ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ДЕННОЇ ОСВІТИ

ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА КОМП‘ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Допускається до захисту

Завідувач кафедри_____О.ОЛЬХОВСЬКА
(підпис)

«_____»_____2021 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО ДИПЛОМНОЇ РОБОТИ

на тему

“ОПТИМІЗАЦІЯ ДОБОВИХ ОБСЯГІВ ВИРОБНИЦТВА ДЕТАЛЕЙ: ПРОГРАМНА РЕАЛІЗАЦІЯ ТРЕНАЖЕРА (МОДЕЛЮВАННЯ ТА РОЗВ’ЯЗУВАННЯ) ДИСТАНЦІЙНОГО КУРСУ «ПРОЕКТНЕ НАВЧАННЯ З КУРСУ «МЕТОДИ ОПТИМІЗАЦІЇ ТА ДОСЛІДЖЕННЯ ОПЕРАЦІЙ»”

зі спеціальності 122 «Комп’ютерні науки»
освітня програма «Комп’ютерні науки»
ступеня магістра

Виконавець роботи Гальчун Андрій Миколайович _____ «___»_____2021р.
(підпис)
Науковий керівник докт., фіз.-мат. наук, Колєчкіна Людмила Миколаївна
_____ «___»_____2021р.
(підпис)

ПОЛТАВА 2021 р.

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

ЗАТВЕРДЖУЮ

Завідувач кафедри_____О.ОЛЬХОВСЬКА
(підпис)

«03» вересня 2021р.

ЗАВДАННЯ ТА КАЛЕНДАРНИЙ ГРАФІК
ВИКОНАННЯ ДИПЛОМНОЇ РОБОТИ

Здобувач вищої освіти зі спеціальності 122 «Комп'ютерні науки»
Освітня програма «Комп'ютерні науки»
Прізвище, ім'я, по батькові Гальчун Андрій Миколайович
1. **Тема «Оптимізація добових обсягів виробництва деталей: програмна реалізація тренажера(моделювання та розв'язування)дистанційного курсу «Проектне навчання з курсу «Методи оптимізації та дослідження операцій» затверджена наказом ректора № ___ від «___» _____ 2021 р.**
Термін подання студентом дипломної роботи «__»_____ 2021 р.

2.Вихідні дані до дипломної роботи курс лекцій з «Методів оптимізації та дослідження операцій» ,інформаційні джерела.

3. Зміст пояснювальної записки (перелік питань, які потрібно розробити) ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ,СИМВОЛІВ,ОДИНИЦЬ,СКОРОЧЕНЬ,ТЕРМІНІВ

ВСТУП
1 ПОСТАНОВКА ЗАДАЧІ
- 1.1 Змістова постановка задачі

2 ІНФОРМАЦІЙНИЙ ОГЛЯД
- 2.1 Огляд робіт
- 2.2 Позитивні аспекти оглянутих робіт
- 2.3 Вади оглянутих робіт

3 ТЕОРЕТИЧНА ЧАСТИНА
- 3.1 Алгоритм тренажера
- 3.2 Блок-схеми

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ
- 4.1 Опис процесу програмної реалізації
- 4.2 Опис програми

ВИСНОВКИ

4. Перелік графічного матеріалу (з точним визначенням кількості блок-схем, іншого графічного матеріалу) 2 аркуші блок-схем, ілюстрації за необхідністю

5. Консультанти розділів дипломної роботи

Розділ	Прізвище, ініціали, посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1.Вступ	Колєчкіна Л.М.		
2.Постановка задачі	Колєчкіна Л.М.		
3.Інформаційний огляд	Колєчкіна Л.М.		
4.Теоретична частина	Колєчкіна Л.М.		
5.Практична реалізація	Колєчкіна Л.М.		

6. Календарний графік виконання дипломної роботи

Зміст роботи	Термін виконання	Фактичне виконання
Вступ		
Вивчення методичних рекомендацій та стандартів та звіт керівнику		
Постановка задачі		
Інформаційний огляд джерел бібліотек та інтернету		
Теоретична частина		
Практична частина		
Закінчення оформлення		
Доповідь студента на кафедрі		
Доробка (за необхідністю), рецензування		

Дата видачі завдання «03» вересня 2021 р.

Здобувач вищої освіти _____
(підпис)

Науковий керівник _____ _____
(підпис) (науковий ступінь, вчене звання, ініціали та прізвище)

Результати захисту дипломної роботи

Дипломна робота оцінена на _____
(балів, оцінка за національною шкалою, оцінка за ECTS)

Протокол засідання ЕК № ____ від «____» _____ 2021 р.

Секретар ЕК_____ _____
(підпис) (ініціали та прізвище)

Затверджую
Зав. кафедрою _____
к.ф.-м.н. О. ОЛЬХОВСЬКА

«____» _____ 2021 р.

