій двох, трьох, чотирьох, п’ятьох, шістьох змінних

Для функції n змінних вона складається з 2^n клітинок, причому кожна клітинка відповідає певному набору змінних.

Вхідні змінні розміщуються із зовнішніх сторін карти проти її рядків та стовпців. Значення вхідної змінної стосується усіх клітинок у рядку або стовпці і дорівнює одиниці, якщо проти рядка або стовпця є дужка з позначенням цієї змінної. Для решти рядків і стовпців значення змінної дорівнює нулю.

Дужки з позначеннями вхідних змінних треба розміщувати так, щоб кожна клітинка карти відповідала єдиному набору змінних. При цьому кожна змінна обов’язково поділяє площину карти на дві рівні частини, тобто одна половина клітинок відповідає значенню вхідної змінної, що дорівнює одиниці, а друга – нулю.

У клітинках карти Карно записується те значення функції, якого вона набуває, якщо набори вхідних змінних відповідають цим клітинкам. Розглянемо процес заповнення карти Карно для функції, заданої таблицею істинності (див. табл. 3.1). Ця функція є функцією трьох змінних.

Таблиця 3.1 – Таблиця істинності

a	b	c	ab	abc
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1

Відповідна карта Карно складається з $2^n = 8$ клітинок (рис. 3.2, а). Клітинки карти пронумеровано для зручності пояснення процесу заповнення.

Функція *f* дорівнює одиниці для першого, другого, шостого, сьомого та восьмого наборів вхідних змінних. Першому набору змінних $\left(a=0,b=0,c=0\right)$ відповідає перша клітинка карти Карно, другому набору $\left(a=0,b=0,c=1\right)$ – друга, шостому $\left(a=1,b=0,c=1\right)$ – шоста, сьомому $\left(a=1,b=1,c=0\right)$ – восьма, восьмому $\left(a=1,b=1,c=1\right)$ – сьома. У першій, другій, шостій, сьомій і восьмій клітинках карти Карно ставимо одиниці, у решті клітинок – нулі. Знизу під картою записуємо позначення функції *f*. На цьому заповнення карти закінчується.

Карту Карно можна скласти не тільки за таблицею істинності, але й за алгебричним виразом функції, заданої у ДНФ або КНФ. Якщо функцію задано у ДНФ, то її можна розгорнути у ДДНФ та заповнити одиницями клітинки карти, що відповідають конституентам одиниці. Проте для заповнення карти Карно не обов’язково розгортати функцію у ДДНФ, можна заповнити одиницями зразу групи клітинок, що відповідають елементарним кон’юнкціям. Нехай, наприклад, для функції

$$f_1 = abc + \bar{a}\bar{b} + bcd$$

треба заповнити карту Карно (рис. 3.2, б).

Функція f_1 є функцією чотирьох змінних. Елементарній кон’юнкції abc відповідають дві клітинки – 11- і 12-та, кон’юнкції $\bar{a}\bar{b}$ відповідає група клітинок – перша, друга, третя, четверта, кон’юнкції bcd – клітинки сьома і 11-та. Заповнивши ці клітинки одиницями, отримаємо карту Карно, що зображує функцію f_1.

Якщо функцію задано у КНФ, то для заповнення карти Карно можна розгорнути функцію у ДКНФ і в клітинках карти, що відповідають конституентам нуля, записати 0, а в решті клітинок – 1. Оскільки складаючи конституенти нуля, змінні, що дорівнюють нулеві у цьому наборі, записують без знака інверсії, а змінні, що дорівнюють одиниці, – зі знаком інверсії, то, наприклад, конституенті нуля $a+\bar{b}+c$ відповідає клітинка, для якої a = 0,

$b = 1$, $c = 0$, конституенті нуля $\bar{a} + b + \bar{c}$ – клітинка, для якої $a = 1$, $b = 0$, $c = 1$ тощо.

Для заповнення карти Карно можна не розгортати функцію у ДКНФ, а заповнювати нулями зразу групи клітинок, що відповідають елементарним диз'юнкціям. Як приклад розглянемо заповнення карти Карно для функції

$$f_2 = (a+b)(b+\bar{c}+d)(\bar{c}+\bar{d}),$$

заданій у КНФ (рис. 3.2, в).

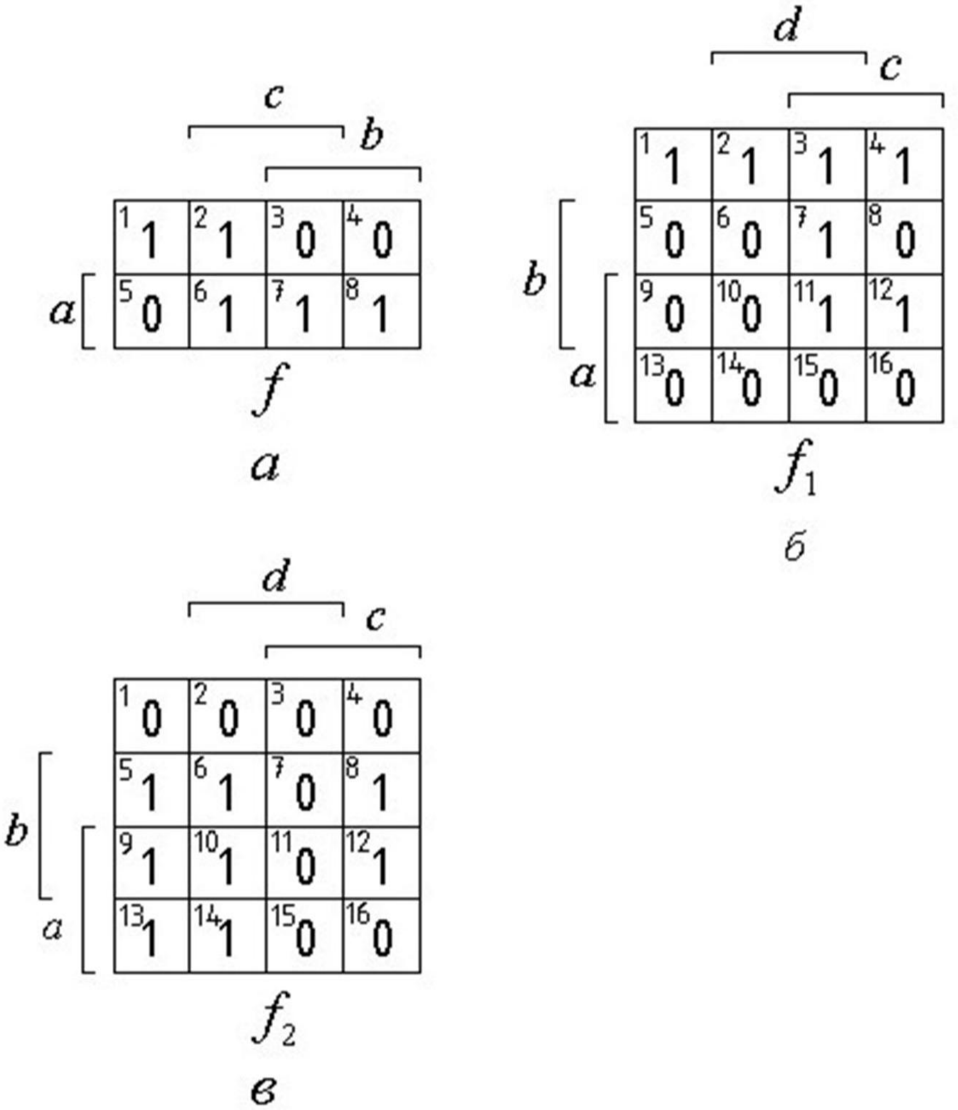


Рисунок 3.2 – Карти Карно: а – для функції, що задана таблицею істинності 3.1; б – для функції $f_1 = abc + \bar{a}\bar{b} + bcd$; в – для функції $f_2 = (a+b)(b+\bar{c}+d)(\bar{c}+\bar{d})$

Елементарній диз'юнкції $a+b$ відповідає група клітинок, для яких a = 0, b = 0. Це перша, друга, третя та четверта клітинки карти Карно. Диз'юнкції $b+\overline{c}+d$ відповідають клітинки, для яких b = 0, c = 1, d = 0, тобто четверта і 16-та. Диз'юнкції $\overline{c}+\overline{d}$ відповідає група клітинок, для яких c = 1, d = 1. Це третя, сьома, 11- і 15-та клітинки. Заповнивши ці клітинки нулями, а решту – одиницями, отримаємо карту Карно, що зображує функцію f_2.

Ознайомившись з принципом заповнення карт Карно для логічних функцій, перейдемо до розгляду їх властивостей. Виявляється, що за допомогою карт Карно досить просто відшукувати кон'юнкції або диз'юнкції, до яких можна застосувати операцію склеювання і таким чином мінімізувати логічну функцію.

Розглянемо карту Карно чотирьох змінних (див. рис. 3.1) і запишемо конституенти одиниці для всіх клітинок одного рядка карти, наприклад другого знизу, і одного стовпця, наприклад другого зліва. Тоді отримаємо:

для клітинок у рядку:	для клітинок у стовпці:
$ab\overline{c}\overline{d}$	$\overline{a}\overline{b}\overline{c}d$
$ab\overline{c}d$	$\overline{a}b\overline{c}d$
$abcd$	$ab\overline{c}d$
$abc\overline{d}$	$a\overline{b}\overline{c}d$

Аналізуючи ці конституенти, легко дійти висновку, що конституанти для клітинок, що розміщуються поряд, відрізняються значенням тільки однієї змінної, тобто для них можна застосувати операцію склеювання.

Будь-які клітинки, яким відповідають склеювані конституенти одиниці, називаються *сусідніми*. Сусідніми є не тільки клітинки, що розміщені поряд в одному рядку або стовпці, але й клітинки на протилежних кінцях одного рядка або стовпця. У цьому легко переконатися, порівнявши перші й останні конституенти для клітинок у рядку або стовпці.

Для карти чотирьох змінних знаходити сусідні клітинки досить просто. Дещо складніше їх знаходити для карт п’ятьох, шістьох й більшої кількості змінних. Карту п’ятьох змінних можна подати у вигляді двох карт чотирьох змінних, карту шістьох змінних – у вигляді чотирьох карт чотирьох змінних тощо. Карти чотирьох змінних, що відрізняються значенням тільки однієї змінної, також можна назвати сусідніми. Тоді сусідні карти будуть розміщені поряд або на протилежних кінцях одного рядка або стовпця з карт чотирьох змінних і, отже, сусідніми будуть клітинки, що є сусідніми у тій ж самій карті чотирьох змінних, а також клітинки у сусідніх картах, які розміщуються симетрично відносно ліній, що ділять карту великої кількості змінних на карти чотирьох змінних.

