НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ім. Ігоря Сікорського»
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

КУРСОВА РОБОТА

з дисципліни

Програмування та алгоритмічні мови

на тему:___Розв`язок систем рівнянь з одним невідомим в алгебрі множин

Студента 1 курсу групи КА-02
Галузь знань 12 Інформаційні технології

__Тункіна Євгена Андрійовича_____

Керівник _Старший викладач Назарчук Ірина Василівна

Національна оцінка _____
Кількість балів: _____Оцінка: ECTS _____

Члени комісії _____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)
_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)
_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали

Київ - 2021рік

НТУУ „КПІ ім.І.Сікорського” ІПСА
(назва вищого закладу освіти)

Кафедра	***математичних методів системного аналізу***

Дисципліна	***Програмування та алгоритмічні мови***

Галузь знань	***12 Інформаційні технології***

Курс	***перший***	Група	***КА—02***	Семестр	***другий***

ЗАВДАННЯ
на курсовий проект(роботу) студента

Тункін Євген Андрійович
(прізвище, ім’я, по батькові)

1. Тема проекту(роботи)	Розв`язок систем рівнянь з одним невідомим в алгебрі множин

2. Строк здачі студентом закінченого проекту(роботи)	***21.05.2021 р.***

3. Вихідні дані до проекту(роботи)	Реалізувати введення системи рівнянь як символьні
рядки. Реалізувати синтаксичний аналіз введених рядків та розв`язання системи рівнянь	
відносно невідомої множини із виведенням проміжних результатів. Відповідь надати у	
вигляді символьного рядка. Проект виконати із використанням заголовкового файлу	
windows.h.	

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)	
1. Постановка задачі.	
2. Метод розв’язку задачі	
3. Загальна блок-схема алгоритму та опис алгоритму	
4. Опис програмного продукту.	
5. Результати роботи.	
6. Висновки.	
7. Список літератури.	
Додаток А. Текст програми.	

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)	
1. Загальна блок-схема алгоритму.	
2. Ілюстрації роботи програми.	

6. Дата видачі завдання	16.02.2021

КАЛЕНДАРНИЙ ПЛАН

№/п	Назва етапів курсового проекту (роботи)	Строк виконання етапів роботи	Примітка
1.	Вибір теми курсової роботи. Опрацювання відповідної літератури. Оформлення листа Завдання.	20.02.2021	
2.	Аналіз постановки задачі.	27.02.2021	
3.	Вибір та дослідження методів, вибір відповідних структур даних, розробка алгоритму. Перше узгодження з керівником.	05.03.2021	
4.	Проектування інтерфейсу.	12.03.2021	
5.	Друге узгодження з керівником.	19.03.2021	
6.	Програмна реалізація.	26.03.2021	
7.	Демонстрація першого варіанту. Трете узгодження з керівником.	02.04.2021	
8.	Заключне тестування програми.	16.04.2021	
9.	Аналіз результатів. Оформлення звіту.	23.04.2021	
10.	Захист та демонстрація курсової роботи.	14.05.2021 – «A» 21.05.2021 –«B,C» 29.05.2021 – «D,E»	

Студент		
	(підпис)	

Керівник		
	(підпис)	(прізвище, ім’я, по батькові)

(дата)	

Анотація

Курсова робота є прикладом програмної реалізації універсального методу розв`язку системи рівнянь з одним невідомим в алгебрі множин. Для реалізації алгоритму була використана математична модель представлення рівняння у вигляді тернарного дерева.

Для реалізації методу була обрана мова програмування C++. Інтерфейс головного меню був виконаний за допомогою бібліотеки «windows.h». та елементів псевдографіки.

Продукт курсової роботи призначений для навчальних цілей. Даний продукт допомагає у вивченні такого розділу дискретної математики, як «теорія множин». Програмний продукт може бути корисним в першу чергу для студентів першого курсу технічних спеціальностей.

Annotation

The course work is an example of software implementation of a universal method for solving a system of equations with one unknown in the algebra of sets. To implement the algorithm, a mathematical model of representation of the equation in the form of a ternary tree was used.

The C ++ programming language was chosen to implement the method. The main menu interface was made using the "windows.h" library. and elements of pseudography.

The product of the course work is intended for educational purposes. This product helps in the study of such a section of discrete mathematics as "set theory". The software product can be useful primarily for first-year students of technical specialties.

ЗМІСТ

ВСТУП

Ні для кого не секрет, що точні науки, для яких математика є основою, мають найбільш важливе місце у розвитку техніки та людства в цілому. Саме через це їх викладанню у багатьох закладах середньої та вищої освіти приділяється значна увага. Таким чином, тема цієї курсової роботи є актуальною у наш час, тому що задача розв`язання системи рівнянь з одним невідомим в алгебрі множин є типовою задачею дискретної математики.

Метою цієї курсової роботи було знайти, покращити та реалізувати алгоритм розв`язку системи рівнянь з одним невідомим в алгебрі множин, після чого створити консольну програму із дружнім користувацьким інтерфейсом: необхідно було створити головне меню із можливостями керування за допомогою клавіш, видалити введені рівняння, перейти до сторінки допомоги або вийти з програми. Інтерфейс був реалізований у алфавітно-цифровому режимі консольного вікна із використанням бібліотеки «windows.h».

На основі аналізу літературних джерел у якості алгоритму був обраний універсальний метод розв`язку системи рівнянь з одним невідомим в алгебрі множин, для реалізації якого була розроблена математична модель.

Розроблений програмний продукт може стати у нагоді будь-кому, хто має відношення да даної дисципліни: по-перше, студенти отримують можливість не тільки запам`ятати закони алгебри множин, а й побачити їх використання на практиці; по-друге, викладачі також отримують користь: за допомогою програмного продукту вони зможуть пришвидшити процес перевірки розрахункових та контрольних робіт та знаходження помилки у студентських розв`язках.

При написанні курсової роботи було використане середовище розробки Dev C++ Version 5.11 та операційна система Windows 10 21H1. Для оформлення документації до розробленого програмного продукту були

використані графічні редактори Paint та Gimp 2.10.22, а також текстовий редактор Microsoft Word 2019.

Робота складається із вступу, трьох розділів, висновків, додатку та списку використаних джерел.

РОЗДІЛ 1 Постановка задачі

1.1 Огляд існуючих підходів до розв'язання поставленої задачі

Проблема розв`язання системи рівнянь в алгебрі множин постала після формування теорії множин у наївному вигляді німецьким математиком Георгом Кантором у 19-му столітті. Чіткого визначення поняття «множина» не існує, проте, є «наївне» означення, що множина – це довільний набір об`єктів, що попарно відрізняються. Не дивлячись на те, що наведене визначення призводить до парадоксів, «наївна» теорія множин, що базується на ньому цілком придатна для розв`язання широкого класу прикладних проблем.

У теорії множин визначені відношення між множинами, такі як рівність та вкладення. Також існує декілька базових операцій над множинами: об`єднання, переріз та доповнення. За допомогою них визначаються операції різниці та симетричної різниці. Аналогічно до алгебри висловлень, у теорії множин існує чотири базові закони над цими операціями, з яких виводяться ще декілька.

З визначення операцій випливає декілька еквіваленцій, які знадобляться нам для перевірки відповіді.

Наше завдання полягає у тому, щоби маючи пари рівних множин, сформованих скінченою кількістю операцій над скінченною кількістю відомих множин та однією невідомою множиною, сформувати умови до невідомої множини, при виконанні яких можливо було довести рівність вихідних множин за допомогою наведених властивостей.

Першим засобом, що в деяких випадках дозволяє «вгадати» відповідь є діаграми Венна або круги Ейлера. Ці діаграми наочно ілюструють результати виконання операцій в алгебрі множин.

На діаграмі Венна універсальну множину зображують у вигляді прямокутника, кожну іншу множину – у вигляді круга. Якщо відомо, що

множини не перерізаються, відповідні круги зображують такими, що не мають спільних точок. Якщо відомо, що множина А міститься в множині В, круг множини А зображують всередині круга множини В. Якщо апріорі нічого не відомо про взаємне положення множин, відповідні круги зображують такими, що перерізаються, та кожен круг не лежить цілком в середині іншого.

Головним недоліком цього способу є те, що він не доводить тотожність, а тільки ілюструє її, отже при вирішенні реальних задач цим алгоритмом користуватись не потрібно. Також при збільшенні кількості множин різко зростає кількість варіантів їх взаємного розташування, отже складність малюнків та розпізнавання відповіді збільшується також.

Ще одним підходом до розв`язання поставленої задачі є універсальний метод розв`язку системи рівнянь з одним невідомим в алгебрі множин. Цей спосіб полягає у виділенні множин, що не перерізаються з невідомою множинною, перерізаються з невідомою множинною або перерізаються з її доповненням. Це відбувається за допомогою використання законів алгебри множин на заданих рівняннях та властивостей операцій над множинами.

Розглянемо загальний метод розв`язку рівняння вигляду

$$\mathcal{A}_1(X) = \mathcal{A}_2(X) \ (1)$$

1. Використовуючи еквівалентність

$$(A = B) \Leftrightarrow (A \triangle B = \emptyset),$$

Зводимо рівняння (1) до рівняння з порожньою множиною у правій частині.

$$\mathcal{A}(X) = \emptyset, \text{де } \mathcal{A}(X) = \mathcal{A}_1(X) \triangle \mathcal{A}_2(X)$$

2. За допомогою дистрибутивності та винесення за дужки зобразимо це рівняння у вигляді

$$(\mathcal{B}_1 \cap X) \cup (\mathcal{B}_2 \cap X^c) \cup \mathcal{B}_3 = \emptyset,$$

Де $\mathcal{B}_i$ – формули, що не містять входжень літери X.

Тоді відповіддю до рівняння буде

$$\text{Якщо } \mathcal{B}_3 = \emptyset, \text{то } \mathcal{B}_2 \subset X \subset \overline{\mathcal{B}_1}. \text{Інакше розв`язків немає.}$$

У відповіді до системи рівнянь X містить об`єднання всіх лівих частин відповідей до рівнянь та міститься у перетині всіх правих частин.

У розробленому продукті був використаний саме цей спосіб через його універсальність та чітку послідовність дій.