Погоджено
Науковий керівник _____
докт.,ф.-м.н. Л.КОЛЄЧКІНА

«____» _____ 2021 р.

План

Дипломної роботи здобувача вищої освіти ступеня магістра
спеціальності 122 Комп'ютерні науки
освітня програма 122 Комп'ютерні науки
Прізвище, ім'я, по батькові: Гальун Андрій Миколайович
На тему: «Оптимізація добових обсягів виробництва деталей: програмна реалізація тренажера(моделювання та розв'язування)дистанційного курсу «Проектне навчання з курсу «Методи оптимізації та дослідження операцій»

Вступ
1 Постановка Задачі
    1.2 Змістова постановка задачі
2 Інформаційний огляд
    2.1 Огляд робіт
    2.2 Позитивні аспекти оглянутих робіт
    2.3 Вади оглянутих робіт
3 Теоретична частина
    3.1 Алгоритм тренажера
    3.2 Блок-Схеми
4 Практична реалізація
    4.1 Опис процесу програмної реалізації
    4.2 Опис програми
Висновки

Здобувач вищої освіти _____ А.ГАЛЬЧУН
«____» _____ 2021 р.

РЕФЕРАТ

Записка: 50 сторінки, 10 рисунків, 2 додатки (на 20 сторінках), 13 літературних джерела.

Предмет розробки – програмування елементів навчального тренажера.

Мета роботи – алгоритмізація та за програмування елементів навчального тренажера з побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей.

Методи, які були використані для розв'язування задачі – метод з побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей.

Розроблено алгоритм навчального тренажера для побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей.

Здійснена програмна реалізація побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей.

Ключові слова: ПОБУДОВА, МАТЕМАТИЧНА МОДЕЛЬ, ТРЕНАЖЕР, ЗАДАЧА, ОПТИМІЗАЦІЯ , ВИРОБНИЦТВО ДЕТАЛЕЙ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, символів, скорочень
$A_1, \dots, A_m$	Умовні види сировини
Д1, Д2	Деталь 1 та 2 відносно
a_{ij}	Коефіцієнт витрат
b_i	Права частина обмеження
ЦФ	Цільова функція
x_{ij}	Добовий обсяг виробництва деталей

ВСТУП

Актуальність. Навчитись побудові математичної моделі задачі оптимізації добових обсягів виробництва деталей.

Мета дипломної роботи. Алгоритмізувати та запрограмувати елементи навчального тренажера з побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей.

Для досягнення мети були поставлені такі завдання:

- ознайомитися з літературою по побудові математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей;
- розробка алгоритму тренажеру;
- складання блок-схеми алгоритму;
- програмування елементів навчального тренажеру;
- перевірка працездатності програми.

Об’єктом дипломної роботи є побудова математичної моделі задачі.

Предметом дипломної роботи є алгоритмізація та програмування елементів навчального тренажеру з побудови математичної моделі оптимізації добових обсягів виробництва деталей.

При реалізації дипломної роботи використано середовище візуальної розробки програм Microsoft Visual Studio та об’єктно-орієнтована мова програмування C#.

Пояснювальна записка до дипломної роботи складається з двох розділів, а саме теоретичного, який містить в собі підрозділи: постановка задачі, алгоритм тренажеру та блок-схеми; практична частина, містить в собі, — опис процесу програмного реалізації та опис програми.

РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ

1.1. Змістова постановка задачі

Задачею дипломної роботи є алгоритмізація та програмування елементів навчального тренажера з побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей. Розробити елементи повноцінного тренажера для навчання студентів побудові математичних моделей. Тренажер створюється за прикладом застосування даного методу з курсу лекцій «Методи оптимізації та дослідження операцій»[1-2].

Щоб скористатися математичною моделлю для конкретної виробничо-економічної ситуації, слід застосувати інформаційну технологію. Інформаційна технологія дозволяє безпомилково виділити з безлічі реальних виробничо-економічних ситуацій саме ту, яка повністю відповідає конкретним обставинам. Ця технологія складається з наступних восьми етапів.

1. Вибір об'єкта моделювання (наприклад: склад готової деталей; організація випуску нової деталей або системи транспортних перевезень і т.п.).

2. Аналіз проблемної ситуації, що склалася в даному об'єкті моделювання. Наприклад, для нормального функціонування складу готової деталей необхідно пов'язати швидкість споживання деталей з часом поставки і розмірами складських площ, обіговими коштами, які завжди виявляються обмеженими.

3. Тип і число спостережуваних параметрів (відшукуються значень ЦФ і основних змінних X_j), визначення яких дозволить вибрати обґрунтоване управління конкретного економічного об'єкта.