Як приклад на рис. 3.3 показано карту п’ятьох змінних. Лінії, які з’єднують точки, показують те, що пари стовпців є сусідніми. Наприклад, лінії у рядку $a = 0$, $b = 0$ (верхній рядок карти) показують, що кожний стовпець є сусіднім зі стовпцями, які розміщуються поряд з ним праворуч та ліворуч. У рядку $a = 0$, $b = 1$ показано, що сусідніми є крайні стовпці карт чотирьох та п’ятьох змінних. У рядку $a = 1$, $b = 1$ позначено як сусідні третій і шостий, другий та восьмий стовпці, тобто стовпці, які розміщуються симетрично відносно лінії, що ділить карту п’ятьох змінних на дві карти чотирьох змінних.

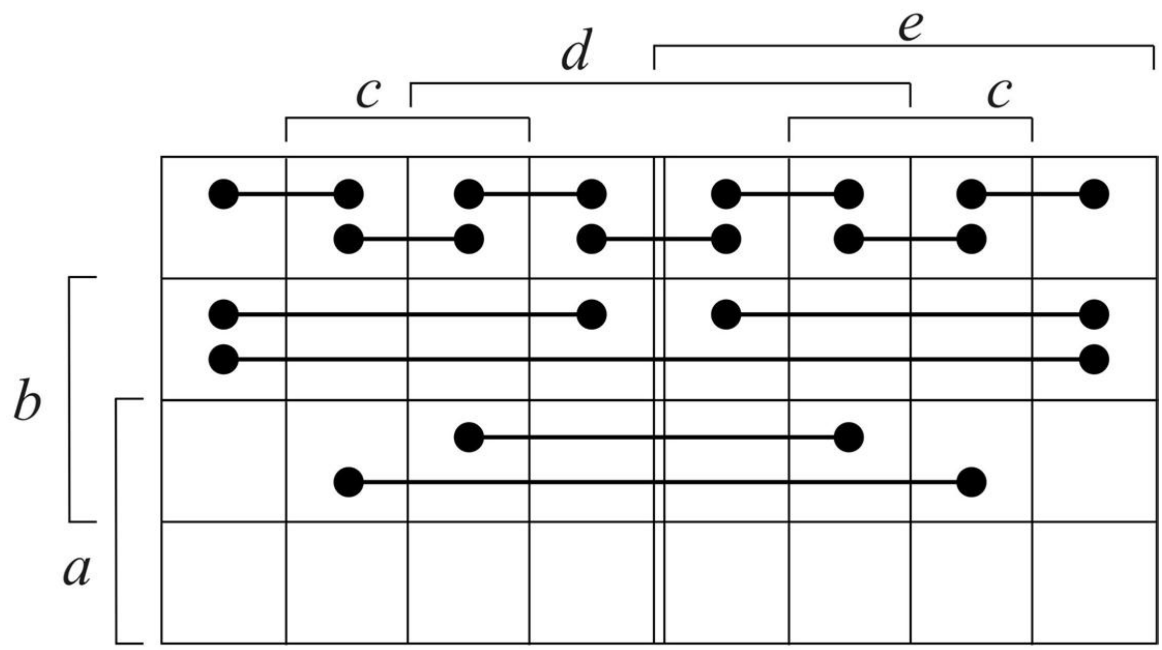


Рисунок 3.2 – До визначення сусідніх клітинок для карти Карно п'ятьох змінних

Властивості карт Карно, які дозволяють досить просто відшукувати сусідні клітинки, використовуються для мінімізації логічних функцій.

Розглянемо, наприклад, логічну функцію, подану картою Карно (рис. 3.4, а). Функція f_1 у ДДНФ має вигляд

$$f_1 = \overline{a}b\overline{c}d + \overline{a}bcd \ .$$

Застосувавши теорему склеювання, отримаємо $f_1 = \overline{a}bd$.

Цей результат можна отримати, не записуючи функцію у ДДНФ, таким способом. Одиниці стоять у сусідніх клітинках. Ці клітинки можна об'єднати в контур. Якщо об'єднуються дві сусідні клітинки, то конституенти одиниці для них розрізняються тільки однією змінною: у першу конституанту одиниці входить змінна *c* з інверсією, а в другу – без неї. У загальному виразі для контуру з двох клітинок змінної с не повинно бути, оскільки заміна двох сусідніх клітинок одним контуром аналогічна застосуванню теореми склеювання.

Так само для функцій, поданих картами Карно на рис. 3.3 б,в,г, можна записати

$$f_2 = bcd\ ;\ f_3 = \overline{b}\overline{c}\overline{d}\ ;\ f_4 = ab\overline{d}\ .$$

При цьому слід звернути увагу на те, що клітинки з одиницями у картах Карно на рис.3.3, в, г є сусідніми, хоч і розділені просторово, і, отже, їх також можна об'єднати в один контур.

Розглянемо тепер карти Карно на рис.3.3, д–к. Довершена диз'юнктивна нормальна форма функції f_5 має вигляд

$$f_5 = \overline{a}b\overline{c}\overline{d} + \overline{a}b\overline{c}d + \overline{a}bcd + \overline{a}bc\overline{d}\ .$$

Застосувавши теорему склеювання, отримаємо

$$f_5 = \overline{a}b\overline{c} + \overline{a}bc = \overline{a}b\ .$$

Такий самий вираз виходить, якщо об'єднати чотири клітинки в контур. Для решти карт $f_6 = bd$; $f_7 = \overline{b}\overline{c}$; $f_8 = \overline{b}\overline{d}$.

Слід звернути увагу на те, що у вираз для контуру не входять ті змінні, межі яких перетинаються контуром. Наприклад, контур на карті, перетинає межі змінних *a* і *c*, і саме ці змінні не входять в остаточний вираз функції f_6 .

У контур можна об'єднати й вісім клітинок. Приклади таких об'єднань показано на рис. 3.3, л, м, н, п. Функції, подані цими картами, мають вигляд

$$f_9 = a\ ;\ f_{10} = \overline{b}\ ;\ f_{11} = c\ ;\ f_{12} = \overline{d}\ .$$

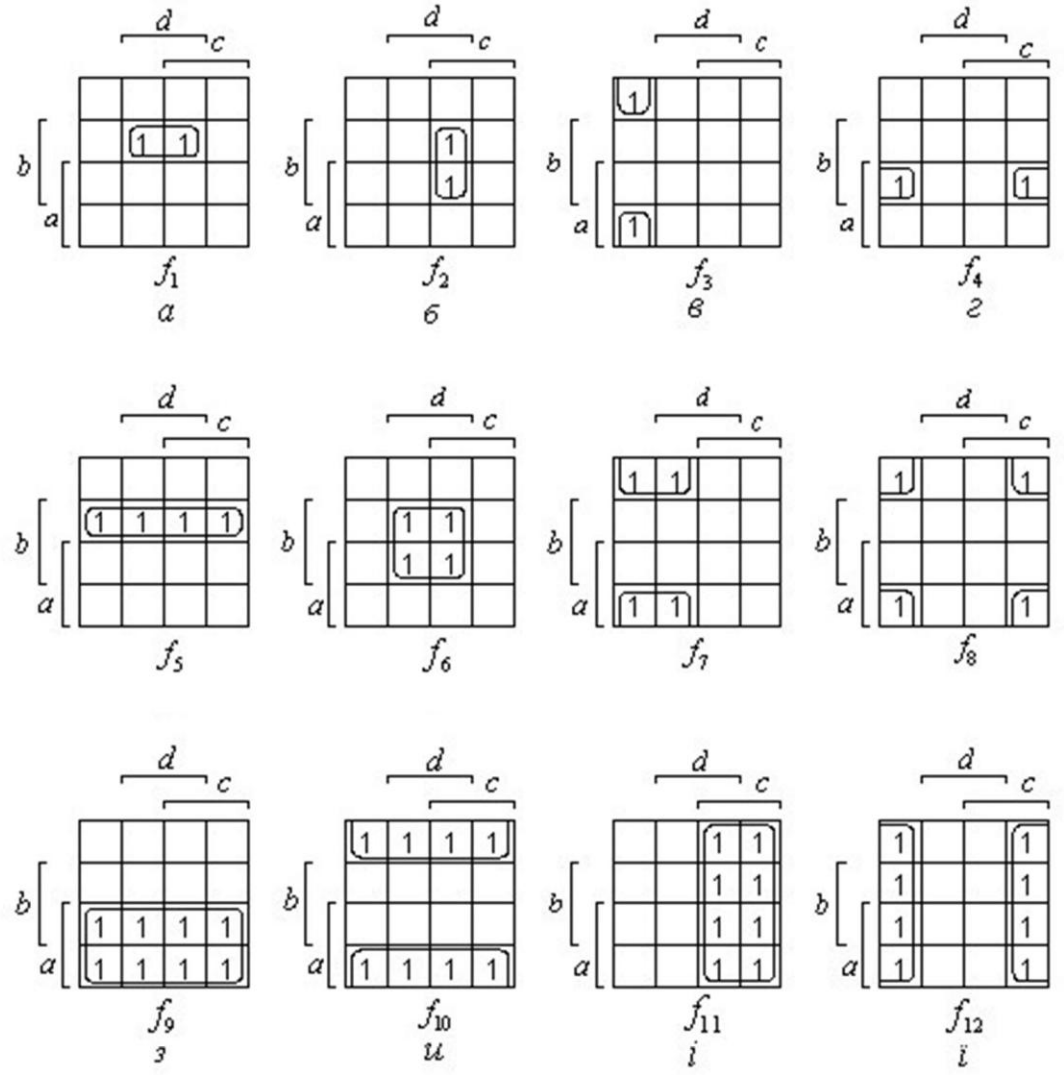


Рисунок 3.3 – Варіанти об’єднання клітинок для карт Карно чотирьох змінних

Розглянувши карти Карно (див. рис. 3.3), можна дійти таких висновків. Якщо в контур об’єднано дві клітинки, то вираз для контуру містить на одну змінну менше порівняно з конституентою одиниці, якщо чотири клітинки – на дві змінні менше, вісім – на три. Взагалі, якщо контур містить 2^n клітинок, то у вираз для контуру входить на n змінних менше. Щоб використати карту Карно для мінімізації логічних функцій, необхідно побудувати карту для відповідної кількості змінних і нанести на неї задану функцію. Потім слід об’єднати сусідні клітинки з одиницями у контури, записати вирази для контурів і скласти їх диз’юнкцію.