1.2 Уточнена постановка задачі на розробку програмного забезпечення

Постановка задачі полягає у створенні консольної програми, за допомогою якої користувач може отримати відповідь до заданої системи рівнянь в алгебрі множин з одним невідомим та побачити процес розв`язку із теоретичними поясненнями та обґрунтуванням. Під відповіддю мається на увазі дві множини **A** та **B**, що задовольняють умові «деяка множина є розв`язком системи рівнянь тоді і тільки тоді, коли множина **A** міститься у цій множині та множина **B** містить цю множину». Теоретичними поясненнями є опис кожного кроку його метою та назвою закону, за яким він відбувається.

Найважливішою частиною розробки програмного продукту є реалізація універсального методу розв`язку системи рівнянь з одним невідомим в алгебрі множин, модифікованого використанням розв`язків поточного рівняння у майбутніх рівняннях, що суттєво спростить розв`язок та зробить його більш швидким та зрозумілим.

Введення рівнянь планується зробити у вигляді текстових рядків за допомогою стандартних функцій введення текстового рядка. Аналогічно відповідь становитиме собою текстовий рядок, у якому множини будуть розташовані за відношенням вкладення.

Потрібно передбачити «крайні випадки» у відповідях, наприклад, якщо невідома множина міститься у порожній множині, то система може мати на більше одного розв`язку – тільки порожню мноожину.

Для зручності користувача буде створене головне меню, яке міститиме у собі можливість видалення системи, виходу з програми та переходу до сторінки з інструкцією користувача, що складатиметься з деяких теоретичних відомостей, умовних позначень операцій над множинами, списку клавіш

навігації та правил коректного введення рівнянь. Переміщення між пунктами головного меню здійснюватиметься за допомогою клавіш KeyLeft та KeyRight, а вибір пункту за допомогою клавіши Enter.

Також планується зробити програму захищеною від введення некоректних даних із сповіщенням про це.

РОЗДІЛ 2 Розробка програмного продукту

2.1 Метод розв'язку задачі

Для написання даної програми була використана мова C++ та бібліотека «windows.h».

Кожне рівняння системи розв`язується окремо. Для цього створений глобальний масив символів, у нульовий рядок якого записується множина, що отримана симетричною різницею правої та лівої частини вихідного рівняння. Оскільки із властивості симетричної різниці ця множина еквівалентна порожній множині, нас не цікавить права частина, і ми можемо аналізувати тільки отриманий вираз.

Наступний крок полягає у тому, щоби виділити зовнішню операцію та двох або одного (у випадку доповнення) виразів, до яких вона застосовується. Ця дія повторно застосовується до кожної з отриманих множин, допоки довжина всіх вихідних виразів не буде дорівнювати 1, тобто це позначатиме, що операції застосовуються до пропозиційних літер.

На кожному такому кроці всі отримані вирази зберігаються в масив за принципом тернарного дерева для зручного доступу до них за допомогою індексу результуючої множини. Наприклад, отримавши вираз

$$\overline{(A \cap B)} \triangle \big((C \cup X) \setminus E\big)$$

Програма представляє його, як вказано на рисунку 2.1.

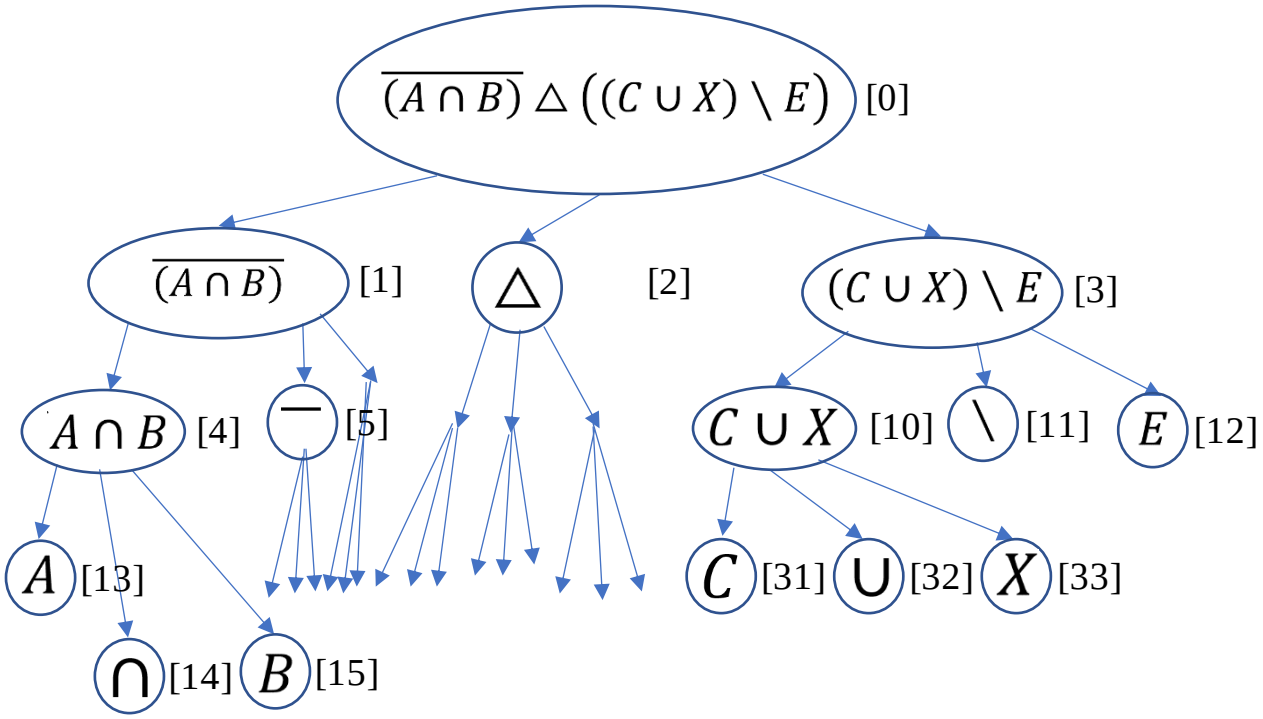

Рис. 2.1 Збереження рівняння у програмі

Завдяки такій структурі, маючи індекс множини i, ми можемо отримати зовнішню операцію у цій множині за індексом 3*i + 2 та множини, до яких вона застосовується за індексами 3*i + 1 та 3*i + 3.

За алгоритмом розв`язку, наступним кроком потрібно замінити операції симетричної різниці, різниці та доповнення до складної множини за допомогою операцій об`єднання та перерізу. За наведеною схемою, рядки з операціями мають індекс, що ділиться на 3 з остачею 2, тому перевірку можливо здійснювати циклом за індексом виду 3*i + 2.

Ідея полягає в тому щоби знайшовши операцію, яку потрібно замінити, написати, використовуючи множини, до яких вона застосовується, її батьківський елемент за відомою формулою, а потім перейти до батьківського елементу та сформувати рядок, що знаходиться ще на рівень вище, і, повторюючи цю дію, дійти до нульового рядка, Після чого заповнити дерево отриманим рядком, в якому вже не буде цієї операції, та почати перевірку спочатку. Для більш простого розуміння, наведу приклад.

Наприклад, замінюємо різницю за формулою

$$A \setminus B = A \cap \bar{B}$$

Отриманий індекс буде дорівнювати 11. Оскільки рядок з операцією є середнім з дочірніх, знайти індекс батьківського елементу можна націло поділивши 11 індекс на 3. Отримаємо 3. Тоді за формулою ми маємо записати у третій рядок $([10]) \cap (\overline{[12]})$. Очевидно, що 10 та 12 – індекси множин, до яких застосовувалася різниця, отримані додаванням та відніманням одиниці до індексу операції. Таким чином, у [3] буде записано $(C \cup X) \cap \bar{E}$.

Наступним кроком потрібно «зібрати догори» дерево за рядком з індексом 3. Спочатку підрахуємо індекс батьківського елементу, поділивши націло i – 1 на 3, де i – індекс рядка, з якого потрібно зібрати дерево, в нашому випадку це 3. Індекс батьківського елементу дорівнює 0. Тоді знайдемо індекс меншого з його дочірніх елементів за формулою 3*j + 1, де j – індекс батьківського елементу, у нашому випадку 0. Знайдений індекс дорівнює 1, і, оскільки він менший з дочірніх елементів, ми знаємо, що [1] та [3] – множини, до яких застосовується операція у рядку [2]. Отже, не аналізуючи вмісту рядків, у [0] записуємо ([1])[2]([3]). В даному випадку це $\overline{(A \cap B)} \triangle (A \cap \bar{B})$. Якщо б індекс рядка, у який ми це записали, не дорівнював би 0, дія з цього абзацу була б повторена необхідну кількість разів до зміни нульового рядка.

Далі дерево заповнюється за тією ж самою схемою з малюнка 2.1 та аналізується на операції, що потрібно замінити, допоки вони не скінчаться.

У отриманому потрібно за допомогою дистрибутивності зробити операції об`єднання зовнішніми, тобто аргументами перерізів повинні бути або пропозиційні літери, або множини, сформовані лише перетинами. Перевірку виконуємо таким самим чином, але тепер умовою заміни є те, що рядок з індексом і дорівнює «∩», а рядок з індексом $3 * (i - 1) + 2$ або $3 * (i + 1) + 2$ дорівнює «∪». Не складно перевірити, що рядки саме з такими індексами містять зовнішні операції множин, до яких застосовується поточна операція, тобто яка під індексом і. Заміна відбувається таким самим чином. У $[i/3]$ записується $(([i + 1]) \cap ([3 * (i - 1) + 1])) \cup (([i + 1]) \cap ([3 * (i - 1) + 3]))$, якщо $3 * (i - 1) + 2 =$ «∪», або $(([i - 1]) \cap ([3 * (i + 1) + 1])) \cup$

$\left(([i-1]) \cap ([3*(i+1)+3])\right)$, якщо $3*(i+1)+2 =$ «∪». Потім збираємо дерево з рядка i / 3 та повторюємо ці дію поки присутні зовнішні перетини.

Оскільки переріз асоціативний, у отриманому рівнянні можна прибрати усі дужки окрім тих, що стоять поряд із об`єднанням. Тоді проаналізуємо кожну отриману дужку на входження X або $\bar{X}$. Якщо дужка містить X та $\bar{X}$, вона видаляється. Якщо ні, то весь її текст окрім X копіюється у окремий відповідний рядок для множин, що не взаємодіють з X, взаємодіють з X, або взаємодіють з $\bar{X}$. Таким чином, з`єднавши назад ці рядки, ми винесли X та $\bar{X}$ за дужки.