4. Тип і число спостережуваних параметрів (що задаються значень правих частин обмежень b_i, коефіцієнтів витрат a_{ij}, граничних умов для відшукуються змінних.

5. Умова адекватності, тобто, впевненість в тому, що математична модель економічного об'єкта повністю (або в головних рисах) характеризує його дійсне

оптимальне функціонування. Зазвичай адекватність ставиться в залежність від чисельного значення критерію оптимальності (або декількох таких критеріїв при багатокритеріальної оптимізації).

6. Використовуваний математичний апарат, відповідний конкретному математичному опису виробничо-економічної ситуації. (Наприклад, аналітичні зв'язку між основними параметрами руху запасів).

7. Аналіз результатів моделювання економічного об'єкта: оптимальних значень основних змінних і цільової функції. Ці значення становлять основу економічного аналізу конкретного об'єкта, за яким слідують висновки.

8. Прийняття рішення. За результатами оптимальних значень і зроблених на етапі 7 висновків приймається рішення по управлінню економічним об'єктом.

Тепер побудуємо математичну модель для вирішення задачі оптимізація добових обсягів виробництва деталей, і покажемо, що ця задача відноситься до задач лінійного програмування.

Нехай підприємство виготовляє два види деталей - Д1 і Д2, яка надходить в оптову продаж. Для виробництва деталей використовуються два види сировини - А і В. Максимально можливі добові запаси сировини становлять 9 і 13 одиниць відповідно. Витрати сировини на одиницю деталей виду Д1 і виду Д2 дані в табл. 1.

Таблиця 1.1. Витрати сировини

Сировина	Витрати сировини на 1 од. деталей		Запас сировини, од.
	Д1	Д2	
А	2	3	9
В	3	2	13

Вивчення ринку збуту показало, що добовий попит на продукцію Д1 ніколи не перевищує попиту на продукцію Д2 більш, ніж на 1 од. Крім того, встановлено, що попит на продукцію Д2 ніколи не перевищує 2 од. на добу.

Оптові ціни одиниці деталей дорівнюють: 3 ден. - для Д1 і 4 ден. - для Д2.

Необхідно побудувати математичну модель, яка дозволяє встановити, яку кількість деталей кожного виду потрібно виготовити, щоб прибуток від реалізації деталей був максимальним.

Перш ніж побудувати математичну модель задачі, тобто записати її за допомогою математичних символів, необхідно чітко розібратися з економічною ситуацією, описаної в умови. Для цього необхідно з точки зору економіки, а не математики, відповісти на наступні питання:

1) Які величини необхідно знайти в задачі?

2) Яка мета вирішення? Який параметр завдання служить критерієм ефективності (оптимальності) рішення, наприклад, прибуток, собівартість, час і т.д. В якому напрямку повинно змінюватися значення цього параметра (до max або до min) для досягнення найкращих результатів?

3) Які умови щодо шуканих величин і ресурсів завдання повинні бути виконані?

Ці умови встановлюють, як повинні співвідноситися один з одним різні параметри завдання, наприклад, кількість ресурсу, витраченого при виробництві, і його запас на складі; кількість деталей що виготовляється і ємність складу, де вона буде зберігатися; кількість деталей, яка виготовляється і ринковий попит на цю продукцію і т.д.

Тільки після економічного відповіді на всі ці питання можна приступати до запису цих відповідей в математичному вигляді, тобто до запису математичної моделі.

1) Шукані величини є змінними завдання, які, як правило, позначаються малими латинськими літерами з індексами, наприклад, однотипні змінні зручно представляти у вигляді $X = (x_1, x_2, ..., x_n)$.

2) Мета рішення записується у вигляді цільової функції, що позначається, наприклад, $f(x)$. Математична формула ЦФ $f(x)$ відображає спосіб розрахунку значень параметра - критерію ефективності задачі.

3) Умови, які накладаються на змінні і ресурси завдання, записуються у вигляді системи рівностей або нерівностей, тобто обмежень. ліві і праві частини

обмежень відображають спосіб отримання (розрахунок або чисельні значення з умови задачі) значень тих параметрів завдання, на які було накладено відповідні умови.

В процесі запису математичної моделі необхідно вказувати одиниці виміру змінних задачі, цільової функції і всіх обмежень. Побудуємо модель нашої задачі використовуючи описану методику.

Змінні завдання

У задачі потрібно встановити, скільки деталей кожного виду необхідно виготовити. Тому шуканими величинами, а значить, і змінними задачі є добові обсяги виробництва кожного виду деталей:

x_1 - добовий обсяг виробництва деталей Д1, [од. / добу.];

x_2 - добовий обсяг виробництва деталей Д2, [од. / добу.]