Сусідні клітинки спочатку об'єднують у пари, потім у четвірки із сусідніх пар, тобто пар, що відрізняються тільки однією змінною, після цього сусідні четвірки об'єднують у вісімки тощо. Чим більше клітинок об'єднано у контур, тим простіший вираз, що відповідає контуру, тому слід прагнути того, щоб кожний контур мав якомога більше сусідніх клітинок. При цьому деякі контури можуть частково перекриватися, тобто ті ж самі клітинки можуть одночасно входити у кілька контурів. У контур можна об'єднувати не будь-яку парну кількість клітинок, а тільки 2^n клітинок, тобто 2, 4, 8, 16 тощо.

Крім того, необхідно уникати створення зайвих контурів, тобто контурів, усі клітинки яких уже належать до інших контурів. Для цього об'єднання слід починати з тих одиниць, які можуть увійти тільки в один контур. Це положення ілюструється картою Карно (рис.3.4, а) Контур з чотирьох одиниць тут зайвий через те, що всі клітинки цього контуру вже увійшли до інших контурів.

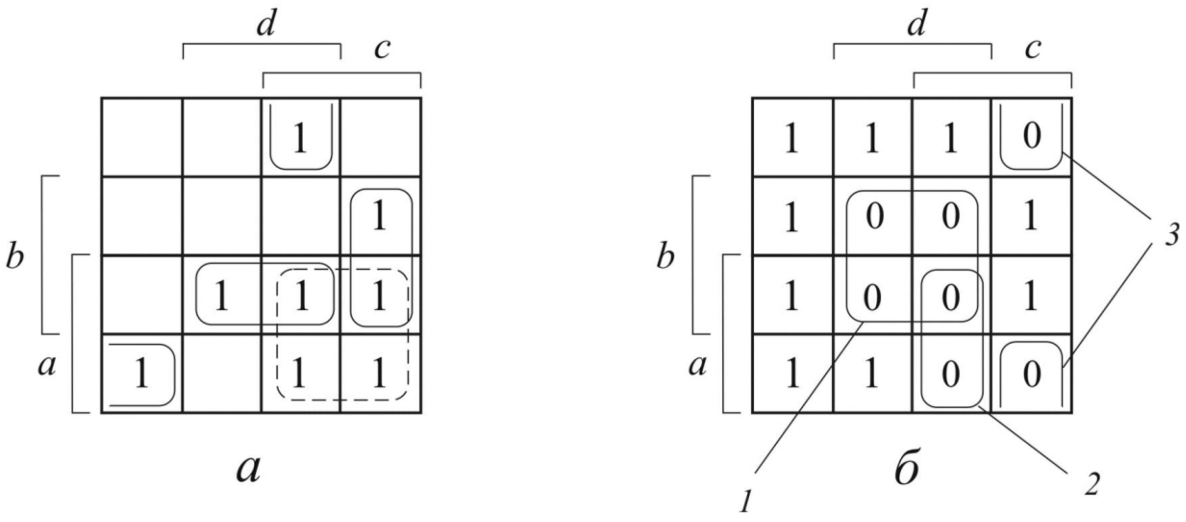


Рисунок 3.4 – Приклади об'єднання клітинок карти Карно в контури: а – з одиницями; б – з нулями

У контури можна об’єднувати клітинки не тільки з одиницями, але й з нулями. При цьому всі правила об’єднування залишаються попередніми, а функція записується у вигляді кон’юнкції елементарних диз’юнкцій, що відповідають контурам з нулями.

Вираз для контуру з нулями записують у вигляді диз’юнкції інверсій координат контуру. Наприклад, об’єднавши клітинки з нулями у карті Карно, отримаємо для першого контуру $\bar{b}+\bar{d}$, для другого $\bar{a}+\bar{c}+\bar{d}$, для третього $b+\bar{c}+d$. Функція у КНФ матиме вигляд

$$f=\left(\bar{b}+\bar{d}\right)\left(\bar{a}+\bar{c}+\bar{d}\right)\left(b+\bar{c}+d\right).$$

Приклади мінімізації логічних функцій за допомогою карт Карно ілюструє рис. 3.5, на якому показано способи об’єднання клітинок з одиницями у контури. У результаті виконаних об’єднань отримано такі мінімізовані вирази функцій:

- $f_1=\bar{a}d+cd+\bar{b}c+\bar{a}b\bar{c}$;
- $f_2=bc+\bar{b}\bar{d}+a\bar{d}\bar{e}$;
- $f_3=cd+\bar{b}\bar{d}+\bar{a}\bar{d}$;
- $f_4=bd+be+\bar{c}e$;
- $f_5=\bar{c}\bar{d}\bar{e}+\bar{a}\bar{b}\bar{e}+b\bar{c}\bar{e}+ac\bar{d}$.

У розглянутих прикладах функції були повністю визначеними, тобто кожному наборові вхідних змінних відповідало цілком визначене значення функції: 0 або 1. Проте під час роботи багатьох дискретних систем керування можуть бути не всі набори вхідних змінних. Наприклад, одночасно не може бути сигналів про верхній та нижній рівні рідини, про наявність механізму у крайньому лівому і крайньому правому положеннях тощо. Набори вхідних змінних, яких за заданих умов роботи ніколи не може бути, називаються *невикористаними станами входів*. У клітинках карти Карно, що відповідають таким станам, проставляють риски. Функції, що описують схеми з невикористаними станами входів, називають *неповністю визначеними*.

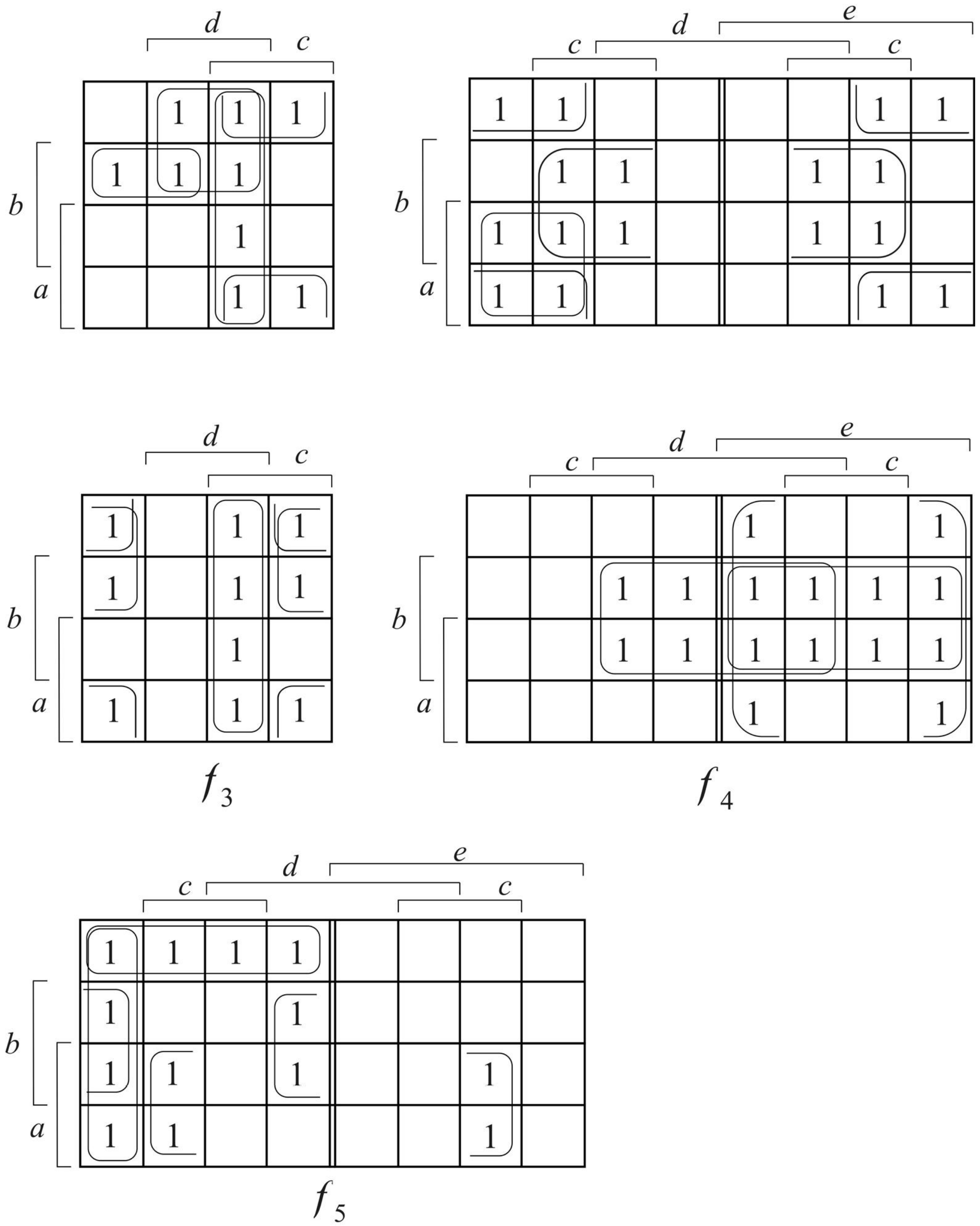


Рисунок 3.5 – Приклади мінімізації логічних функцій за допомогою карт Карно

Наявність невикористаних станів дозволяє спрощувати логічну функцію. Нехай, наприклад, функцію задано картою Карно (рис. 3.6, а), причому відомо, що сигнали a і c одночасно не можуть дорівнювати одиниці. У клітинках карти, що відповідають значенням a = 1, c = 1, проставлено риски. Якщо скласти вираз функції f_1, об’єднавши в контури тільки клітинки з одиницями, то отримаємо

$$f_1 = a\bar{d}\bar{c} + ab\bar{c} + \bar{a}bc. \quad (3.1)$$

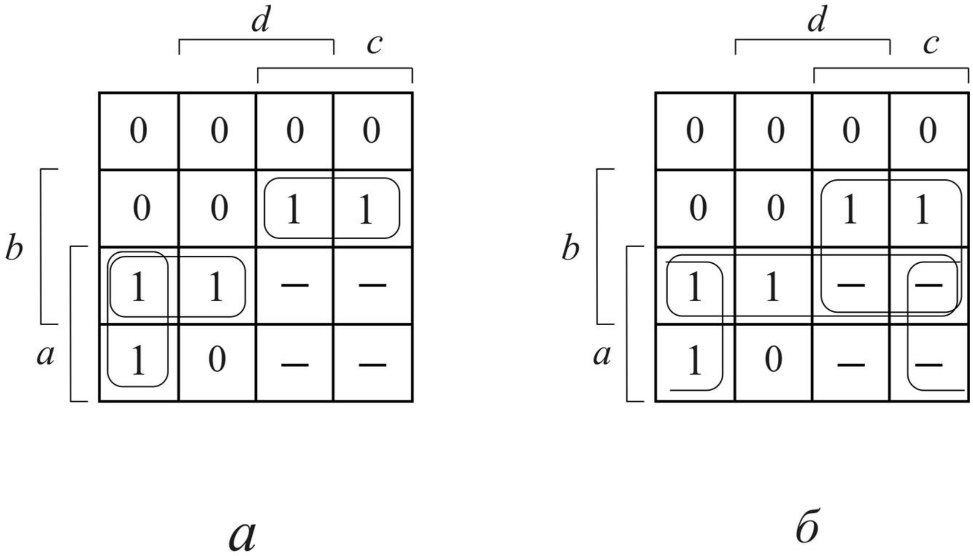


Рисунок 3.6 – Приклади об’єднання клітинок з одиницями в картах Карно: а – без урахування невикористаних станів; б – з їх урахуванням

Для спрощення функції у контури можна об’єднувати клітинки не тільки з одиницями, але й із рисками (див. рис. 3.6, б). Тоді отримаємо

$$f_2 = ab + a\bar{d} + bc. \quad (3.2)$$

Функції f_1 і f_2, точно кажучи, не рівносильні. Наприклад, якщо a = 1,
b = 0, c = 1, d = 0, функція f_1 = 0, а f_2 = 1. Проте, якщо вважати, що a і c одночасно не дорівнюватимуть одиниці, то схеми, побудовані за формулами (3.1) і (3.2), працюватимуть однаково.