Відповідь до розглянутого рівняння отримано. X містить множину, що взаємодіє з $\bar{X}$ та міститься у доповненні до множини, що взаємодіє з X.

2.2 Алгоритм розв'язку задачі

Робота програми починається з можливості вибору користувачем певного пункту головного меню, яке було реалізоване за допомогою функцій бібліотеки «windows.h». Меню містить п`ять пунктів: «Add equation», «Solve system», «Clear», «Help», «Exit». Вибір певного пункту меню користувачем відбувається за допомогою клавіши Enter. Загальна структурна блок-схема роботи програми наведена на рисунку 2.2.

При виборі користувачем пункту «Add equation», консольне вікно очищується, та користувачу надається можливість ввести рівняння із використанням умовних позначень операцій над множинами, список яких можна отримати, обравши пункт «Help» у головному меню. Після натискання клавіши Enter, введене рівняння аналізується на коректність за допомогою функції IsCorrect. Якщо воно було введено правильно, то це рівняння вноситься до символьного масиву рівнянь та з`являється під написом «Your system:» під головним меню. Якщо – ні, то виводиться напис «The equation was entered incorrectly», та після натискання клавіши Enter, програма повертається до головного меню.

При виборі користувачем пункту «Solve system», за допомогою серії викликів функцій fill, connect, unite та zakony, послідовно розв`язується кожне рівняння системи та виводиться відповідь до неї. Блок-схема роботи пункту меню «Solve system» наведена на рисунку 2.3.

При виборі користувачем пункту «Clear», видаляються всі введені рівняння та очищуються всі допоміжні масиви: масив розв`язків рівняння, масив глобальних розв`язків, масив порожніх множин та інші. Цей пункт меню використовується користувачем при помилці у введеному рівнянні або для введення наступної системи.

При виборі користувачем пункту «Help», у очищене консольне вікно виводиться вміст файлу «help.txt», що знаходиться у папці з програмою. Цей файл містить список умовних позначень операцій над множинами, правила коректного введення рівнянь та коротку інструкцію користувача. Виведення відбувається за допомогою заголовкового файлу <fstream> та об`єкту класу ifstream.

При виборі користувачем пункту «Exit», відбувається закриття потоків та вихід з програми.

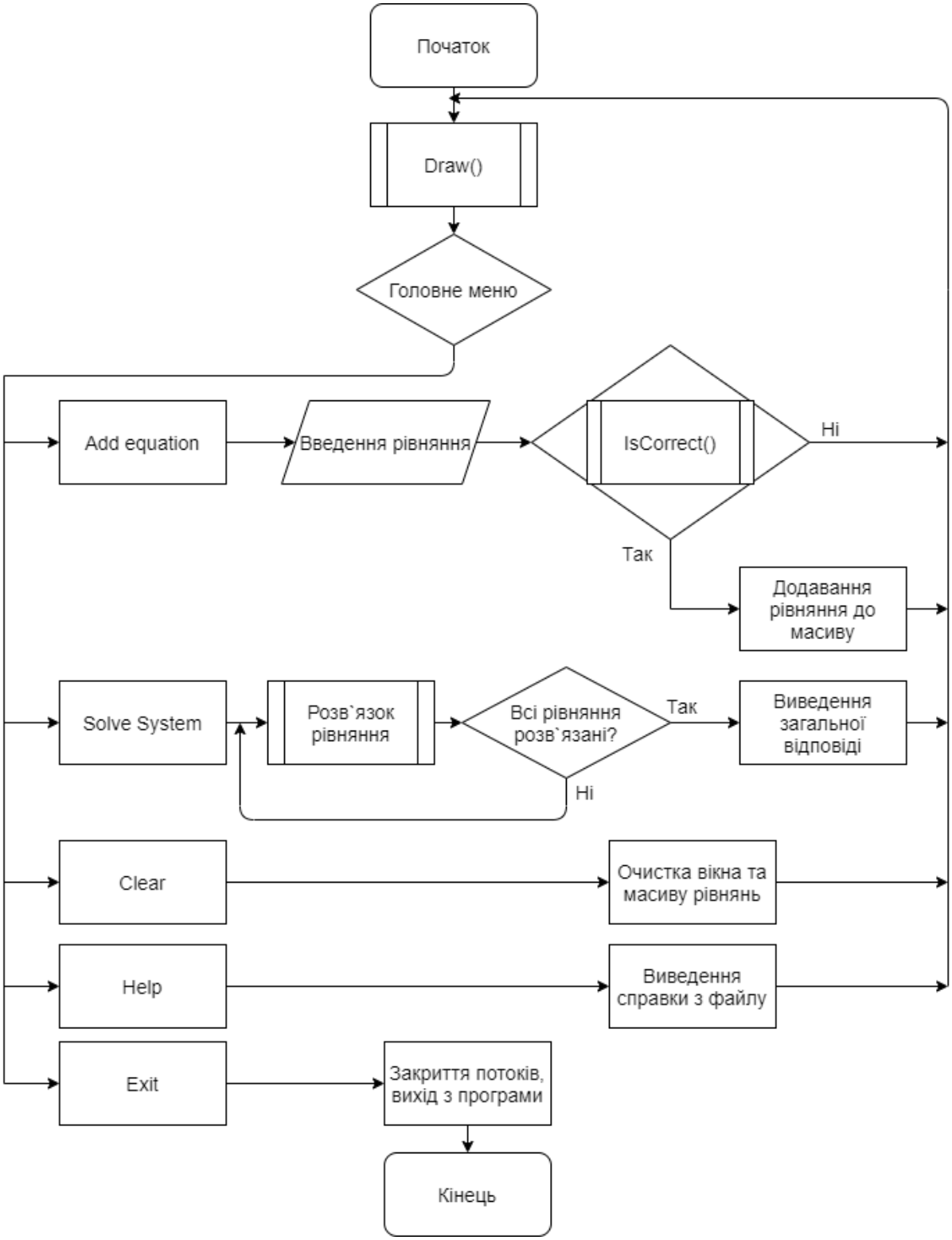

Рис. 2.2 Загальна структурна блок-схема роботи програми

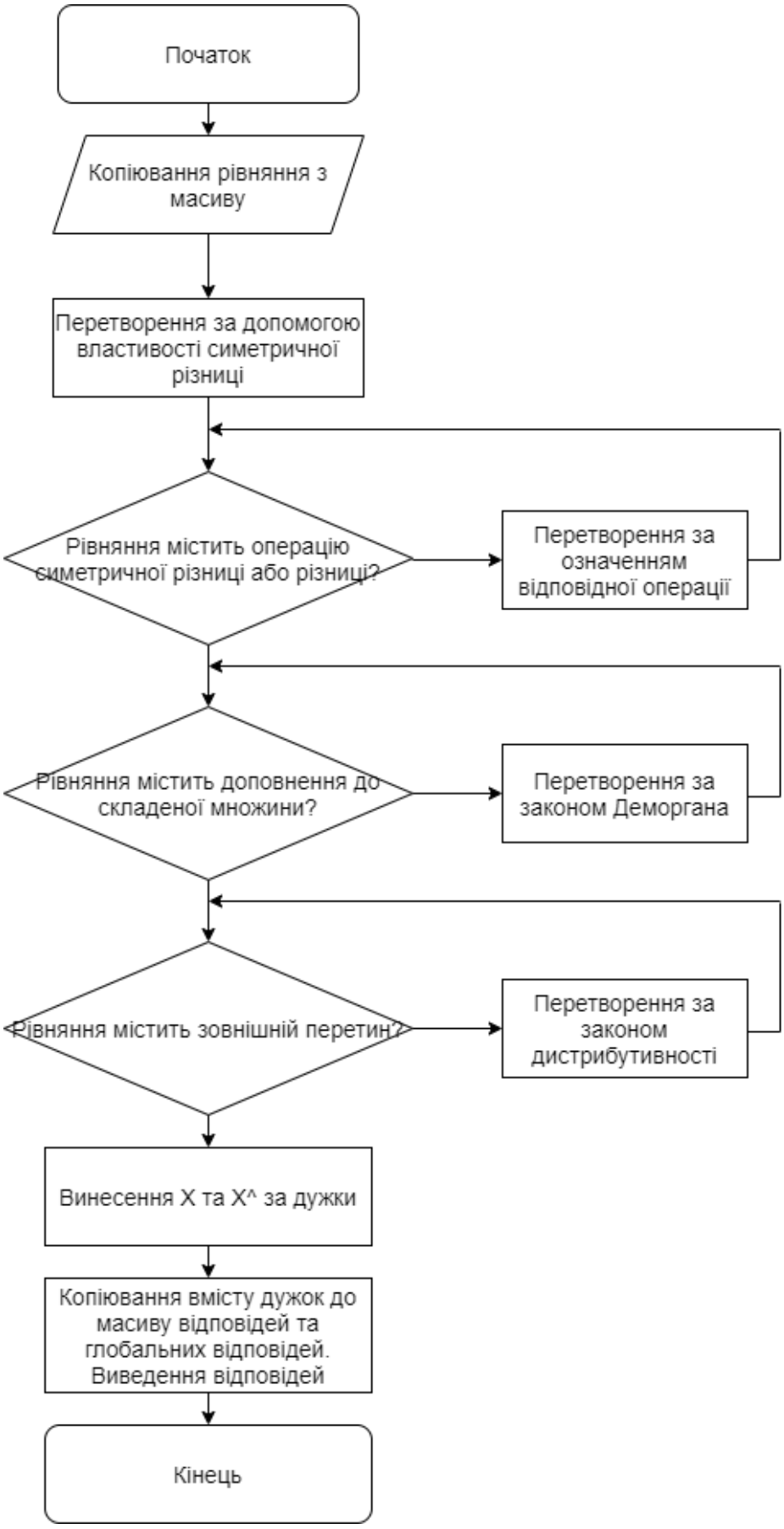

Рис. 2.3 Розв`язок рівняння

РОЗДІЛ 3 Опис розробленого програмного продукту

3.1 Опис головних структур і змінних програми

Таблиця 3.1. Опис головних змінних програми

№	Змінна	Тип змінної	Призначення змінної
1	m[]	char*	Містить текст головного меню
2	hstdout	HANDLE	Дескриптор консольного вікна, який використовується для виведення головного меню
3	act	int	Номер активного пункту меню, що змінюється при натисканні користувачем клавіш навігації
4	equations[][]	char	Масив рядків, кожен з яких містить введене рівняння
5	x[][]	char	Глобальна змінна. Масив рядків, у якому відбувається розв`язок поточного рівняння
6	answers[][][]	char	Масив з трьох рядків, що містить множини поточного рівняння, відокремлені за взаємодією з невідомою множиною
7	globalAns[][]	char	Аналогічний до answers масив, що містить сукупність аналогічних множин всіх розв`язаних рівнянь
8	EmptyMn[][]	char	Масив, що містить множини, про які стало відомо, що вони порожні
9	nskobok	int	Містить кількість відкритих скобок, за допомогою чого фіксується зовнішня операція, коли ця кількість дорівнює 0
10	nEquation	int	Містить кількість введених рівнянь у систему для копіювання наступного коректно введеного рівняння у потрібне місце масиву рівнянь