Цільова функція (ЦФ)

В умові задачі сформульована мета - досягти максимального доходу від реалізації деталей. Тобто критерієм ефективності служить параметр добового доходу, який повинен прагнути до максимуму. Щоб розрахувати величину добового доходу від продажу деталей обох видів, необхідно знати обсяги виробництва деталей, тобто x_1 і x_2 одиниць на добу, а також оптові ціни на продукцію Д1 і продукцію Д2. Таким чином, дохід від продажу добового обсягу виробництва деталей Д1 дорівнює 3х1 ден. од. на добу, а від продажу деталей Д2 - 2х2 ден. од. на добу. Тому запишемо ЦФ у вигляді суми доходу від продажу деталей Д1 і Д2 (при допущенні незалежності обсягів збуту кожної з продукцій).

$(X) = 3x1 + 4x2 \rightarrow \max$ [ден.од./доб].

Обмеження

Можливі обсяги виробництва деталей x_1 і x_2 обмежуються такими умовами:

- кількість сировини А і В, витрачений протягом доби на виробництво деталей обох видів, не може перевищувати добового запасу цієї сировини на складі;

- згідно з результатами вивчення ринкового попиту добовий обсяг виробництва деталей Д1 може перевищувати обсяг виробництва деталей Д2, але

не більше, ніж на 1 од. деталей;

- обсяг виробництва деталей Д2 не повинен перевищувати 2 од. на добу, що також випливає з результатів вивчення ринків збуту;

- обсяги виробництва деталей не можуть бути негативними.

Таким чином, всі обмеження задачі поділяються на три групи, обумовлені:

1) витратами сировини;

2) ринковим попитом на продукцію;

3) нейтрально обсягів виробництва.

Обмеження на витрати кожного з видів сировини мають наступну змістовну форму запису.

$$\begin{pmatrix}\text{Затрати конкретного сировини}\\ \text{на виробництво двох видів деталей}\end{pmatrix} \leq \begin{pmatrix}\text{Максимально можливий}\\ \text{запас даної сировини}\end{pmatrix}$$

Запишемо ці обмеження в математичній формі.

Ліва частина обмеження - це формула розрахунку добових витрат конконкретної сировини на виробництво деталей. Так, з умови відомі витрати сировини А на виробництво 1 од. деталей Д1 і 1 од. деталей Д2 (див. табл. 1.1). Тоді на виробництво x_1 од. деталей Д1 і x_2 од. деталей Д2 потрібно 2x1 + 3x2 од. сировини А.

Права частина обмеження - це величина добового запасу сировини на складі, наприклад, 9 од. сировини А на добу Таким чином, обмеження на витрати А має вигляд.

$$2x_1 + 3x_2 \leq 9$$

Аналогічна математична запис обмеження на витрати В

$$3x_1 + 2x_2 \leq 13$$

Примітка. Слід завжди перевіряти розмірність лівої і правої частини кожного з обмежень, оскільки їх розбіжність свідчить про принципової помилку при складанні обмежень.

Обмеження щодо добового обсягу виробництва деталей Д1 в порівнянні з обсягом виробництва деталей Д2 має змістовну форму

$$\begin{pmatrix}\text{Перевищення об'єму виробництвадеталей Д1}\\ \text{над об'ємом виробництва деталей Д2}\end{pmatrix} \leq \left(1\frac{\text{од. деталей}}{\text{доба}}\right)$$

і математичну формулу

$$x_1 - x_2 \leq 1$$

Обмеження щодо добового обсягу виробництва деталей Д2 має змістовну форму

$$(\text{Попит на продукцію}) \leq \left(2\frac{\text{од. деталей}}{\text{доба}}\right)$$

математичну форму

$$x_2 \leq 1$$

Невід'ємність обсягів виробництва задається як

$$x_1 \geq 0;\ x_2 \geq 0$$

Таким чином, математична модель цієї задачі має вигляд

$$f(x) = 3x_1 + 4x_2 \to max$$

$$\begin{cases} 2x_1 + 3x_2 \leq 9, \\ 3x_1 + 2x_2 \leq 13, \\ x_1 - x_2 \leq 1, \\ x_2 \leq 2, \\ x_1 \geq 0, x_2 \geq 0. \end{cases}$$

2.ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1. Огляд робіт.

Під час написання дипломної роботи було розглянуто декілька тренажерів студентів, а саме студента Томченка О.В. , Івахової Ю. С.

Тренажер Томченко О.В. на тему “Перший алгоритм Гоморі ” з дисципліни “Методи оптимізації та дослідження операцій” був розроблений на мові програмування Java у середовищі NetBeans. При запуску тренажера студент бачить простий дизайн інформацію про розробника та керівника, також присутня одна кнопка після натиску якої відбудеться проходження тренажеру.