Відомі також інші методи мінімізації логічних функцій. Проте карти Карно, завдяки наочності та простоті, мають незаперечні переваги порівняно з іншими методами. Їх найчастіше застосовують для мінімізації логічних функцій з порівняно невеликою кількістю змінних (до шести) [3, 4, 7].

3.2. Приклади застосування карт Карно для мінімізації логічних функцій

Приклад 1.

Логічну функцію

$$y_{ДДНФ} = x_1\overline{x_2}x_3 \vee \overline{x_1}\,\overline{x_2}x_3 \vee \overline{x_1}\,\overline{x_2}\,\overline{x_3} \vee \overline{x_1}x_2\overline{x_3} \vee x_1x_2\overline{x_3} \vee x_1x_2x_3,$$

задану в ДДНФ, мінімізувати за допомогою карти Карно.

Розв'язання.

1. Зобразимо карту Карно для трьох змінних X_1, X_2, X_3 і відмітимо в ній одиничні мінтерми $x_1\overline{x_2}x_3$, $\overline{x_1}\,\overline{x_2}x_3$, $\overline{x_1}\,\overline{x_2}\,\overline{x_3}$, $x_1x_2\overline{x_3}$, $x_1x_2x_3$ і $\overline{x_1}x_2\overline{x_3}$ (рис 3.7, а).

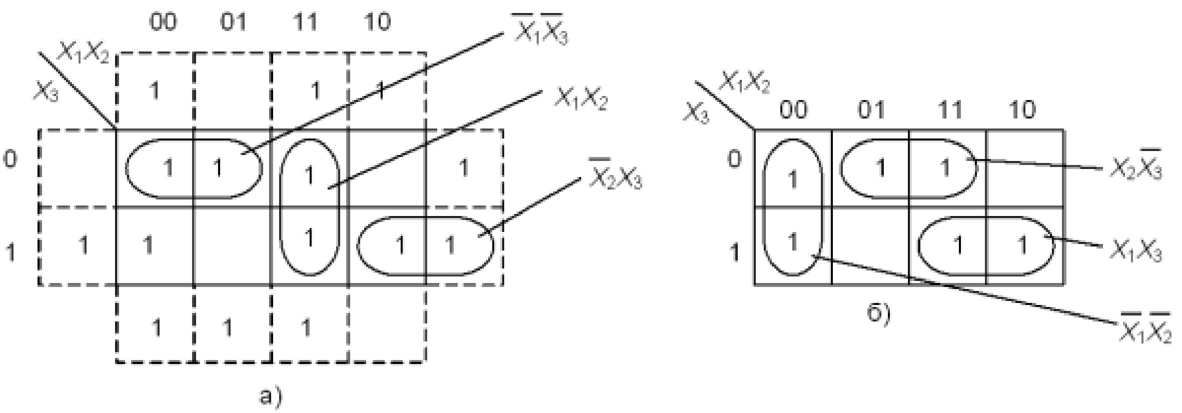


Рисунок 3.7 – Мінімізація логічної функції за допомогою карт Карно

2. В карті Карно (див. рис. 3.7, а) мінтерми утворюють три групи, кожна з яких містить два мінтерми. Перша складається з $\overline{x_1}x_2\overline{x_3}$ і $\overline{x_1}\,\overline{x_2}\,\overline{x_3}$, з цієї групи змінна x_2 може бути виключена. Друга група складається з $x_1x_2x_3$ та $x_1x_2\overline{x_3}$ і з цієї групи може бути виключена змінна x_3. Третя група складається з $x_1\overline{x_2}x_3$ і $\overline{x_1}\,\overline{x_2}x_3$, з якої може бути виключена змінна x_1.

3. Записуємо мінімізовану логічну функцію в ДНФ:

$$y1 = x_1x_2 \vee \overline{x_2}x_3 \vee \overline{x_1}\,\overline{x_3}.$$

Вибираючи групи мінтермів по-іншому (див. рис. 3.7, б), отримуємо другу мінімальну форму логічної функції, заданої рівнянням

$$y_{ДДНФ} = x_1\overline{x_2}x_3 \vee \overline{x_1}\,\overline{x_2}x_3 \vee \overline{x_1}\,\overline{x_2}\,\overline{x_3} \vee \overline{x_1}x_2\overline{x_3} \vee x_1x_2\overline{x_3} \vee x_1x_2x_3,$$

$$y2 = \overline{x_1}\,\overline{x_2} \vee x_2\overline{x_3} \vee x_1x_3.$$

Приклад 2.

Мінімізувати логічну функцію, що представлена у вигляді карти Карно (рис. 3.8).

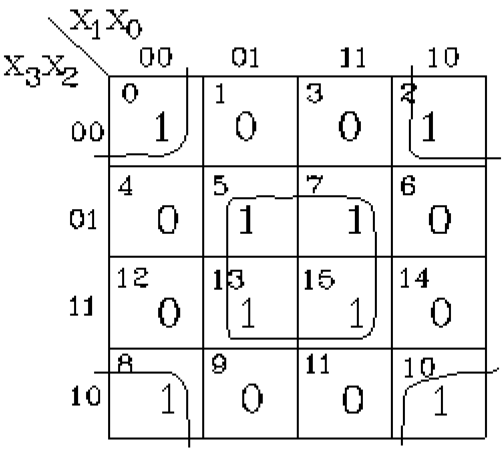


Рисунок 3.8 – Логічна функція, що представлена у вигляді карти Карно

Розв’язання.

Об’єднуючи сусідні клітини з істинними значеннями функції (тобто клітини 0, 2, 8, 10 та клітини 5, 7, 13, 15), записуємо їх незмінні координати:

для центральних клітин: $y_1 = x_0x_2$;

для крайніх клітин: $y_2 = \overline{x_0}\,\overline{x_2}$.

Мінімізована логічна функція має вигляд:

$$y = y_1 + y_2 = x_0x_2 + \overline{x_0}\,\overline{x_2}.$$

Аналогічний результат буде одержаний і при об’єднанні клітин з нулями. Для вертикальної групи клітин (клітини 1, 3, 9, 11) знаходимо:

$$y_3 = x_0\overline{x_2}.$$

Для групи горизонтальних клітин (клітини 4, 6, 12, 14):

$$y_4 = \overline{x_0} x_2 .$$

Мінімізована функція:

$$\overline{y} = x_0 \overline{x_2} + \overline{x_0} x_2$$

операцією інверсії та теоремою де Моргана зводиться до отриманої раніше [3, 4].

3.3. Алгоритм роботи калькулятора

Розглянемо основні етапи роботи калькулятора:

- введення логічного виразу;
- перевірка введеного виразу;
- обробка і виведення результату.

Введення логічного виразу.

Користувачу відображається поле для введення виразу, поле з кнопками, та можливість виведення підказки.

Якщо натиснути кнопку з символом «¬», то у поле вводиться логічний оператор «Заперечення». Щоб заперечити певний вираз, слід ввести оператор і в «дужках» вказати даний вираз.

При натисненні кнопки з символом «(» у поле вводиться відповідний символ.

При натисненні кнопки з символом «)» у поле вводиться відповідний символ.

При натисненні кнопки з символом «□» у поле вводиться логічний оператор «Диз’юнкція».

При натисненні кнопки з символом «□ » у поле вводиться логічний оператор «Кон’юнкція».

При натисненні кнопки з символом «→» у поле вводиться логічний оператор «Імплікація».

При натисненні кнопки з символом «←» у поле вводиться логічний оператор «Зворотна імплікація».

При натисненні кнопки з символом «~» у поле вводиться логічний оператор «Еквівалентність».

При натисненні кнопки з символом «⊕» у поле вводиться логічний оператор «Виключна диз'юнкція».

При натисненні кнопки з символом «|» у поле вводиться логічний оператор «Штрих Шеффера».

При натисненні кнопки з символом «↓» у поле вводиться логічний оператор «Стрілка Пірса».

Якщо натиснути кнопку з символом «Back», то видаляється останній введений символ.

Якщо натиснути кнопку з символом «Del», то поле вводу очищається.

При натисненні кнопки з символом «a» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «b» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «c» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «d» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «e» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «f» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «g» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «h» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «i» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «j» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «k» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «l» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «m» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «n» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «o» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «p» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «q» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «r» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «s» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «t» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «u» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «v» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «w» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «x» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «y» у поле вводиться відповідна змінна.

При натисненні кнопки з символом «z» у поле вводиться відповідна змінна.

Змінні та символи «(», «)» також можна ввести з клавіатури. Якщо змінні вводяться підряд, то вважається, що між ними ставиться оператор «Кон’юнкція».

Після введення слід натиснути кнопку «Мінімізувати логічну функцію».

Перевірка введеного виразу.

Якщо логічний вираз містить більше шести змінних, то виводиться повідомлення «За допомогою карт Карно легко вирішуються задачі мінімізації функцій з кількістю змінних до шести включно. При більшій кількості змінних пошук мінімальних форм запису функцій значно ускладняється і наочність карт Карно втрачається.».

Якщо вираз починається з логічного оператора, крім «Заперечення», то виводиться помилка: «Помилка! Вираз не може починатися з логічного оператора, крім «Заперечення».».