3.2 Опис головних функцій програми

Таблиця 3.2. Опис головних функцій програми

№	Синтаксис	Опис
1	int main()	Головна функція програми. Здійснює вибір пункту головного меню та викликає функції, потрібні для розв`язку системи
2	void draw(HANDLE h, COORD c, int k, int n, int len)	Виводить на екран головне меню із активним пунктом, номер якого переданий в якості параметра
3	void getparts(int xx)	Отримує номер рядка в x[][]. Для цього рядка знаходить зовнішню операцію та множини, до яких вона застосовується. Заповнює ними дочірні рядки для отриманого рядка
4	void fill(int h)	Шляхом виклику функції getparts(int) послідовно для кожного рядка в x[][], заповнює тернарне дерево для виразу у нульовому рядку x[][]
5	void connect(int i)	Використовується при заміні операцій або використанні закону дистрибутивності. Для отриманого номеру зміненого рядка переписує його батьківській рядок для того, щоб оновити його. Якщо нульовий рядок не було змінено, викликає себе ще раз
5	void unite(int j)	Формує рівняння під вказаним номером із використанням відповідних множин в answers[][][]
6	void zakony(int j)	Спрощує відповіді до вказаного рівняння за допомогою законів та тотожностей алгебри множин. Виводить назви використаних законів

Продовж. табл. 3.2.

7	bool IsCorrect(char *p)	Аналізує введене рівняння на коректність. Викликається при виборі користувачем першого пункту головного меню

3.3 Опис інтерфейсу

На початку роботи програми у консольному вікні відображається головне меню, яке містить в собі 5 пунктів. Керування у меню реалізоване за допомогою клавіш KeyLeft та KeyRight, вибір – за допомогою клавіші Enter. Меню показано на рисунку 3.1.

Рис. 3.1 Головне меню

Якщо користувач обирає перший пункт меню, консольне вікно очищується та на нього для зручності виводиться список умовних позначень операцій над множинами. Після цього користувачу надається можливість ввести рівняння. Це показано на рисунку 3.2.

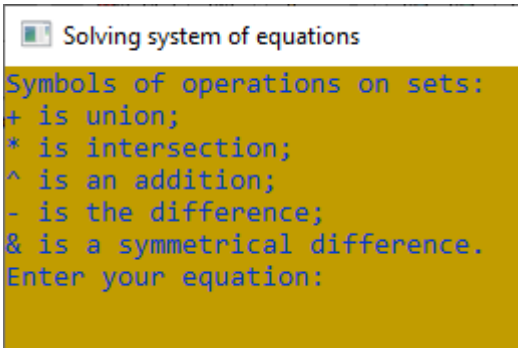

Рис. 3.2 Введення рівняння

Після натискання клавіши Enter, якщо рівняння було введено коректно, програма повертається до головного меню і воно відображається під написом

«Your system:». Якщо ні, буде виведено «The equation was entered incorrectly» та після натискання клавіши Enter, програма повернеться до головного меню. Приклад наведено на рисунку 3.3.

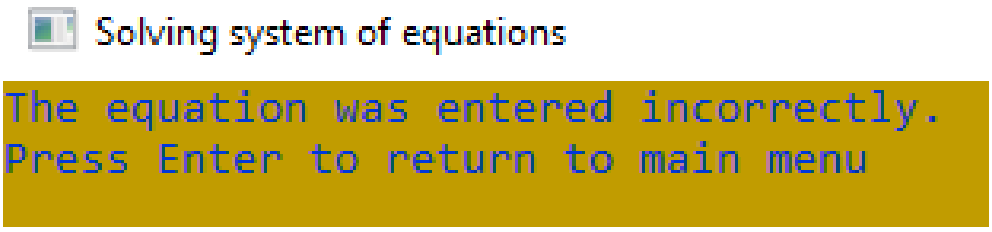

Рис. 3.3. Сповіщення про некоректно введене рівняння

Якщо користувач обирає другий пункт меню, введена система вирішується та виводиться відповідь. Приклад наведено на рисунку 3.4.

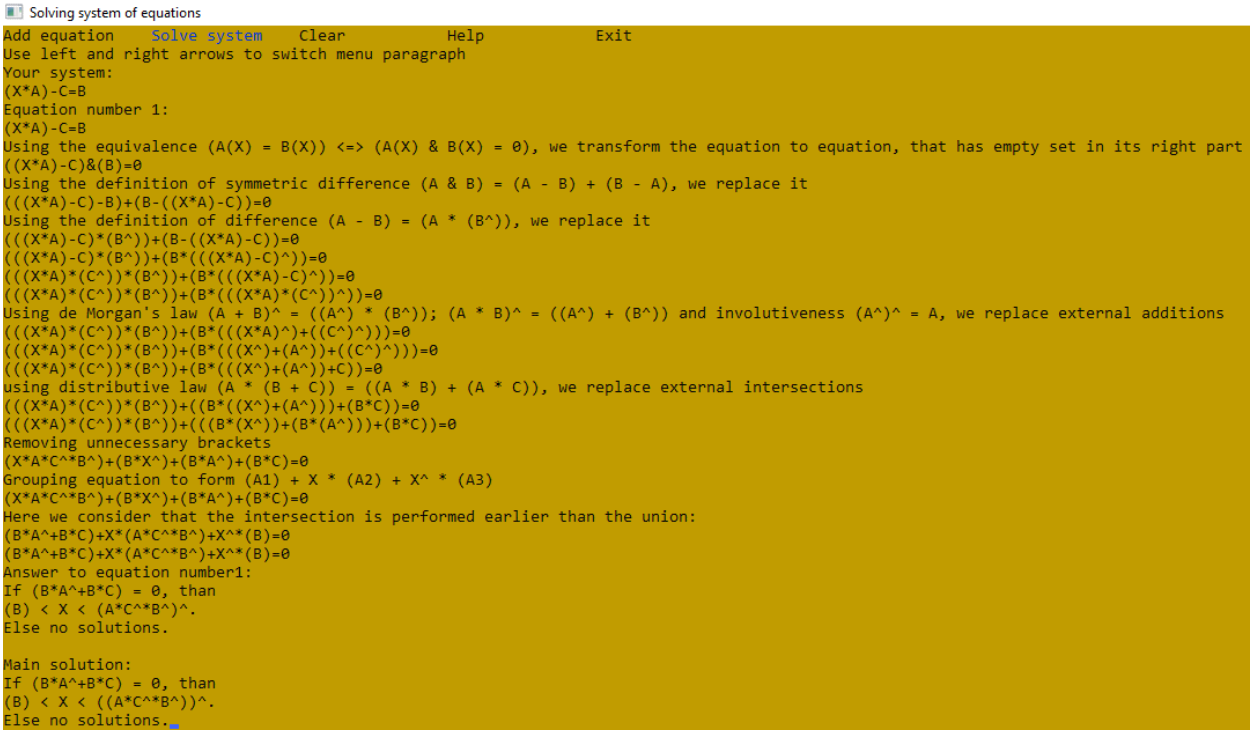

Рис. 3.4 Розв`язок системи з одного рівняння

Якщо користувач обирає третій пункт меню, видаляються всі введені рівняння та очищуються всі допоміжні масиви: масив розв`язків рівняння, масив глобальних розв`язків, масив порожніх множин та інші. Цей пункт меню використовується користувачем при помилці у введеному рівнянні або для введення наступної системи.

Якщо користувач обирає четвертий пункт меню, у очищене консольне вікно виводиться вміст файлу «help.txt», що знаходиться у папці з програмою.

Цей файл містить список умовних позначень операцій над множинами, правила коректного введення рівнянь та коротку інструкцію користувача. Приклад наведено на рисунку 3.5.

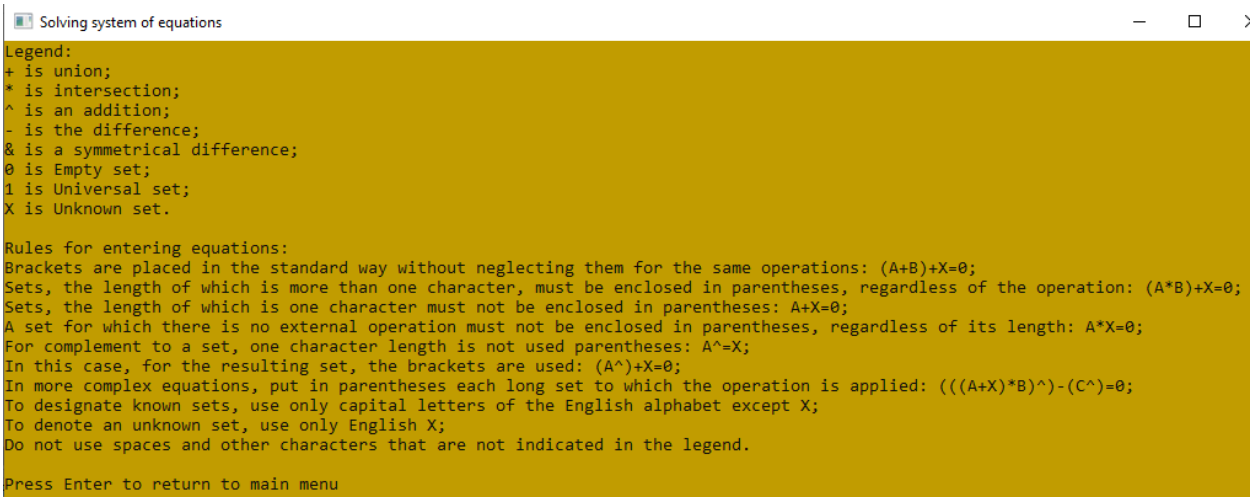

Рис 3.5 Сторінка допомоги

Якщо користувач обирає п`ятий пункт меню, відбувається закриття потоків та вихід з програми.