Навчальний тренажер студентки Івахової Ю. С. [..] на тему «Матриця суміжності та інцидентності» для дистанційного навчального курсу «Дискретної математики». Тренажер написаний на об’єктно-орієнтованій мові програмування C#. Після запуску навчального тренажеру є можливість вибрати один з шістьох типів задач для розв’язку прикладу для тренування.

2.2. Позитивні аспекти оглянутих робіт.

1. У цих робіт досить простий та зрозумілий інтерфейс який не має нічого зайвого що б могло заважати.

2. Ці тренажери можна пройти без доступу до інтернету, що надає їм перевагу над іншими тренажерами, також вони займають мало місця на жорсткому диску.

3. В цих тренажерах досить просто навчитися тому, чого не розумієш або не знаєш.

4. Не до всіх запитань є можливість повторного введення даних.

2.3. Вади розробок з оглянутих робіт.

1. В тренажерах використовується не сучасний дизайн.

2. Для запуску тренажерів необхідне додаткове програмне забезпечення.

3. В описі завдань використовується досить малий шрифт.

3.ТЕОРЕТИЧНА ЧАСТИНА

3.1 Алгоритм тренажера

Для того, щоб розпочати роботу алгоритму тренажера, користувач мусить вибрати в головному меню програми «Навчання» - > «Пройти навчання».

На екрані по черзі будуть з’являтися питання з варіантами відповідей, серед яких буде лише одна правильна, якщо студент обрав правильну відповідь, то програма запропонує відповісти на наступне питання, а якщо ні то виводиться повідомлення «Відповідь є неправильною».

Правильні відповіді виділені «курсивом».

Крок 1. З’являється перше питання «Закон упорядкованості руху матеріального потоку у виробництві базується на:»

а) стандартизації і типізації міжцехових і внутрішньо цехових технологічних маршрутів;

б) співставленні витрат виробництва від години простою робочого місця і години простою партії виробничої програми;

в) виявленні причинно-наслідкових зв’язків між параметрами виробничої програми і структурою елементів виробництва;

г) вірні відповіді а), б) і в).

Крок 2. Після правильної відповіді відкривається наступне питання «Закон безперервності ходу виробничого процесу базується на:»

а) стандартизації і типізації міжцехових і внутрішньоцехових технологічних маршрутів;

б) співставленні витрат виробництва від години простою робочого місця і години простою партії предметів праці;

в) виявленні причинно-наслідкових зв’язків між параметрами виробничої програми і структурою елементів виробництва;

г) вірні відповіді а), б) і в).

Крок 3. З'являється наступне питання: «Закон ритму виробничого циклу виготовлення виробу базується на:»

а) стандартизації і типізації міжцехових і внутрішньоцехових технологічних маршрутів;

б) співставленні витрат виробництва від години простою робочого місця і години простою партії предметів праці;

в) виявленні причинно-наслідкових зв’язків між параметрами виробничої програми і структурою елементів виробництва;

г) вірні відповіді а), б) і в).

Крок 4. «Принципи організації внутрішньовиробничих логістичних систем включають:»

а) паралельність;

б) пропорційність;

в) гнучкість;

г) вірні відповіді а), б) і в).

Крок 5. «Яке твердження є вірним:»

а) в умовах непотокового виробництва мінімальний рівень витрат може бути забезпечений за рахунок організації безперервного завантаження робочих

місць;

б) в умовах потокового виробництва мінімальний рівень витрат може бути забезпечений за рахунок організації безперервного завантаження робочих місць;

в) в умовах непотокового виробництва мінімальний рівень витрат може бути забезпечений за рахунок мінімізації часу міжопераційного очікування деталей;

г) вірні відповіді а) і в).

Крок 6. «Суть “гнучкого виробництва” (lean production) полягає у:»

а) виявленні вузьких місць як шансу їх повної ліквідації;

б) можливості швидко складати виробничі плани та графіки;

в) можливість здійснення моделювання;

г) вірні відповіді б) та в).

Крок 7. «Системи, “що штовхають”, у сфері виробництва:»

а) побудовані за принципом подачі матеріалів, деталей та вузлів у виробничий процес по команді центральної системи управління;

б) спрямовані на випередження (по відношенню до попиту) формування товарних запасів на складах гуртових та роздрібних підприємств;

в) побудовані за принципом подачі матеріалів, деталей та вузлів з попередньої технологічної операції на наступну відповідно до замовлення ланки, яка виконує наступну операцію;

г) спрямовані на випередження стимулювання попиту в роздрібній торговельній ланці.

Крок 8. «Системи, “що тягнуть”, у сфері виробництва:»

а) побудовані за принципом подачі матеріалів, деталей та вузлів у виробничий процес по команді центральної системи управління;

б) спрямовані на випередження (по відношенню до попиту) формування товарних запасів на складах гуртових та роздрібних підприємств;

в) побудовані за принципом подачі матеріалів, деталей та вузлів з попередньої технологічної операції на наступну;

відповідно до замовлення ланки, яка виконує наступну операцію;

г) спрямовані на випередження стимулювання попиту в роздрібній торговельній ланці.