Якщо вираз містить декілька логічних операторів підряд, то виводиться помилка: «Помилка! Вираз не може містити декілька логічних операторів підряд». Також наводиться кількість введених операторів.

Якщо вираз закінчується логічним оператором, то виводиться помилка: «Помилка! Вираз не може закінчуватися логічним оператором.».

Якщо вираз містить «дужки» і їх кількість непарна, то виводиться помилка: «Помилка! Вираз містить невірну кількість «дужок».». Якщо ж вони неправильно розставлені, то «Помилка! Перевірте правильність розстановки «дужок».»

Якщо вираз містить лише одну змінну, то виводиться помилка: «Помилка! Вираз має містити більше, ніж одну змінну.».

Обробка і виведення результату.

Спочатку будується таблиця істинності логічної функції. Якщо функція завжди дорівнює 1, то виводиться повідомлення «При всіх значеннях змінних функція завжди дорівнює 1.». Якщо функція дорівнює 0 – «При всіх значеннях змінних функція завжди дорівнює 0.»

В іншому випадку виводяться таблиці істинності логічних операцій, що були використані.

Наступним кроком будується карта Карно для функції відповідної кількості змінних (див. рис. 3.1). Після чого виводиться текст «Об'єднуючи сусідні клітини з істинними значеннями функції, отримуємо мінімізовану логічну функцію:» та відображається отриманий результат.

Для того, щоб побачити як отримано мінімізовану функцію, потрібно навести курсор (натиснути) на відповідну кон'юнкцію і в таблиці виведеться відповідна виділена область.

3.4. Обґрунтування вибору програмних засобів

У мови Java ™ є багато переваг перед іншими мовами програмування, що дозволяє вирішувати з його допомогою практично будь-які завдання.

Нижче перераховані основні переваги Java:

- Мова Java проста для вивчення.

При розробці Java було приділено велику увагу простоті мови, тому програми на Java, в порівнянні з програмами на інших мовах, простіше писати, компілювати, налагоджувати і вивчати.

- Java - це об'єктно-орієнтована мова.

Це дозволяє створювати модульні програми, вихідний код яких може використовуватися багаторазово.

- Мова Java не залежить від платформи.

Одним з основних переваг мови Java є можливість перенесення програм з однієї системи в іншу. Оскільки програми на Java не залежать від платформи як на рівні вихідного коду, так і на довічним рівні, їх можна запускати в різних системах, що особливо важливо для програм, призначених для World Wide Web.

Широкі можливості Java, простота застосування, незалежність від платформи і вбудовані функції захисту роблять цю мову програмування одним з кращих для створення додатків для Internet [8].

NetBeans - це інтегроване середовище розробки, яке підтримує масу мов, включаючи найбільш популярні, такі як Python, Java, C / C ++, обробку XML, взаємодія з базами даних та інші функції, характерні для сучасної IDE.

Як і в будь-який інший подібної системи, в NetBeans реалізована підтримка рефакторинга коду, його профілювання, а також колірне виділення і генерація ділянок коду на льоту. Ще одна схожість для всіх основних середовищ розробки - це необхідність попередньої установки Java Developer Kit для запуску NetBeans.

IDE підтримує основні платформи для малих, середніх і великих підприємств: Java Enterprise і Standard Edition. З огляду на розвиток мобільних пристроїв, нові версії працюють і з платформою Java Micro Edition, яка призначена для створення додатків на девайсах, ресурси яких істотно обмежені.

Дивлячись на істотне навантаження і концентрацію на більш комерційних проектах, компанія Oracle прийняла рішення про передачу NetBeans в руки іншої компанії. Так, починаючи з 2016 року, середа розробки ПО підтримується фондом Apache Software Foundation.

Загальним недоліком для всіх середовищ, які запускаються через JDK / SDK, буде істотне споживання ресурсів пристрою. Як Eclipse, так і NetBeans не застраховані від цього. Але від поновлення до оновлення все стає краще. Та й Microsoft потроху пом'якшує агресивну політику по відношенню до Java [9].

4. ПРАКТИЧНА ЧАСТИНА

4.1. Опис процесу програмної реалізації

Першим кроком було створено графічний вид калькулятора (рис. 4.1):

- додано поле для вводу виразу;
- кнопки з операціями;
- кнопки із змінними;
- селектор для зміни мови;
- кнопка для перевірки введеного виразу та виведення результату.

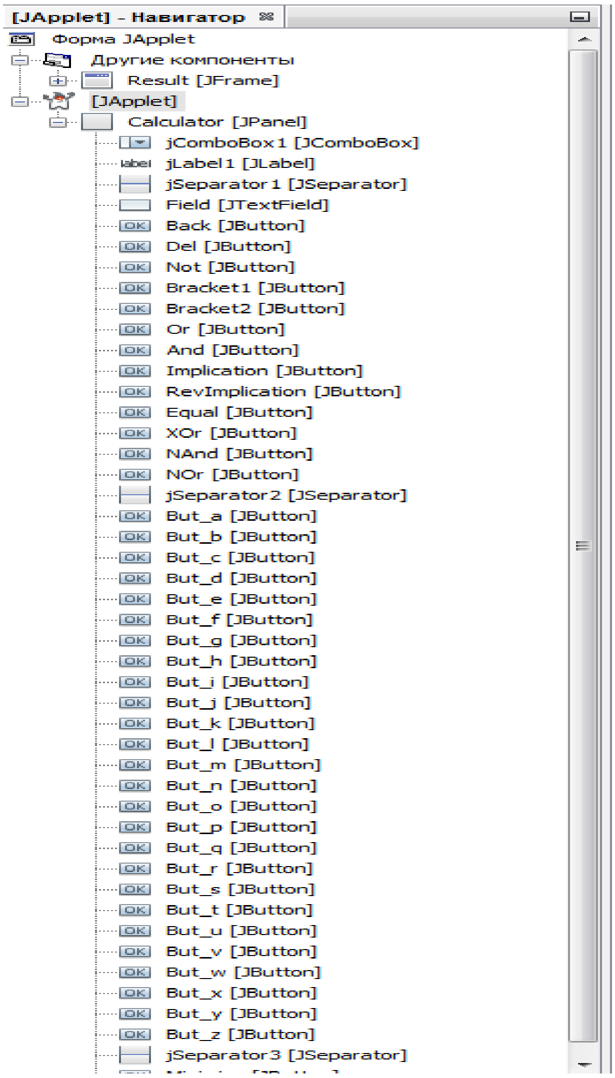


Рисунок 4.1 – Створені елементи калькулятора

Під час програмної реалізації було підключено бібліотеки, що використовуються в деяких функціях та подіях.

```
import java.awt.Font;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;
import javax.script.ScriptException;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
```

Також створено глобальні змінні.

```
String str;
char[] ch;
List<String> list;
```

str – зберігається введений вираз, ch – використовується при символьному розбитті str, list – зберігаються введені змінні.

За ініціалізацію і відображення калькулятора відповідає наступний код:

```
public class JApplet extends javax.swing.JApplet {
    /**
     * Initializes the applet JApplet
     */
    @Override
    public void init() {
        /* Create and display the applet */
```

```
        try {
            java.awt.EventQueue.invokeAndWait(new Runnable() {
                public void run() {
                    initComponents();
                }
            });
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
```

При виборі мови відбувається подія jComboBox1ItemStateChanged, що змінює назву калькулятора і кнопки «Мінімізувати логічну функцію» на вибрану.

```
    private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
        if(jComboBox1.getSelectedIndex()==0) {
            jLabel1.setText("Карти Карно");
            Minimize.setText("Мінімізувати логічну функцію");
        } else {
            jLabel1.setText("Karnaugh map");
            Minimize.setText("Minimize logical function");
        }
    }
```

В поле з клавіатури можна вводити лише літери a-z або «дужки», це забезпечується подією FieldKeyTyped.

```
    private void FieldKeyTyped(java.awt.event.KeyEvent evt) {
        char c = evt.getKeyChar();
        String s = Character.toString(c);
```

```
        if (!s.matches("[a-z()]")) {
            evt.consume();
        }
    }
```

Знаки операції можна вводити лише натиснувши відповідну кнопку калькулятора. Приклад, знак «Заперечення»:

```
    private void NotActionPerformed(java.awt.event.ActionEvent evt) {
        Field.setText(AddChar('¬'));
    }
```

При цьому використовується функція AddChar(char c), що була створена для введення вказаного символу на місці курсору. Символ задається через параметр char c.

```
    public String AddChar(char c) {
        str = Field.getText();
        return str.substring(0, Field.getCaretPosition()) + c + str.substring(Field.getCaretPosition());
    }
```

Змінні також можна вводити таким чином. Приклад, символ «a» :

```
    private void But_aActionPerformed(java.awt.event.ActionEvent evt) {
        Field.setText(AddChar('a'));
    }
```

Кнопка «Del» очищує поле для вводу, а «Back» – видаляє останній введений символ.

```
    private void DelActionPerformed(java.awt.event.ActionEvent evt) {
        Field.setText("");
    }
```

```
private void BackActionPerformed(java.awt.event.ActionEvent evt) {
   str = Field.getText();
   if(str.length()>0) {
      Field.setText(str.substring(0, str.length()-1));
   }
}
```

При натисненні кнопки «Мінімізувати логічну функцію» відбувається подія MinimizeActionPerformed. Спочатку перевіряється введений вираз за допомогою функції Check(), а потім проводиться обрахунок і виведення результату за допомогою Processing().

```
private void MinimizeActionPerformed(java.awt.event.ActionEvent evt) {
   if(Check()) {
      Processing();
   }
}
```

Функція Check() перевіряє кожен символ виразу і якщо одна з наведених в алгоритмі умов помилки виконується, то виводиться відповідне повідомлення (див. Додаток А).

```
private boolean Check() {
…
   for(int i=0; i<str.length(); i++) {
      if(ch[i] == '(') {
         br1_count++;
         if(i == str.length()-1) {
            if(jComboBox1.getSelectedIndex()==0) {
               ShowError("Помилка! Перевірте правильність розстановки «дужок».");
```

```
                } else {
                    ShowError("Error! Check the placement of the brackets.");
                }
                err = true;
                break;
            }
        }
…
```

Оскільки у функції Processing() проводяться обрахунки і вона виводить таблицю істинності виразу та решту результатів, то в ній використовуються декілька інших функцій: String Out(boolean b), String CreateExpr(String s).