3.4 Результати роботи програмного продукту

Робота алгоритму завершується виведенням проміжних результатів розв`язку системи, формулювань законів, за якими була зроблена та чи інша дія та відповіді до системи рівнянь.

Використання означення симетричної різниці – еквіваленції, що застосовується найпершою під час розв`язку будь-якого рівняння в алгебрі множин наведено на рисунку 3.6.

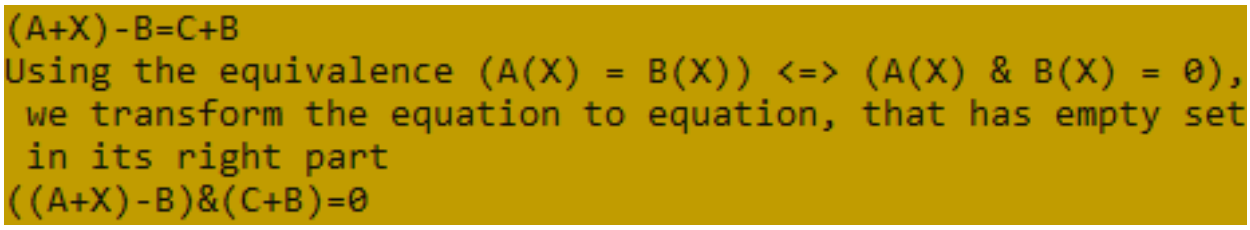

Рис. 3.6 Означення симетричної різниці

Важливою перевагою розробленого алгоритму порівняно із базовим є відслідковування множин, що повинні бути порожніми для того, щоб система мала розв`язки. Такі множини замінюються у рівнянні нулем для його швидшого та простішого розв`язку. Також порожні множини замінюються

нулем у вже розв`язаних та ще не розв`язаних рівняннях, після чого програма повертається до поточного рівняння. Приклад наведено на рисунку 3.7.

```
(C^*B^*A+C*B+B)+X*(C^*B^)+X^*(C*A^+B*A^)=0
In the bracket that does not contain X we see that B is 0, so we can replace it
(C^*0^*A+C*0+0)+X*(C^*0^)+X^*(C*A^+0*A^)=0
Also we can replace it in solved equations:
Also we can replace it in not solved equations:
Equation number 2
(X*A)-C=0
Returning to our equation:
(C^*0^*A+C*0+0)+X*(C^*0^)+X^*(C*A^+0*A^)=0
```

Рис 3.7 Заміна порожньої множини

Програма також розрахована на часткові випадки відповідей, що не підпадають під загальну форму відповіді. Наприклад, система може не мати розв`язків через неможливу умову до деякої множини. На рисунку 3.8 наведено фрагмент сеансу роботи при заданому рівнянні $(A \cap \bar{A}) \cap X = \mho$, що не має розв`язків.

```
(1)+X*(0)+X^*(1)=0
Answer to equation number1:
If (1) = 0, than
(1) < X < (0)^.
Else no solutions.

The system does not have solutions
```

Рис 3.8 Відповідь до системи, що не має розв`язків

ВИСНОВКИ

Під час розробки даного програмного продукту вдалося реалізувати універсальний спосіб розв`язання системи рівнянь з одним невідомим в алгебрі множин. Для зручності користувача був створений дружній та інтуїтивно зрозумілий інтерфейс. Поставлене завдання можна вважати виконаним.

За допомогою мови програмування C++ вдалося реалізувати математичну модель, яка використовується у процесі роботи програми. Інтерфейс був реалізований у алфавітно-цифровому режимі консольного вікна із використанням бібліотеки «windows.h». Основним середовищем розробки був Dev C++ Version 5.11. Для оформлення документації до розробленого програмного продукту були використані графічні редактори Paint та Gimp 2.10.22, а також текстовий редактор Microsoft Word 2019.

Перевагами розробленого проекту над іншими програмами з аналогічними функціями є, по-перше, виведення проміжних результатів розв`язку рівняння, а по-друге, наявність детальних пояснень кожної дії: при спрощенні або заміні будь-якого елементу рівняння виводиться назва відповідного закону, означення, або тотожності алгебри множин та його формулювання. Це все сприяє засвоєнню теорії та практики з теми «теорія множин», що є важливою складовою частиною у вивченні дискретної математики.

Основним недоліком даної програми автор вважає вигляд відповіді до заданої системи рівнянь. Багато отриманих відповідей можна спростити, застосувавши закон абсорбції, реалізація якого є складнішою відносно реалізації інших законів алгебри множин через велику кількість способів розташування трьох множин та доповнень до них (у всіх інших законах взаємодіють лише 2 множини). Також під час розв`язання рівняння в алгебрі множин людиною для миттєвого спрощення часто застосовується «здоровий

глузд». Його використання програмою неможливе, оскільки людина бачить та аналізує одразу весь вираз, а програма – лише його схожість із наданими шаблонами.

Отже, для розробленого продукту можливе подальше покращення реалізацією закону абсорбції та типових для алгебри множин спрощень. На думку автора, друга частина можлива, наприклад, із використанням нейронної мережі. Також має місце ідея ілюстрації відповіді за допомогою кругів Ейлера на графічному елементі інтерфейсу Windows Forms.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Страуструп, Бьерн. Язык программирования C++, специальное издание: Пер. С англ. – С. Анисимова и М. Кононова. М.: «Издательство БИНОМ», 2002 г. – 1099 с.
2. Методичні вказівки до виконання розрахункових робіт з дисципліни дискретна математика для студентів спеціальностей: 124 «системний аналіз», 122 «комп`ютерні науки» / Уклад.: І.Я. Спекторський, О.В. Стусь, В.М. Статкевич. – К.: НТУУ «КПІ», ІПСА, 2017. – 84 с.
3. Колмогоров А.Н., Фомин С.В. Элементы теории функций и функционального анализа. — изд. четвёртое, переработанное. — М.: Наука, 1976. — 544 с.
4. Програмування та алгоритмічні мови. Програмування: Методичні вказівки до виконання курсового проекту для студентів галузі знань 12 «Інформаційні технології» спеціальність 122 «Комп’ютерні науки та інформаційні технології», 124 «Системний аналіз» / Уклад.: І.В.Назарчук, Г.Г.Швачко – К.НТУУ «КПІ», 2017 – 34с.
5. https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2
6. http://www.c-cpp.ru/content/strcat-fstrcat