Крок 9. «В ланцюгу доданої вартості (цінності) виробництво формує:»

а) цінність форми;

б) цінність часу;

в) цінність місця;

г) цінність володіння.

3.2 Блок-схеми

На рисунках (1.1)-(1.2) показано блок-схеми алгоритму навчального тренажера з побудови математичної моделі задачі вибору шляхів оптимізації добових обсягів виробництва деталей. На рисунку 1.1 — показана блок-схема алгоритму всього тренажера. На рисунку 1.2 — блок-схема формування тестування.

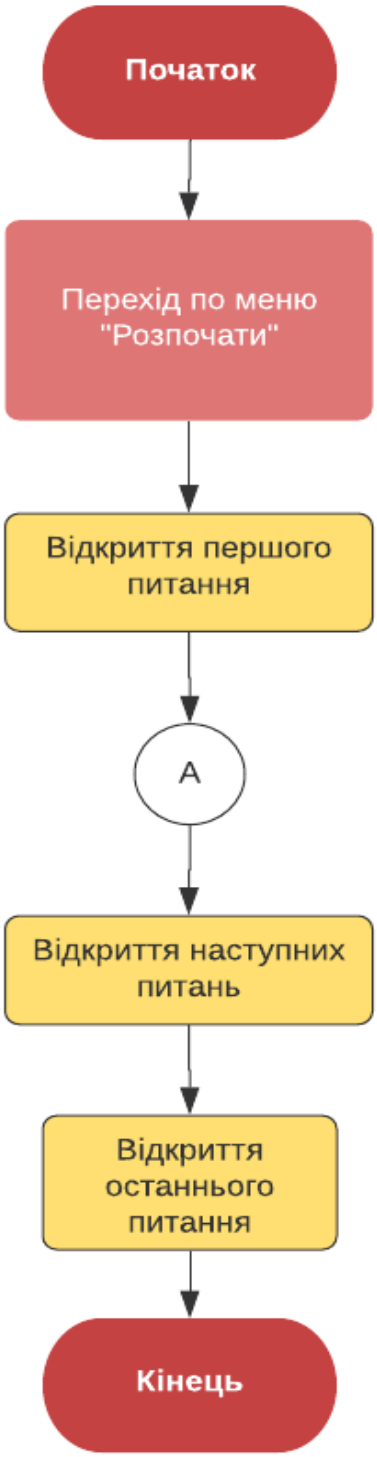

Рисунок 1.1 — Блок-схема алгоритму навчального тренажера

Рисунок 1.2 — Блок-схема обробки питання з правильною відповіддю

РОЗДІЛ 4. ПРАКТИЧНА ЧАСТИНА

4.1 Опис процесу програмної реалізації

Для виконання поставленої задачі було застосовано об’єктно-орієнтовану мову програмування C# із застосуванням середовища візуальної розробки програм Microsoft Visual Studio.

Роботу розпочато зі створення нового проекту “SimulatorAlgorithm”. До проекту було додано батьківське вікно «SimulatorAlgorithmMDI» та три форм класу Windows Form, як зображено на рисунку 2.1.

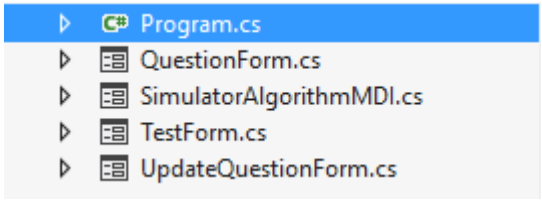

Рисунок 2.1 — Вікна додатку

В батьківському вікні в залежності від вибору меню створюються екземпляри класу «TestForm» або «QuestionForm».

1. Форма «TestForm» призначена для реалізації алгоритму тренажера оптимізації добових обсягів виробництва деталей.
2. Форма «QuestionForm» – для створення, редагування та видалення запитань тринажеру.

Так, як неможливо передбачити всіх нюансів виробництва, яке динамічно змінюється, в даній програмі реалізована можливість розширювати знання тренажера шляхом заміни чи додавання нових питань.

Для зберігання питань та відповідей використовується система управління базами даних Microsoft Access, в якій я створив базу даних з назвою «DataBase».

У формі «TestForm» проводиться завантаження всіх питань із бази даних, після чого по черзі виводиться на екран для користувача.

Для того, щоб повноцінно реалізувати алгоритм тренажера та відобразити інформацію, були використані наступні інструменти: CheckBox, TextBox та

Button.