Так як введений символ зберігається у змінній String, то його потрібно перетворити у логічне вираження. Для цього було розроблено String CreateExpr(String s), де String s – вираз для перетворення (див. Додаток А).

```
public String CreateExpr(String s) {
    for(int i=0; i<s.length(); i++) {
        switch(s.charAt(i)) {
            case '¬':
                s = s.substring(0, i) + "!" + s.substring(i+1);
                break;
            case '□ ':
                s = s.substring(0, i) + " || " + s.substring(i+1);
                break;
            case '□ ':
                s = s.substring(0, i) + " && " + s.substring(i+1);
                break;
            …
```

Функція String Out(boolean b) використовується для заповнення таблиці істинності. В залежності від значення «true/false» виводить 1 або 0.

```
public String Out(boolean b) {
   return b ? "1" : "0";
}
```

4.2. Опис програми

Вікно калькулятора має наступний вигляд (рис. 4.2).

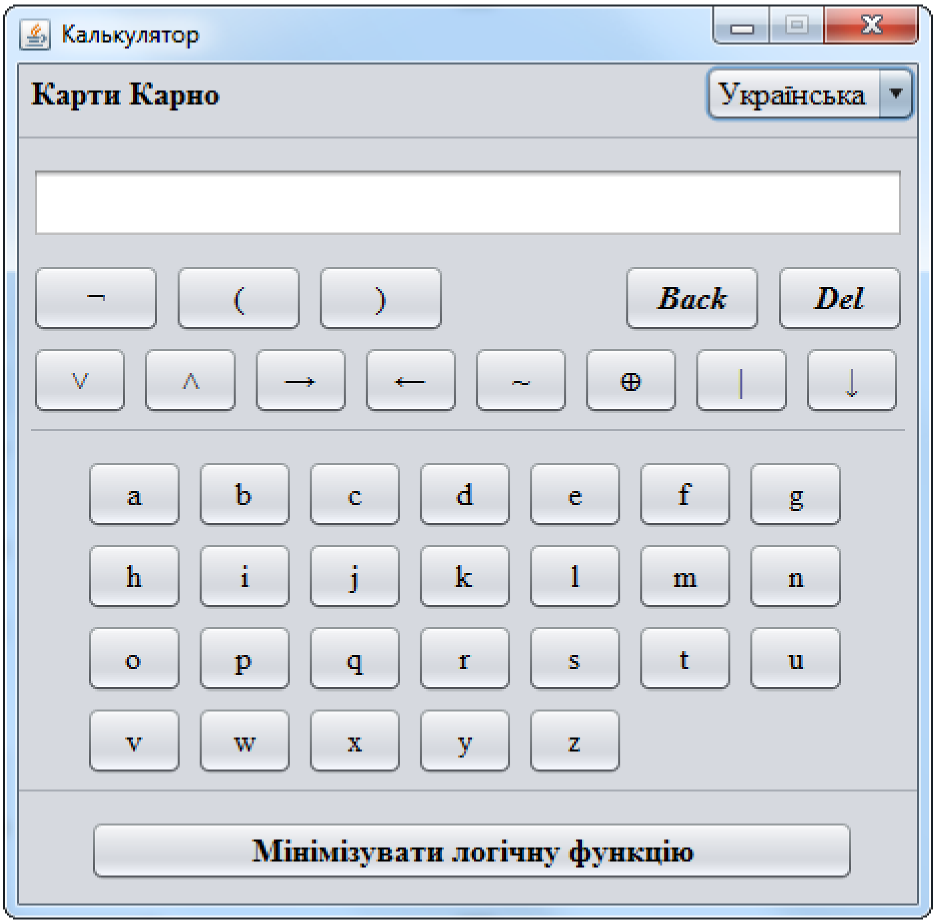


Рисунок 4.2 – Вікно калькулятора

Після зміни мови зміниться назва калькулятора і кнопка «Мінімізувати логічну функцію», всі повідомлення про помилку виводяться на вибраній мові (рис. 4.3-4.4).

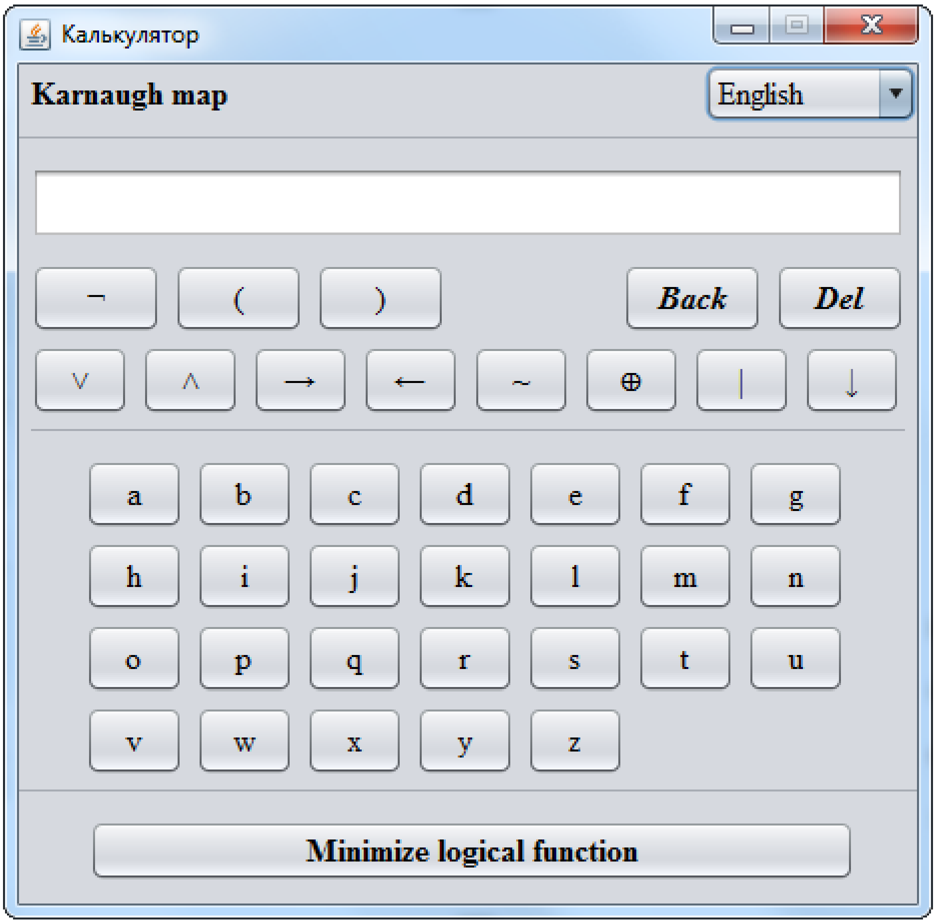


Рисунок 4.3 – Вікно калькулятора англійською мовою

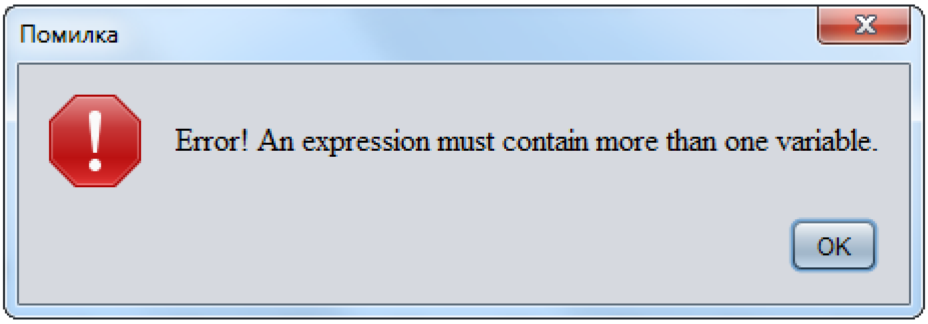


Рисунок 4.4 – Повідомлення про помилку англійською мовою

Знаки операції можна вводити лише натиснувши відповідну кнопку калькулятора. Змінні також можна вводити таким чином. В поле з клавіатури можна вводити лише літери a-z або «дужки» (рис. 4.5).

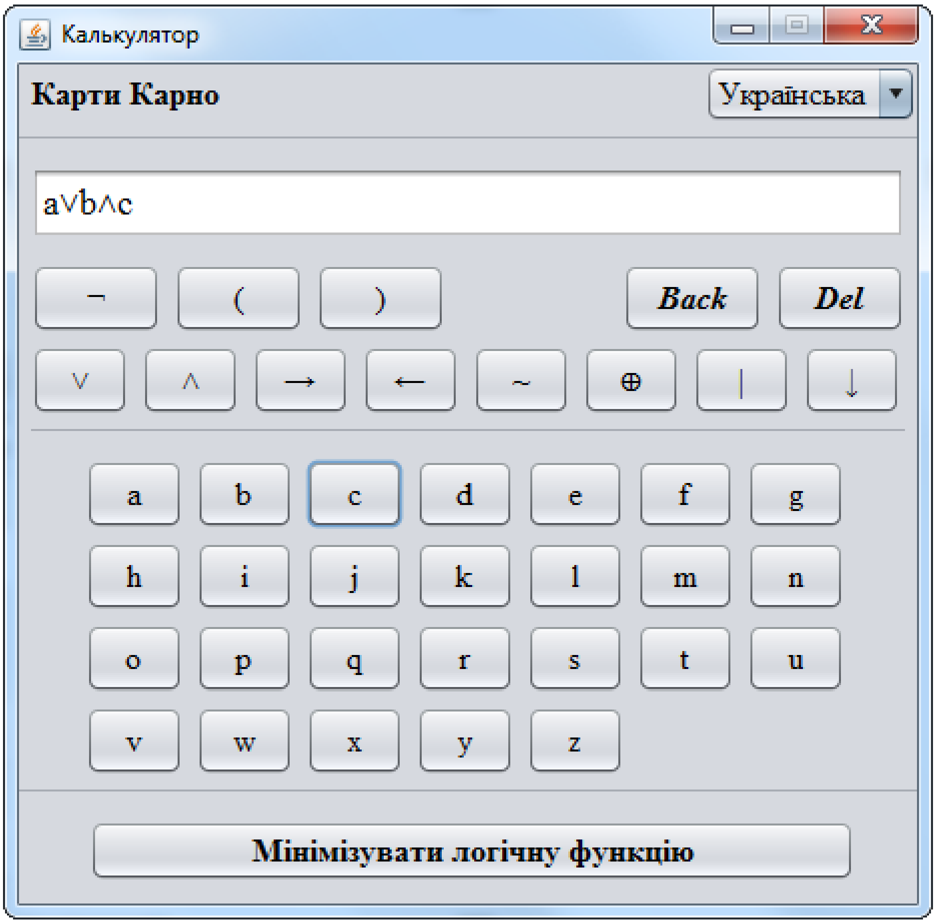


Рисунок 4.5 – Вікно калькулятора

Кнопка «Del» очищує поле для вводу. Кнопка «Back» видаляє останній введений символ.