Додаток А Текст програми

```
#include <iostream>
#include <math.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <windows.h>
#include <fstream>

#define ATTR1 96
#define ATTR2 97
#define KEY_ARROW_RIGHT 77
#define KEY_ARROW_LEFT 75
#define KEY_ENTER 13
#define KEY_ESC 27

using namespace std;
char *m[] = {"Add equation", "Solve system", "Clear", "Help", "Exit"};
bool Globalvx = false, Globalvxd = false, flag = false;
HANDLE hstdout;
int button, act = 0, nEquation = 0, thisEmptyMn = 0;
const int level = 29524, level2 = 100, menu = 5;
char x[level][level2], equations[5][level2], answers[5][3][50], globalAns[3][50],
temp2[50], NulMn, EmptyMn[10];

void draw(HANDLE h, COORD c, int k, int n, int len)
{
	SetConsoleTextAttribute(h, ATTR1);
	//system("cls");
	COORD cc = c;
	COORD c2 = {0, 1};
	for (int i = 0;i < n;i++)
	{
		cc.X=c.X + i*len;
		SetConsoleCursorPosition(h, cc);
		std::cout<<m[i];
	}
	cc.X = c.X + k*len;
	SetConsoleCursorPosition(h, c2);
	cout<<"Use left and right arrows to switch menu paragraph\nYour
system:"<<'\n';
	for(int i = 0; i < nEquation; i++)
```

```
            std::cout<<equations[i]<<'\n';
        SetConsoleTextAttribute(h, ATTR2);
        SetConsoleCursorPosition(h, cc);
        std::cout<<m[k];
}

void getparts(int xx)
{
        int nskobok = 0;
        char *p, *end1, *end2, mn1[level2], mn2[level2], op;
        end1 = x[xx];
        p = x[xx] + 1;
        if(*end1 != '(')
        {
                mn1[0] = *end1;
                mn1[1] = '\0';
        }
        else
        {
                do
                {
                        if(*end1 == '(')      nskobok++;
                        if(*end1 == ')')      nskobok--;
                        end1++;
                }
                while (nskobok);
                end1--;
                for (int i = 0; p != end1; p++, i++)
                {
                        mn1[i] = *p;
                        mn1[i + 1] = '\0';
                }
        }
        op = *(end1 + 1);
        end2 = end1 + 2;
        p = end2 + 1;
        if(*end2 != '(')
        {
                mn2[0] = *end2;
                mn2[1] = '\0';
                end2+=2;
        }
        else
        {
```

```
                do
                {
                        if(*end2 == '(')      nskobok++;
                        if(*end2 == ')')      nskobok--;
                        end2++;
                }
                while (nskobok);
                end2--;
                for (int i = 0; p != end2; p++, i++)
                {
                        mn2[i] = *p;
                        mn2[i + 1] = '\0';
                }
        }
        strcpy(x[3*xx+1], mn1);
        x[3*xx+2][0] = op;
        strcpy(x[3*xx+3], mn2);
}

void fill(int h)
{
        for(int i = 1; i < level; i++)
                for(int j = 0; j < level2; j++)
                        x[i][j] = '\0';
        for(int i = 0; i <= level; i++)
        {
                if(strlen(x[i]) > 1)
                        getparts(i);
        }
}

void connect(int i)
{
        while(i != 0)
        {
                if(i%3 == 0) i-=2;
                for(int k = 0; k <= strlen(x[i/3]); k++)
                        x[i/3][k] = '\0';
                if(strlen(x[i]) > 1) strcat(x[i/3], "(");
                strcat(x[i/3], x[i]);
                if(strlen(x[i]) > 1) strcat(x[i/3], ")");
                strcat(x[i/3], x[i+1]);
                if(strlen(x[i+2]) > 1) strcat(x[i/3], "(");
                strcat(x[i/3], x[i+2]);
```

```
                if(strlen(x[i+2]) > 1) strcat(x[i/3], ")");
                i=i/3;
        }
}

void unite(int j)
{
        for(int t = 0; x[0][t] != '\0'; t++)
                x[0][t] = '\0';
        if(strlen(answers[j][0]) > 2) strcat(x[0], answers[j][0]);
        if(strlen(answers[j][1]) > 5)
                strcat(x[0], answers[j][1]);
        else
                if(Globalvx == true)
                        if(strlen(answers[j][0]) > 2) strcat(x[0], "+X"); else strcat(x[0],
"X");
                if(strlen(answers[j][2]) > 6)
                        strcat(x[0], answers[j][2]);
                else
                        if(Globalvxd == true)
                                if((strlen(answers[j][0]) > 2)||(strlen(answers[j][2]) > 6))
strcat(x[0], "+X^"); else strcat(x[0], "X^");
        std::cout<<x[0]<<"=0\n";
}

void zakony(int j)
{
        for(int y = 0; y < 3; y++)
        {
                do
                {
                        flag = false;
                        if(strstr(answers[j][y], "0^"))
                        {
                                cout<<"Using identity 0^ = 1, we replace it\n";
                                char *p = strstr(answers[j][y], "0^"), *p2;
                                p2 = p+2;
                                *p = '1';
                                p++;
                                while(*p2)
                                {
                                        *p = *p2;
                                        p++;
                                        p2++;
```

```
                        }
                        *p = '\0';
                        p++;
                        *p = '\0';
                        flag = true;
                        unite(j);
                }
                if(strstr(answers[j][y], "1^"))
                {
                        cout<<"Using identity 1^ = 0, we replace it\n";
                        char *p = strstr(answers[j][y], "1^"), *p2;
                        p2 = p+2;
                        *p = '0';
                        p++;
                        while(*p2)
                        {
                                *p = *p2;
                                p++;
                                p2++;
                        }
                        *p = '\0';
                        p++;
                        *p = '\0';
                        flag = true;
                        unite(j);
                }
        } while(flag);
    }
    for(int y = 0; y < 3; y++)
    {
        do
        {
                flag = false;
                //universalni mezi
                if(strstr(answers[j][y], "0*"))
                {
                        cout<<"Using law about universal boundaries 0 * (A) = 0,
we replace it\n";
                        char *p = strstr(answers[j][y], "0*")+1, *p2;
                        p2 = p+2;
                        if(*p2 == '^') p2++;
                        while(*p2)
                        {
                                *p = *p2;
```

```
                        p++;
                        p2++;
                }
                *p = '\0';
                p++;
                *p = '\0';
                flag = true;
                unite(j);
        }
        if(strstr(answers[j][y], "*0"))
        {
                cout<<"Using universal boundaries (A) * 0 = 0, we replace
it\n";
                char *p = strstr(answers[j][y], "*0")-1, *p2;
                p2 = p+3;
                if(*p == '^') p--;
                *p = '0';
                p++;
                while(*p2)
                {
                        *p = *p2;
                        p++;
                        p2++;
                }
                *p = '\0';
                p++;
                *p = '\0';
                flag = true;
                unite(j);
        }
        if((strstr(answers[j][y], "+1+")) || (strstr(answers[j][y], "(1+")) ||
(strstr(answers[j][y], "+1)")))
        {
                cout<<"Using universal boundaries (A) + 1 = 1, we replace
it\n";
                switch(y)
                {
                        case 0: strcpy(answers[j][0], "(1)"); break;
                        case 1: strcpy(answers[j][1], "X*(1)"); break;
                        case 2: strcpy(answers[j][2], "X^*(1)"); break;
                }
                flag = true;
                unite(j);
        }
```

```
//neitralnist
if(strstr(answers[j][y], "0+"))
{
	cout<<"Using neutrality 0 + (A) = (A), we replace it\n";
	char *p = strstr(answers[j][y], "0+"), *p2;
	p2 = p+2;
	while(*p2)
	{
		*p = *p2;
		p++;
		p2++;
	}
	*p = '\0';
	p++;
	*p = '\0';
	flag = true;
	unite(j);
}
if(strstr(answers[j][y], "+0"))
{
	cout<<"Using neutrality (A) + 0 = (A), we replace it\n";
	char *p = strstr(answers[j][y], "+0"), *p2;
	p2 = p+2;
	while(*p2)
	{
		*p = *p2;
		p++;
		p2++;
	}
	*p = '\0';
	p++;
	*p = '\0';
	flag = true;
	unite(j);
}
if(strstr(answers[j][y], "1*"))
{
	cout<<"Using neutrality 1 * (A) = (A), we replace it\n";
	char *p = strstr(answers[j][y], "1*"), *p2;
	p2 = p+2;
	while(*p2)
	{
		*p = *p2;
		p++;
```

```
                        p2++;
                    }
                    *p = '\0';
                    p++;
                    *p = '\0';
                    flag = true;
                    unite(j);
                }
                if(strstr(answers[j][y], "*1"))
                {
                    cout<<"Using neutrality (A) * 1 = (A), we replace it\n";
                    char *p = strstr(answers[j][y], "*1"), *p2;
                    p2 = p+2;
                    while(*p2)
                    {
                        *p = *p2;
                        p++;
                        p2++;
                    }
                    *p = '\0';
                    p++;
                    *p = '\0';
                    flag = true;
                    unite(j);
                }
                //idempotentnist
                for(char *ch = answers[j][y]; *ch; ch++)
                {
                    if((*ch == '*') && (((isalpha(*(ch-1))) && (isalpha(*(ch+1))) && ((*(ch+2)) != '^') && (*(ch+1)==*(ch-1))) || ((isalpha(*(ch-2))) && (isalpha(*(ch+1))) && ((*(ch-1)) == '^') && ((*(ch+2)) == '^') &&(*(ch-2)==*(ch+1)))))
                    {
                        cout<<"Using idempotence (A) * (A) = (A), we replace it\n";
                        char *p = ch, *p2;
                        p2 = p+2;
                        if(*p2 == '^') p2++;
                        while(*p2)
                        {
                            *p = *p2;
                            p++;
                            p2++;
                        }
```

```
                        *p = '\0';
                        p++;
                        *p = '\0';
                        flag = true;
                        unite(j);
                    }
                    if(((*ch == '+') && (*(ch-3) != '*') && (*(ch-2) != '*') &&
(*(ch+3) != '*') && (*(ch+2) != '*') && (isalpha(*(ch-1))) && (isalpha(*(ch+1)))
&& ((*(ch+2)) != '^') && (*(ch+1)==*(ch-1))) || ((isalpha(*(ch-2))) &&
(isalpha(*(ch+1))) && ((*(ch-1)) == '^') && ((*(ch+2)) == '^') &&(*(ch-
2)==*(ch+1))))
                    {
                        cout<<"Using idempotence (A) + (A) = (A), we
replace it\n";
                        char *p = ch, *p2;
                        p2 = p+2;
                        if(*p2 == '^') p2++;
                        while(*p2)
                        {
                            *p = *p2;
                            p++;
                            p2++;
                        }
                        *p = '\0';
                        p++;
                        *p = '\0';
                        flag = true;
                        unite(j);
                    }
                }
                //dopovnenist
                for(char *q = answers[j][y]; *q; q++)
                {
                    if((*q == '^') && (((*(q - 1) == *(q - 3)) && (isalpha(*(q
- 1))) && (*(q - 2) == '*')) || ((*(q - 1) == *(q + 2)) && (isalpha(*(q - 1))) && (*(q
+ 1) == '*'))))
                    {
                        cout<<"Using complementarity (A) * (A)^ = 0, we
replace it\n";
                        char *p = q - 3;
                        *p = '0';
                        p++;
                        q++;
                        for(; *q; q++, p++)
```

```
                                        *p = *q;
                                    *p = '\0';
                                    *(p+1) = '\0';
                                    *(p+2) = '\0';
                                    *(p+3) = '\0';
                                    *(p+4) = '\0';
                                    unite(j);
                                    flag = true;
                                }
                            }
                        } while(flag);
                    }
                }

bool IsCorrect(char *q)
{
        const char operations[7] = "+*^-&=", symbols[11] = "+*^-&=()10";
        int n = 0, n0 = 0, nRavno;
        char s[2], s1[2], s0[2];
        if(strstr(operations, &*q)) return false;
        for(char *p = q; *p; p++)
        {
                s[0] = *p;
                s1[0] = *(p + 1);
                s0[0] = *(p - 1);
                if(*p == '(')
                {
                        if(strstr(operations, s1)) return false;
                        n++;
                }
                if(*p == ')')
                {
                        if(strstr(operations, s0)) return false;
                        n--;
                }
                if(n == 0) n0++;