На рисунку 2.2. показано код реалізації перевірки правильної відповіді та закінчення всіх запитань.

```
private void NextBtn_Click(object sender, EventArgs e) {
  if (((AnswerABoolCBox.Checked) && (_selectedQuestion.AnswerABool))
    || ((AnswerBBoolCBox.Checked) && (_selectedQuestion.AnswerBBool))
    || ((AnswerCBoolCBox.Checked) && (_selectedQuestion.AnswerCBool))
    || ((AnswerDBoolCBox.Checked) && (_selectedQuestion.AnswerDBool))) {
    QuestionNumber++;
    if (QuestionNumber <= MaxQuestion) {
      GetQuestionByQuestionNumber(QuestionNumber);
    } else {
      MessageBox.Show("Запитання закінчилися!", "Увага", MessageBoxButtons.OK);
      this.Close();
    }
  } else {
    MessageBox.Show("Ви відповіли неправильно!", "Увага", MessageBoxButtons.OK);
  }
}
```

Рисунок 2.2 — Перевірка вибору правильної відповіді

На рисунку 2.3. показано код реалізації методу «GetQuestionByQuestionNumber», що використовується для переходу на наступне питання.

```
private void GetQuestionByQuestionNumber(int QuestionNumber) {
  for (int i = 0; i < _QuestionList.Count(); i++) {
    if (QuestionNumber == _QuestionList[i].Number) {
      _selectedQuestion = _QuestionList[i];
      QuestionNameTBox.Text = _selectedQuestion.QuestionName;
      AnswerATBox.Text = _selectedQuestion.AnswerA;
      AnswerBTBox.Text = _selectedQuestion.AnswerB;
      AnswerCTBox.Text = _selectedQuestion.AnswerC;
      AnswerDTBox.Text = _selectedQuestion.AnswerD;
      AnswerABoolCBox.Checked = false;
      AnswerBBoolCBox.Checked = false;
      AnswerCBoolCBox.Checked = false;
      AnswerDBoolCBox.Checked = false;
    }
  }
}
```

Рисунок 2.3 — код реалізації методу «GetQuestionByQuestionNumber»

Для завантаження та обробки даних з бази даних було реалізовано два класи: «Question» та «QuestionProvider».

Клас «Question» містить в собі інформацію про одне запитання. Його діаграма зображена на рисунку 2.4.

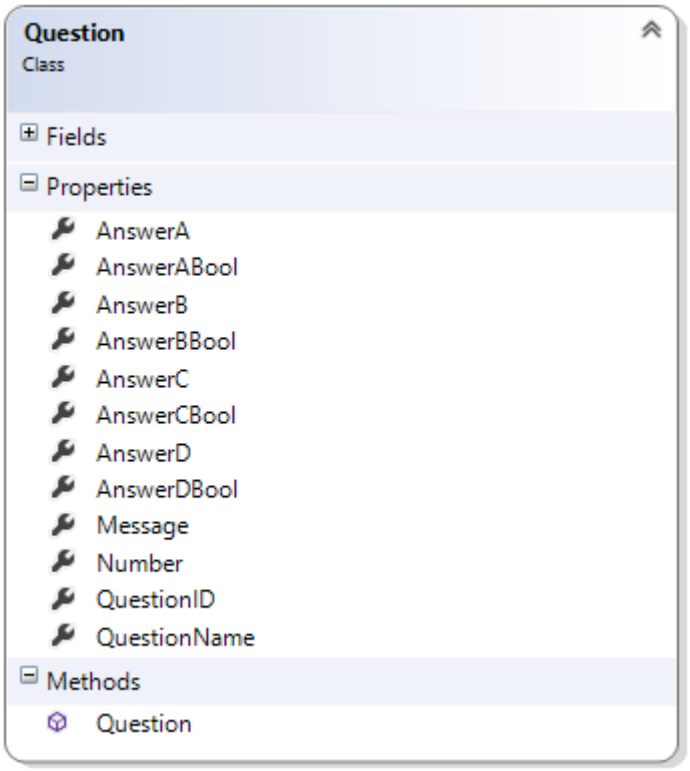

Рисунок 2.4 — Діаграма класу «Question»

Клас «QuestionProvider» містить в собі 5 методів для роботи з таблицею бази даних «Question». Його діаграма зображена на рисунку 2.5.