Якщо у виразі містяться помилки, то виводиться повідомлення про помилку. Наприклад, якщо ввести кінці «)», то виведеться (рис. 4.6):

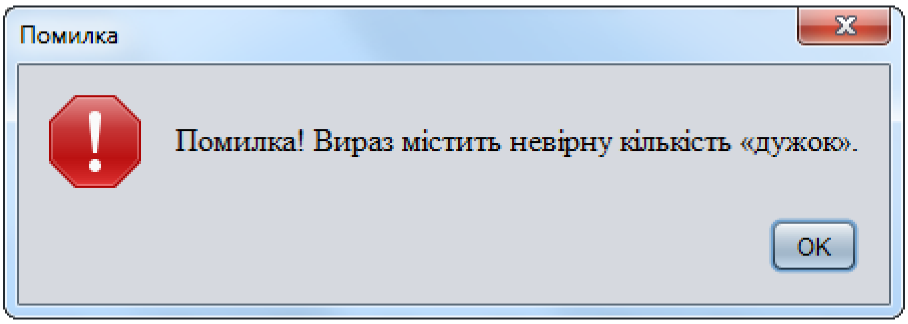
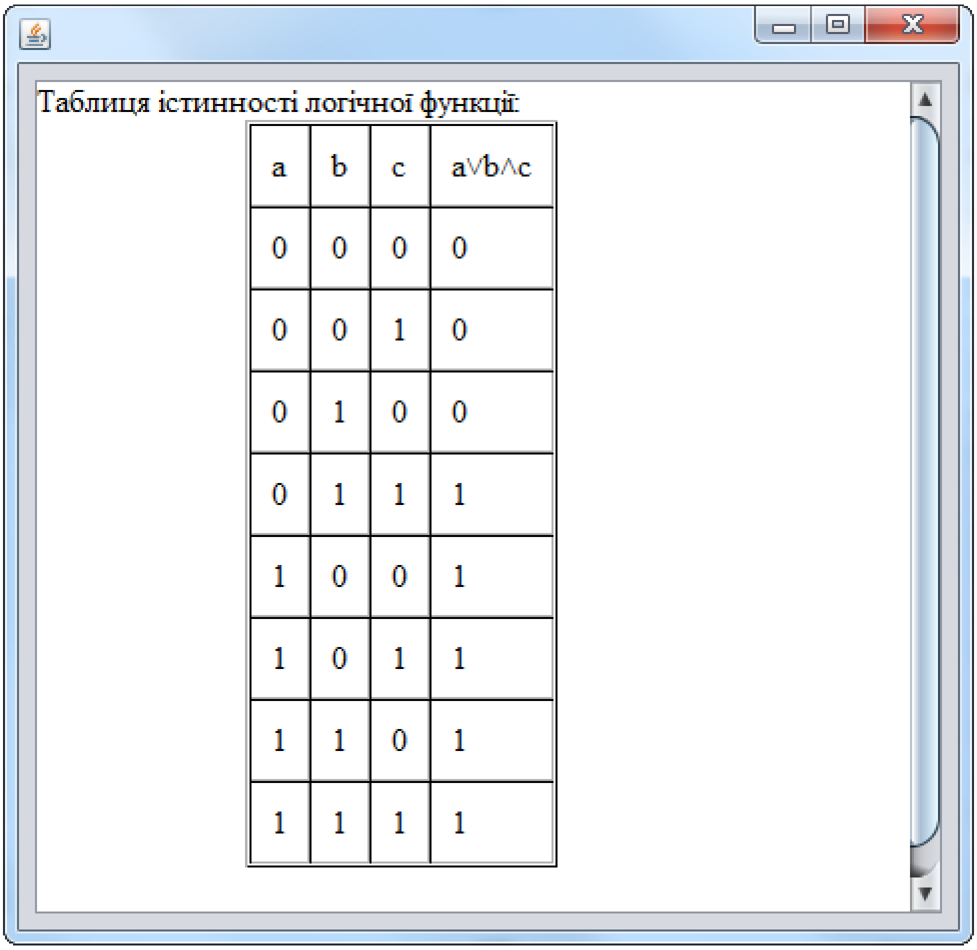


Рисунок 4.6 – Повідомлення про помилку англійською мовою

Якщо помилок не було знайдено, то проводиться обрахунок та виводиться результат (рис. 4.7).



Таблиця істинності логічної функції:

a	b	c	a∨b∧c
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Рисунок 4.7 – Результат

4.3. Інструкція по використанню тренажеру

Після запуску виводиться вікно калькулятора (рис. 4.2). По замовчуванню вибрано українську мову, щоб вибрати англійську слід в селекторі обрати «English» (рис. 4.8).

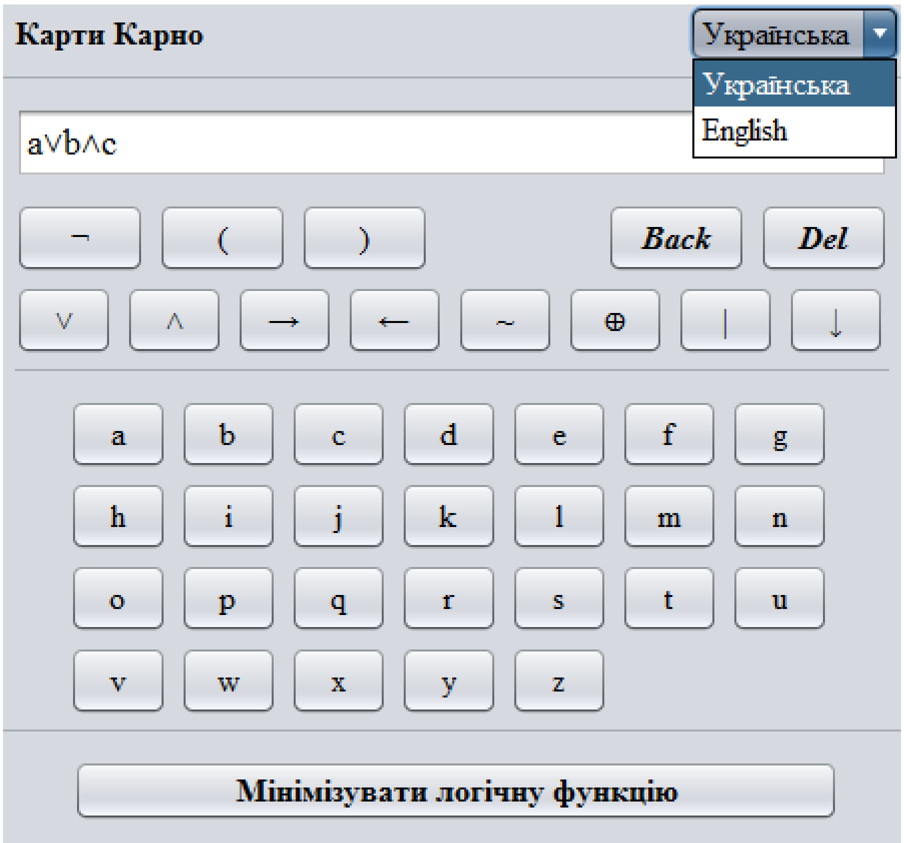


Рисунок 4.8 – Вибір мови

Потім необхідно ввести вираз і натиснути «Мінімізувати логічну функцію».

Якщо логічний вираз містить більше шести змінних, то виводиться повідомлення «За допомогою карт Карно легко вирішуються задачі мінімізації функцій з кількістю змінних до шести включно. При більшій кількості змінних пошук мінімальних форм запису функцій значно ускладняється і наочність карт Карно втрачається.» (рис. 4.9).

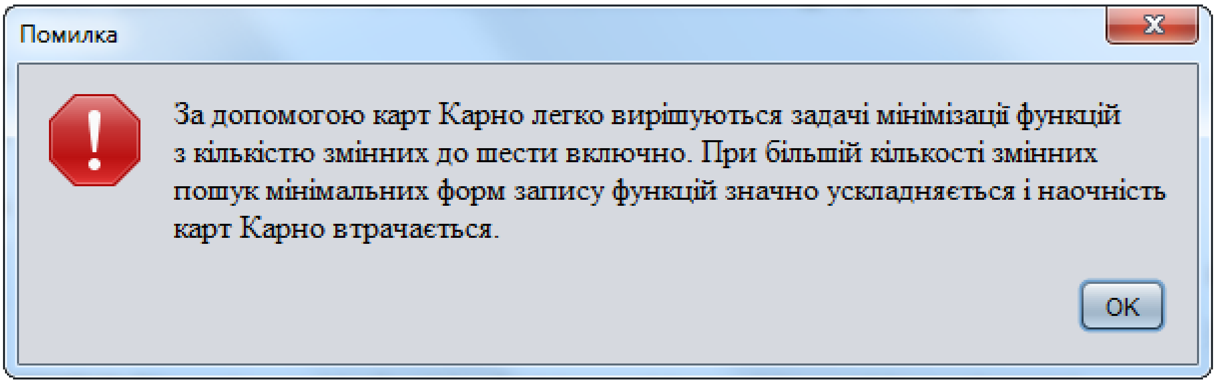


Рисунок 4.9 – Повідомлення, що логічний вираз містить більше шести змінних

Якщо вираз починається з логічного оператора, крім «Заперечення», то виводиться помилка: «Помилка! Вираз не може починатися з логічного оператора, крім «Заперечення».» (рис. 4.10).

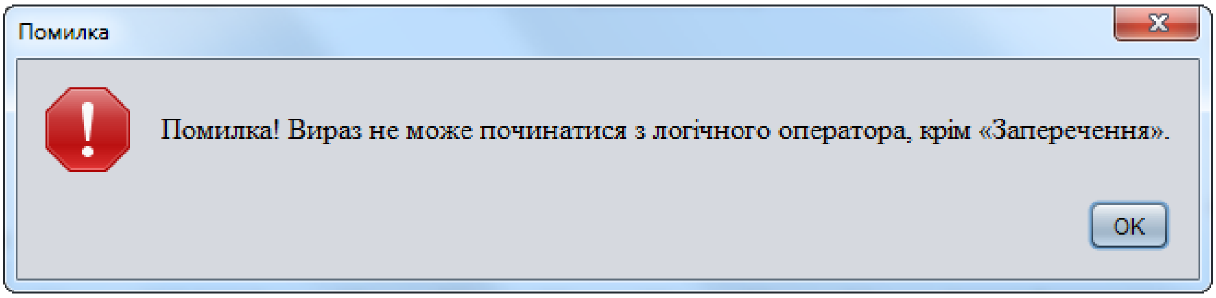


Рисунок 4.10 – Повідомлення, що вираз починається з логічного оператора

Якщо вираз містить декілька логічних операторів підряд, то виводиться помилка: «Помилка! Вираз не може містити декілька логічних операторів підряд» (рис. 4.11).