                if(s[0] == '=')
                {
                        nRavno++;
                        n0 = 0;
                        if(n) return false;
                }
                if((strstr(operations, s)) && (strstr(operations, s1)))
                if((nRavno == 2) || (n0 == 3) || (n < 0))
```

```
                return false;
            if(((s[0] < 'A') || (s[0] > 'Z')) && (strstr(symbols, s) == NULL)) return
false;
        }
        return true;
}

int main(int argc, char** argv)
{
        system("title Solving system of equations");
        int l, kol = 0;
        const int len = 16;
        char *p, q, temp[level2], file[200];
        ifstream my1("help.txt", ios_base::in);
        hstdout = GetStdHandle(STD_OUTPUT_HANDLE);
        COORD pos = {0, 0}, my_pos = {0, 0};
        nEquation = 0;
        label1:
        SetConsoleTextAttribute(hstdout, ATTR1);
        system("cls");
        draw(hstdout, pos, act, menu, len);
        while(true)
        {
                if(kbhit())
                {
                        SetConsoleTextAttribute(hstdout, ATTR1);
                        while(act <= 0)
                        {
                                act = act+menu;
                        }
                        button = getch();
                        if(button == 75)
                        {
                                act = (act - 1)%menu;
                                draw(hstdout, pos, act, menu, len);
                        }

                        if(button == 77)
                        {
                                act = (act + 1)%menu;
                                draw(hstdout, pos, act, menu, len);
                        }

                        if(button == 13)
```

```
{
	act = act % menu;
	switch(act)
	{
		case 0:
			system("cls");
			cout<<"Legend:\n+ is union;\n* is intersection;\n^ is an addition;\n- is the difference;\n& is a symmetrical difference;\n0 is Empty set;\n1 is Universal set;\nX is Unknown set.\nEnter your equation:\n";
			fflush(stdin);
			cin>>equations[nEquation];
			if (IsCorrect(equations[nEquation]))
				nEquation++;
			else
			{
				for(char *p = equations[nEquation]; *p; p++)
					*p = '\0';
				cout<<"The equation was entered incorrectly.\nPress Enter to return to main menu";
				while(true)
				{
					if(kbhit())
					{
						button = getch();
						if(button == 13)
							goto label1;
					}
				}
			}
			goto label1;
			break;
		case 1:
			my_pos.Y = nEquation + 3;
			SetConsoleCursorPosition(hstdout, my_pos);
			for(int y = 0; y < 3; y++)
				for(int u = 0; u < 50; u++)
					globalAns[y][u] = '\0';
			for(int y = 0; y < 10; y++)
				EmptyMn[y] = '\0';
			for(int j = 0; j < nEquation; j++)
			{
				for(int w = 0; w < level; w++)
```

```
                for(int y = 0; y < level2; y++)
                        x[w][y] = '\0';
        strcpy(x[0], equations[j]);
        for(int y = 0; y < 3; y++)
                for(int u = 0; u < 50; u++)
                        answers[j][y][u] = '\0';
        answers[j][0][0] = '(';
        answers[j][1][0] = '+';
        answers[j][1][1] = 'X';
        answers[j][1][2] = '*';
        answers[j][1][3] = '(';
        answers[j][2][0] = '+';
        answers[j][2][1] = 'X';
        answers[j][2][2] = '^';
        answers[j][2][3] = '*';
        answers[j][2][4] = '(';
        cout<<"Equation number "<<j+1<<":\n";
        cout<<x[0]<<"\n";
        cout<<"Using the equivalence (A(X) = B(X)) <=> (A(X) & B(X) = 0), we transform the equation to equation, that has empty set in its right part\n";
        p = &x[0][0];
        temp[0] = '(';
        for(l = 1; *p != '='; p++, l++)
                temp[l] = *p;
        *(temp + l) = ')';
        l++;
        *(temp + l) = '&';
        l++;
        *(temp + l) = '(';
        l++;
                p++;
        for(; *p != '\0'; p++, l++)
                temp[l] = *p;
        *(temp + l) = ')';
        *(temp + l + 1) = '\0';
        l = 1;
        strcpy(x[0], temp);
        cout<<x[0]<<"=0\n";
        fill(j);
        cout<<"Using the definition of symmetric difference (A & B) = (A - B) + (B - A), we replace it\n";
        for(long int i = 2; i < level - 3; i+=3)
```

```
{
	if(x[i][0] == '&') //ubiraem &
	{
		for(int k = 0; k <= strlen(x[i/3]); k++)
			x[i/3][k] = '\0';
		strcat(x[i/3], "(");
		if(strlen(x[i-1]) > 1)
strcat(x[i/3], "(");
		strcat(x[i/3], x[i-1]);
		if(strlen(x[i-1]) > 1)
strcat(x[i/3], ")");
		strcat(x[i/3], "-");
		if(strlen(x[i+1]) > 1)
strcat(x[i/3], "(");
		strcat(x[i/3], x[i+1]);
		if(strlen(x[i+1]) > 1)
strcat(x[i/3], ")");
		strcat(x[i/3], ")+(");
		if(strlen(x[i+1]) > 1)
strcat(x[i/3], "(");
		strcat(x[i/3], x[i+1]);
		if(strlen(x[i+1]) > 1)
strcat(x[i/3], ")");
		strcat(x[i/3], "-");
		if(strlen(x[i-1]) > 1)
strcat(x[i/3], "(");
		strcat(x[i/3], x[i-1]);
		if(strlen(x[i-1]) > 1)
strcat(x[i/3], ")");
		strcat(x[i/3], ")");
		i=i/3;
		connect(i);
		i = 2;
		std::cout<<x[0]<<"=0\n";
		fill(j);
	}
}
cout<<"Using the definition of difference (A - B) = (A * (B^)), we replace it\n";
for(long int i = 2; i < level - 3; i+=3)
{
	if(x[i][0] == '-') //ubiraem -
	{
```

```
                                        for(int k = 0; k <= strlen(x[i/3]); k++)
                                            x[i/3][k] = '\0';
                                        if(strlen(x[i-1]) > 1) strcat(x[i/3], "(");
                                        strcat(x[i/3], x[i-1]);
                                        if(strlen(x[i-1]) > 1) strcat(x[i/3], ")");
                                        strcat(x[i/3], "*");
                                        strcat(x[i/3], "(");
                                        if(strlen(x[i+1]) > 1) strcat(x[i/3], "(");
                                        strcat(x[i/3], x[i+1]);
                                        if(strlen(x[i+1]) > 1) strcat(x[i/3], ")");
                                        strcat(x[i/3], "^)");
                                        i=i/3;
                                        connect(i);
                                        i = 2;
                                        std::cout<<x[0]<<"=0\n";
                                        fill(j);
                                    }
                                }
                                cout<<"Using de Morgan's law (A + B)^ = ((A^) * (B^)); (A * B)^ = ((A^) + (B^)) and involutiveness (A^)^ = A, we replace external additions\n";
                                for(long int i = 2; i < level - 3; i+=3)
                                {
                                    if((x[i][0] == '^') && (strlen(x[i-1]) > 1)) //ubiraem ()^
                                    {
                                        i--;
                                        if(x[3*i+2][0] == '^')
                                        {
                                            strcpy(x[i], x[3*i + 1]);
                                        }
                                        else
                                        {
                                            if(x[3*i+2][0] == '+') x[3*i+2][0] = '*';
                                            else x[3*i+2][0] = '+';
                                            x[i][0] = '\0';
```

```
                                        strcat(x[i], "(");
                                        if(strlen(x[3*i+1]) >
1) strcat(x[i], "(";
                                        strcat(x[i],
x[3*i+1]);
                                        if(strlen(x[3*i+1]) >
1) strcat(x[i], ")");
                                        strcat(x[i], "^");
                                        strcat(x[i], ")");
                                        strcat(x[i],
x[3*i+2]);
                                        strcat(x[i], "(");
                                        if(strlen(x[3*i+3]) >
1) strcat(x[i], "(";
                                        strcat(x[i],
x[3*i+3]);
                                        if(strlen(x[3*i+3]) >
1) strcat(x[i], ")");
                                        strcat(x[i], "^");
                                        strcat(x[i], ")");
                                    }
                                    strcpy(x[i/3], x[i]);
                                    i = i/3;
                                    connect(i);
                                    i = 2;
                                    std::cout<<x[0]<<"=0\n";
                                    fill(j);
                                }
                            }
                            cout<<"using distributive law (A * (B
+ C)) = ((A * B) + (A * C)), we replace external intersections\n";
                            for(int i = 2; i <= (level-3)/3; i+=3)
//ubiraem vneshnie peresecheniya
                            {
                                if((x[i][0] == '*')&&(x[3*(i-
1)+2][0] == '+'))
                                {
                                    for(int k = 0; k < level2;
k++)
                                        x[i/3][k] = '\0';
                                    strcat(x[i/3], "(");
                                    if(strlen(x[i+1]) > 1)
strcat(x[i/3], "(";
                                    strcat(x[i/3], x[i+1]);
```

```
                                        if(strlen(x[i+1])    >    1)
strcat(x[i/3], ")");
                                        strcat(x[i/3], "*");
                                        if(strlen(x[3*i-2])   >   1)
strcat(x[i/3], "(");
                                        strcat(x[i/3], x[3*i-2]);
                                        if(strlen(x[3*i-2])   >   1)
strcat(x[i/3], ")");
                                        strcat(x[i/3], ")");
                                        strcat(x[i/3], "+");
                                        strcat(x[i/3], "(");
                                        if(strlen(x[i+1])    >    1)
strcat(x[i/3], "(");
                                        strcat(x[i/3], x[i+1]);
                                        if(strlen(x[i+1])    >    1)
strcat(x[i/3], ")");
                                        strcat(x[i/3], "*");
                                        if(strlen(x[3*i])    >    1)
strcat(x[i/3], "(");
                                        strcat(x[i/3], x[3*i]);
                                        if(strlen(x[3*i])    >    1)
strcat(x[i/3], ")");
                                        strcat(x[i/3], ")");
                                        i=i/3;
                                        connect(i);
                                        i = -1;
                                        std::cout<<x[0]<<"=0\n";
                                        fill(j);
                                        continue;
                                }
                                if((x[i][0]                          ==
'*')&&(x[3*(i+1)+2][0] == '+'))
                                {
                                        for(int k = 0; k < level2;
k++)
                                                x[i/3][k] = '\0';
                                        strcat(x[i/3], "(");
                                        if(strlen(x[i-1])    >    1)
strcat(x[i/3], "(");
                                        strcat(x[i/3], x[i-1]);
                                        if(strlen(x[i-1])    >    1)
strcat(x[i/3], ")");
                                        strcat(x[i/3], "*");
```

```
                        if(strlen(x[3*i+4]) > 1)
strcat(x[i/3], "(");
                        strcat(x[i/3], x[3*i+4]);
                        if(strlen(x[3*i+4]) > 1)
strcat(x[i/3], ")");
                        strcat(x[i/3], ")");
                        strcat(x[i/3], "+");
                        strcat(x[i/3], "(");
                        if(strlen(x[i-1]) > 1)
strcat(x[i/3], "(");
                        strcat(x[i/3], x[i-1]);
                        if(strlen(x[i-1]) > 1)
strcat(x[i/3], ")");
                        strcat(x[i/3], "*");
                        if(strlen(x[3*i + 6]) > 1)
strcat(x[i/3], "(");
                        strcat(x[i/3], x[3*i + 6]);
                        if(strlen(x[3*i + 6]) > 1)
strcat(x[i/3], ")");
                        strcat(x[i/3], ")");
                        i=i/3;
                        connect(i);
                        i = -1;
                        std::cout<<x[0]<<"=0\n";
                        fill(j);
                        continue;
                }
        }
cout<<"Removing unnecessary
brackets\n";

for(int t = 0; t < level2; t++)
        x[1][t] = '\0';
x[1][0] = '(';
for(int t = 1, y = 0; x[0][y] != '\0'; y++)
{
        if(x[0][y] != '+')
        {
                if((x[0][y] != '(') &&
(x[0][y] != ')'))
                {
                        x[1][t] = x[0][y];
                        t++;
                }
        }
```

```
                else
                {
                        x[1][t] = ')';
                        t++;
                        x[1][t] = '+';
                        t++;
                        x[1][t] = '(';
                        t++;
                }
        }
        x[1][strlen(x[1])] = ')';
        strcpy(x[0], x[1]);