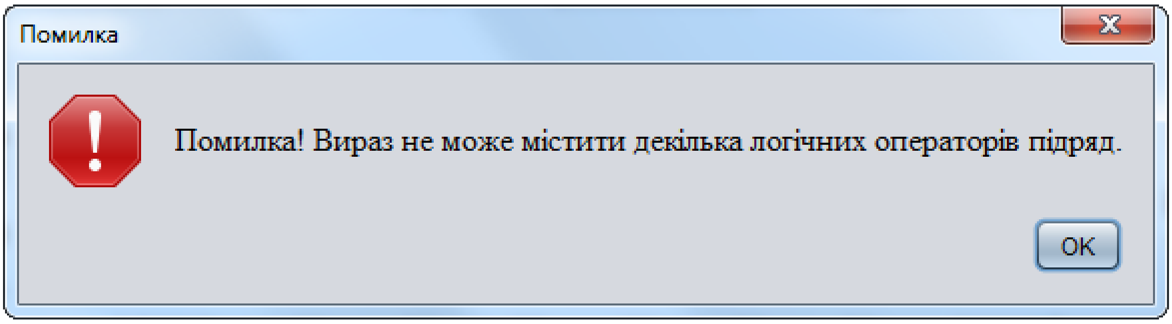


Рисунок 4.11 – Повідомлення, що вираз містить декілька логічних операторів підряд

Якщо вираз закінчується логічним оператором, то виводиться помилка: «Помилка! Вираз не може закінчуватися логічним оператором.» (рис. 4.12).

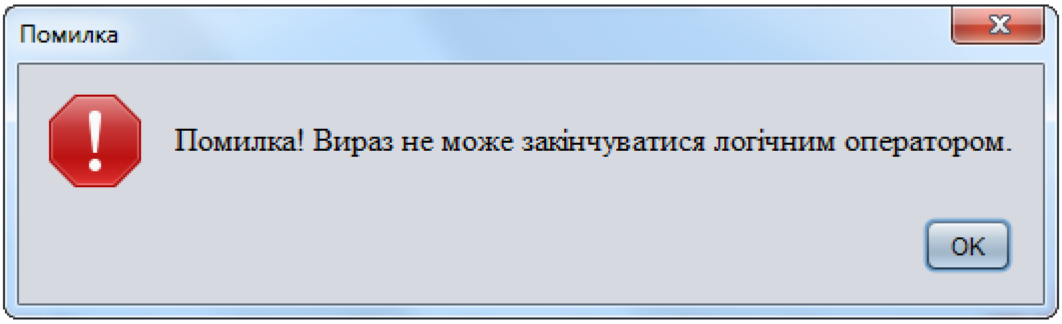


Рисунок 4.12 – Повідомлення, що вираз закінчується логічним оператором

Якщо вираз містить «дужки» і їх кількість непарна, то виводиться помилка: «Помилка! Вираз містить невірну кількість «дужок».» (рис. 4.13).

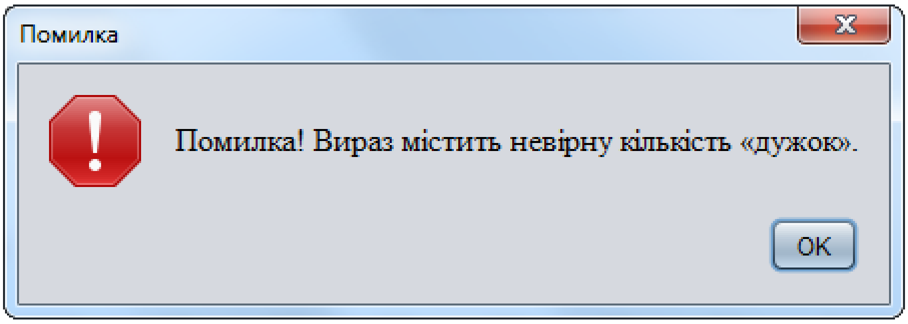


Рисунок 4.13 – Повідомлення, що вираз містить «дужки» і їх кількість непарна

Якщо ж вони неправильно розставлені, то «Помилка! Перевірте правильність розстановки «дужок».» (рис. 4.14).

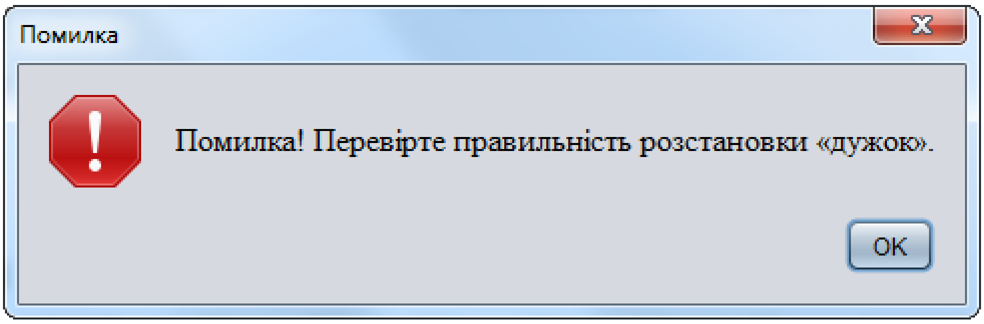


Рисунок 4.14 – Повідомлення, що «дужки» неправильно розставлені

Якщо вираз містить лише одну змінну, то виводиться помилка: «Помилка! Вираз має містити більше, ніж одну змінну.» (рис. 4.15).

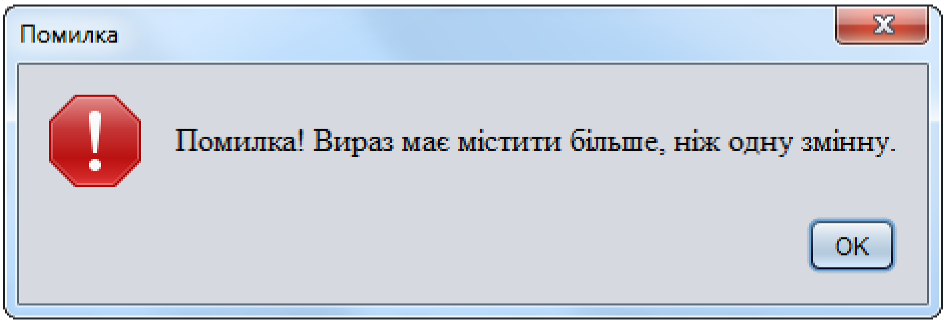


Рисунок 4.15 – Повідомлення, вираз містить лише одну змінну

Якщо проблем не виникає, то виводиться результат (4.7).

ВИСНОВКИ

Метою мінімізації є зменшення вартості технічної реалізації логічних функцій незалежно від використовуваних апаратних засобів.

Задача мінімізації – це задача неоднозначна, і різними шляхами можна отримати різні вирази мінімізованої функції, які відрізнятимуться між собою кількістю змінних і операцій над ними.

Результатом виконання є алгоритм програмного продукту, що реалізує калькулятор з теми «Мінімізація логічних функцій з допомогою карт Карно» дистанційного навчального курсу «Математична логіка» та сам калькулятор.

Основні результати роботи:

- розглянуто теоретичні відомості про мінімізацію логічних функцій з допомогою карт Карно;
- розглянуто специфіку застосування карт Карно;
- розроблено алгоритм калькулятора із застосування карт Карно для мінімізації логічних функцій;
- обргунтувано вибір програмного забезпечення.
- розроблено калькулятор;
- описано процес прогрманої реалізації;
- надано опис програми та інструкцію користувача.

Створено графічний вид калькулятора

- додано поле для вводу виразу;
- кнопки з операціями;
- кнопки із змінними;
- селектор для зміни мови;
- кнопка для перевірки введеного виразу та виведення результату.

При введенні логічного виразу користувачу відображається поле для введення виразу, поле з кнопками, та можливість виведення підказки.

Якщо під час обрахунків виникає помилка, то необхідно вивести повідомлення про дану помилку і її причину.

В результаті спочатку будується таблиця істинності логічної функції. Наступним кроком будується карта Карно для функції відповідної кількості змінних.

В програмі реалізовано можливість переключення між українською та англійською мовами.

Тренажер готовий до використання в дистанційному курсі «Математична логіка».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ємець О. О. Методичні рекомендації до виконання дипломної роботи для студентів ступеня магістра спеціальності 122 «Комп’ютерні науки» / О.О.(Олег) Ємець. – Полтава : РВВ ПУЕТ, 2018. – 35 с.
2. Ємець О. О. Методичні рекомендації щодо оформлення пояснювальних записок до курсових проектів (робіт) для студентів напряму підготовки «Інформатика» і спеціальності «Соціальна інформатика» / О. О. Ємець, Ол-ра О. Ємець. – Полтава : РВВ ПУЕТ, 2014. – 68 с.
3. Мінімізація логічних функцій: Аналітичний спосіб мінімізації. Мінімізація за допомогою карт Карно (діаграм Вейча). [Електронний ресурс] // Познайка.Орг - Сайт знаний. – Режим доступу: https://poznayka.org/s60683t1.html
4. Кондратенко Н.Р. Комп'ютерний практикум з математичної логіки. [Електронний ресурс] / Н.Р. Кондратенко // Навчальні посібники Вінницького національного технічного університету. Мережні електронні видання. – Режим доступу: https://web.posibnyky.vntu.edu.ua/fitki/6kondratenko_komp_praktikum_matlog/
5. Онлайн-сервіс «Онлайн инструменты по математической логике» [Електронний ресурс]. – Режим доступу: http://tablica-istinnosti.ru/ru/
6. Калькулятор «Таблица истинности» онлайн-сервісу «math.semestr.ru» [Електронний ресурс]. – Режим доступу: https://math.semestr.ru/inf/table.php
7. Игошин В. И. Математическая логика и теория алгоритмов : учеб. пособие для студ. высш. учеб. заведений / В. И. Игошин. — 2–е изд., стер. — М. : Издательский центр «Академия», 2008. — 448 с.

8. Преимущества Java [Електронний ресурс] // IBM®IBM Knowledge Center. – Режим доступу: https://www.ibm.com/support/knowledgecenter/ru/ssw_aix_71/performance/advantages_java.html
9. Битва титанов open-source: NetBeans или Eclipse [Електронний ресурс]. – Режим доступу: https://webformyself.com/bitva-titanov-open-source-netbeans-ili-eclipse/

ДОДАТОК А. КОД ПРОГРАМИ