        std::cout<<x[0]<<"=0\n";
        cout<<"Grouping equation to form (A1) + X * (A2) + X^ * (A3)\n";
        while(strstr(x[0], "X*X"))
        {
                int u = 0, u2 = 0;
                while(x[0][u])
                {
                        if((x[0][u] == 'X') && (x[0][u+1] == '*') && (x[0][u+2] == 'X'))
                                u+=2;
                        x[0][u2] = x[0][u];
                        u++;
                        u2++;
                }
                x[0][u2] = '\0';
                x[0][u2 + 1] = '\0';
                x[0][u2 + 2] = '\0';
        }
        while(strstr(x[0], "X^*X^"))
        {
                int u = 0, u2 = 0;
                while(x[0][u])
                {
                        if((x[0][u] == 'X') && (x[0][u+1] == '^') && (x[0][u+2] == '*') && (x[0][u+3] == 'X') && (x[0][u+4] == '^'))
                                u+=3;
                        x[0][u2] = x[0][u];
                        u++;
                        u2++;
                }
```

```
                x[0][u2] = '\0';
                x[0][u2 + 1] = '\0';
                x[0][u2 + 2] = '\0';
                x[0][u2 + 3] = '\0';
        }
        std::cout<<x[0]<<"=0\n";
        x[2][0] = '\0';
        for(int t = 0; t < strlen(x[0]); t++)
        {
                int poch = t, temp = t;
                bool vx = false, vxd = false;
                while((x[0][t] != '+') && (x[0][t]
!= '\0'))
                {
                        if((x[0][t] == 'X') &&
(x[0][t+1] != '^'))
                                {vx = true;
                        if((x[0][t] == 'X') &&
(x[0][t+1] == '^'))
                                {vxd = true,
Globalvxd = true;}
                        t++;
                }
                if((vx == true) && (vxd == true))
                {
                        for(int cpy = t + 1;
x[0][cpy] != '\0'; cpy++)
                        {
                                x[0][poch] =
x[0][cpy];
                                poch++;
                        }
                        t = temp;
                        for(; poch < strlen(x[0]);
poch++)
                                x[0][poch] = '\0';
                        std::cout<<x[0]<<"=0\n";
                }
                if((vx == false) && (vxd ==
false))
                {
                        if(strlen(answers[j][0]) >
1)
```

```
        answers[j][0][strlen(answers[j][0])] = '+';
                                                        while(poch <= t)
                                                        {
                                                                if((x[0][poch] != '(')
&& (x[0][poch] != '+') && (x[0][poch] != ')'))

        x[2][strlen(x[2])] = x[0][poch];
                                                                poch++;
                                                        }
                                                        strcat(answers[j][0], x[2]);
                                                        strcpy(temp2, x[2]);
                                                        for(int u = 0; x[2][u] !=
'\0'; u++) x[2][u] = '\0';
                                                }
                                                if((vx == true) && (vxd ==
false))
                                                {
                                                        if(strlen(answers[j][1]) >
4)

        answers[j][1][strlen(answers[j][1])] = '+';
                                                        if(t - poch == 3)
                                                                x[2][0] = '1';
                                                        else while(poch <= t)
                                                        {
                                                                if((x[0][poch] != '(')
&& (x[0][poch] != '+') && (x[0][poch] != ')') && (x[0][poch] != 'X') && (x[0][poch
+ 1] != 'X') && ((x[0][poch - 1] != 'X') || (x[0][poch - 2] != '(')))

        x[2][strlen(x[2])] = x[0][poch];
                                                                poch++;
                                                        }
                                                        strcat(answers[j][1], x[2]);
                                                        strcpy(temp2, x[2]);
                                                        for(int u = 0; x[2][u] !=
'\0'; u++) x[2][u] = '\0';
                                                }
                                                if((vx == false) && (vxd ==
true))
                                                {
                                                        if(strlen(answers[j][2]) >
5)
```

```
answers[j][2][strlen(answers[j][2])] = '+';
                                                    if(t - poch == 4)
                                                        x[2][0] = '1';
                                                    else while(poch <= t)
                                                    {
                                                        if((x[0][poch] != '(') && (x[0][poch] != '+') && (x[0][poch] != ')') && ((x[0][poch + 1] != 'X') || (x[0][poch + 3] != ')')) && (x[0][poch - 2] != 'X') && (x[0][poch] != 'X') && (x[0][poch - 1] != 'X'))

        x[2][strlen(x[2])] = x[0][poch];
                                                            poch++;
                                                    }
                                                    strcat(answers[j][2], x[2]);
                                                    strcpy(temp2, x[2]);
                                                    for(int u = 0; x[2][u] != '\0'; u++) x[2][u] = '\0';
                                                }
                                            }
                                            answers[j][0][strlen(answers[j][0])] = ')';
                                            answers[j][1][strlen(answers[j][1])] = ')';
                                            answers[j][2][strlen(answers[j][2])] = ')';
                                            cout<<"Here we consider that the intersection is performed earlier than the union:\n";
                                            unite(j);
                                            zakony(j);
                                            unite(j);
                                            int jEmptyMn = thisEmptyMn;
                                            for(char *p = answers[j][0]; *p; p++)
                                            {
                                                if((isalpha(*p)) && (((*(p-1) == '(') || (*(p-1) == '+')) && ((*(p+1) == ')') || (*(p+1) == '+'))))
                                                {
                                                    NulMn = *p;
                                                    cout<<"In the bracket that does not contain X we see that "<<*p<<" is 0, so we can replace it\n";
                                                    if(!(strstr(EmptyMn, &NulMn))) EmptyMn[strlen(EmptyMn)] = NulMn;
                                                    for(int y = 0; y < 3; y++)
```

```
                                        for(char *p1 =
answers[j][y]; *p1; p1++)
                                            if(*p1 ==
NulMn) *p1 = '0';
                                    unite(j);
                                    cout<<"Also we can
replace it in solved equations:\n";
                                    for(int u = 0; u < j; u++)
                                    {
                                        cout<<"Equation
number "<<u+1<<"\n";
                                        for(int y = 0; y < 3;
y++)
                                            for(char *p1
= answers[u][y]; *p1; p1++)
                                                if(*p1
== NulMn) *p1 = '0';
                                        unite(u);
                                        zakony(u);
                                    }
                                    cout<<"Also we can
replace it in not solved equations:\n";
                                    for(int u = j+1; u <
nEquation; u++)
                                    {
                                        cout<<"Equation
number "<<u+1<<"\n";
                                        for(char *p1 =
equations[u]; *p1; p1++)
                                            {
                                                if(*p1
== NulMn) *p1 = '0';
                                            }
        cout<<equations[u]<<"\n";
                                    }
                                    cout<<"Returning to our
equation:\n";
                                    unite(j);
                                    zakony(j);
                                }
                            }
                            Globalvx = false; Globalvxd = false;
```

```
                                        cout<<"Answer to equation
number"<<j+1<<":\n";
                                        if(strlen(answers[j][0]) > 2) cout<<"If
"<<answers[j][0]<<" = 0, than\n";
                                        cout<<answers[j][2] + 4<<" < X <
"<<answers[j][1] + 3<<"^.\nElse no solutions.\n\n";
                                        if(strcmp(answers[j][0], "(0)") == 0)
for(int u = 0; u < 3; u++) answers[j][0][u] = '\0';
                                        if(strcmp(answers[j][2], "+X^*(0)") ==
0) for(int u = 0; u < 7; u++) answers[j][2][u] = '\0';
                                        if(strcmp(answers[j][1], "+X*(0)") ==
0) for(int u = 0; u < 6; u++) answers[j][1][u] = '\0';
                                        for(int u = 0; u < 3; u++)
                                        {
                                                if((strlen(globalAns[u]) > 0) &&
(strlen(answers[j][u]) > 0))
                                                        strcat(globalAns[u], "+");
                                                int inc;
                                                if(u == 0) inc = 0;
                                                if(u == 1) inc = 3;
                                                if(u == 2) inc = 4;
                                                strcat(globalAns[u],
answers[j][u] + inc);

                                        }
                                }
                                if (((strcmp(globalAns[2], "0")) &&
!(strcmp(globalAns[1], "1"))) || (!strcmp(globalAns[0], "(1)")))
                                        cout<<"The system does not have
solutions\n";
                                else
                                {
                                        if(!(strlen(globalAns[0])))
globalAns[0][0] = '0';
                                        if(!(strlen(globalAns[1])))
globalAns[1][0] = '0';
                                        if(!(strlen(globalAns[2])))
globalAns[2][0] = '0';
                                        cout<<"Main solution:\nIf
"<<globalAns[0]<<" = 0";
                                        for(char *p = EmptyMn; *p; p++)
cout<<" and "<<*p<<" = 0";
                                        cout<<", than\n"<<globalAns[2]<<" <
X < ("<<globalAns[1]<<")^.\nElse no solutions.";
```

```
                                        }
                                        break;
                                case 4:
                                        exit(0);
                                        break;
                                case 2:
                                        for(int i = 1; i < 5; i++)
                                                equations[i][0] = '\0';
                                                x[0][0] = '\0';
                                        nEquation = 0;
                                        draw(hstdout, pos, act, menu, len);
                                        goto label1;
                                        break;
                                case 3:
                                        system("cls");
                                        while(!my1.eof())
                                        {
                                                my1.getline(file, 200);
                                                cout<<file<<"\n";
                                        }
                                        cout<<"\nPress Enter to return to main
menu";
                                        while(true)
                                                if(kbhit())
                                                {
                                                        my1.close();
                                                        my1.open("help.txt",
ios_base::in);
                                                        button = getch();
                                                        if(button == 13)
                                                                goto label1;
                                                }
                                        break;
                                }
                        }
                }
        }
        return 0;
}
```