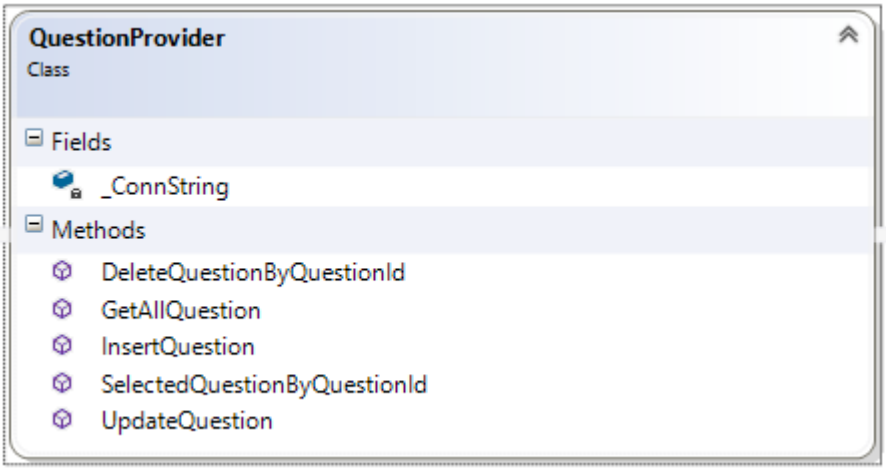

Рисунок 2.5 — Діаграма класу «QuestionProvider»

Методи класу «QuestionProvider» призначені:

1. Метод «DeleteQuestionByQuestionId» – видалення запитання.
2. Метод «GetAllQuestion» – повертає всі питання тренажера.
3. Метод «InsertQuestion» призначений для додавання нового запитання.
4. Метод «SelectedQuestionByQuestionId» призначений для вибору

конкретного питання.

5. Метод «UpdateQuestion» призначений для редагування вибраного питання.

4.2 Опис програми

При запуску програми у користувача з’являється батьківське вікно з меню програми, в якому пропонується або розширити знання тренажера або пройти навчання.

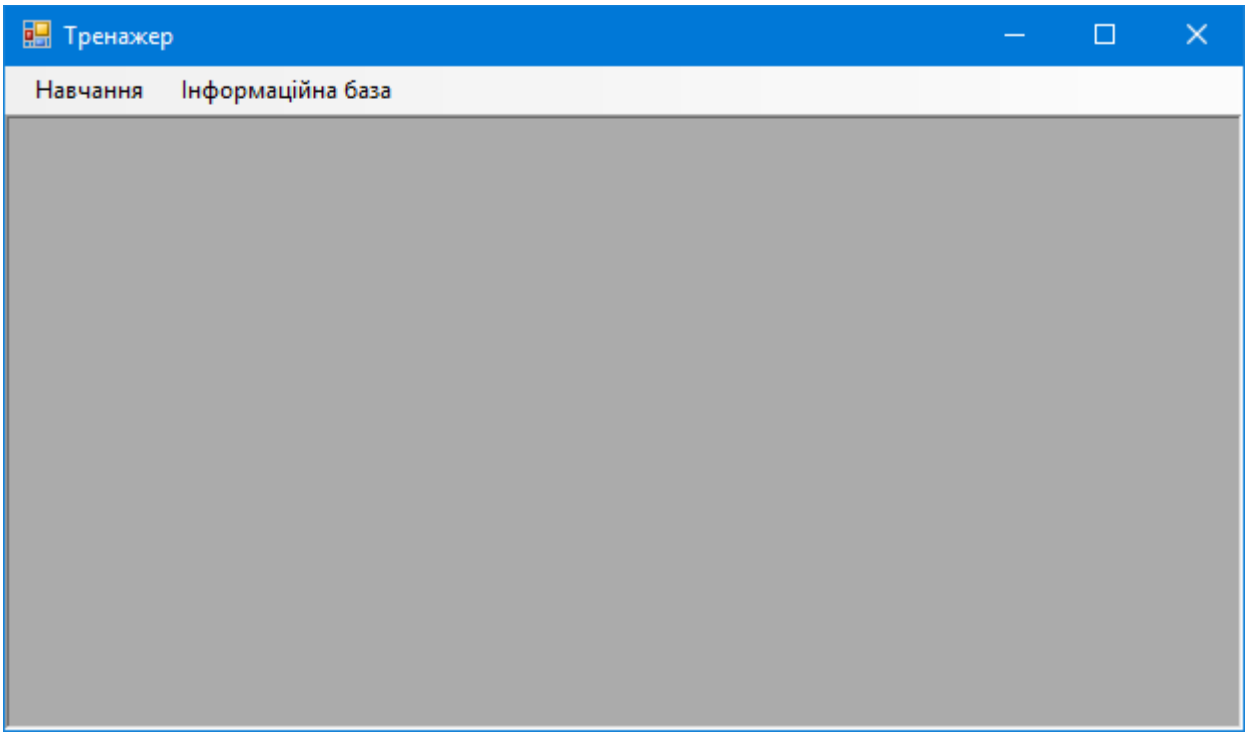

Рисунок 2.6 — Батьківське вікно програми

Після переходу по меню «Навчання» –> «Пройти навчання» виводиться форма, яка показана на рисунку 2.7 із першим питанням та варіантами відповіді на нього.

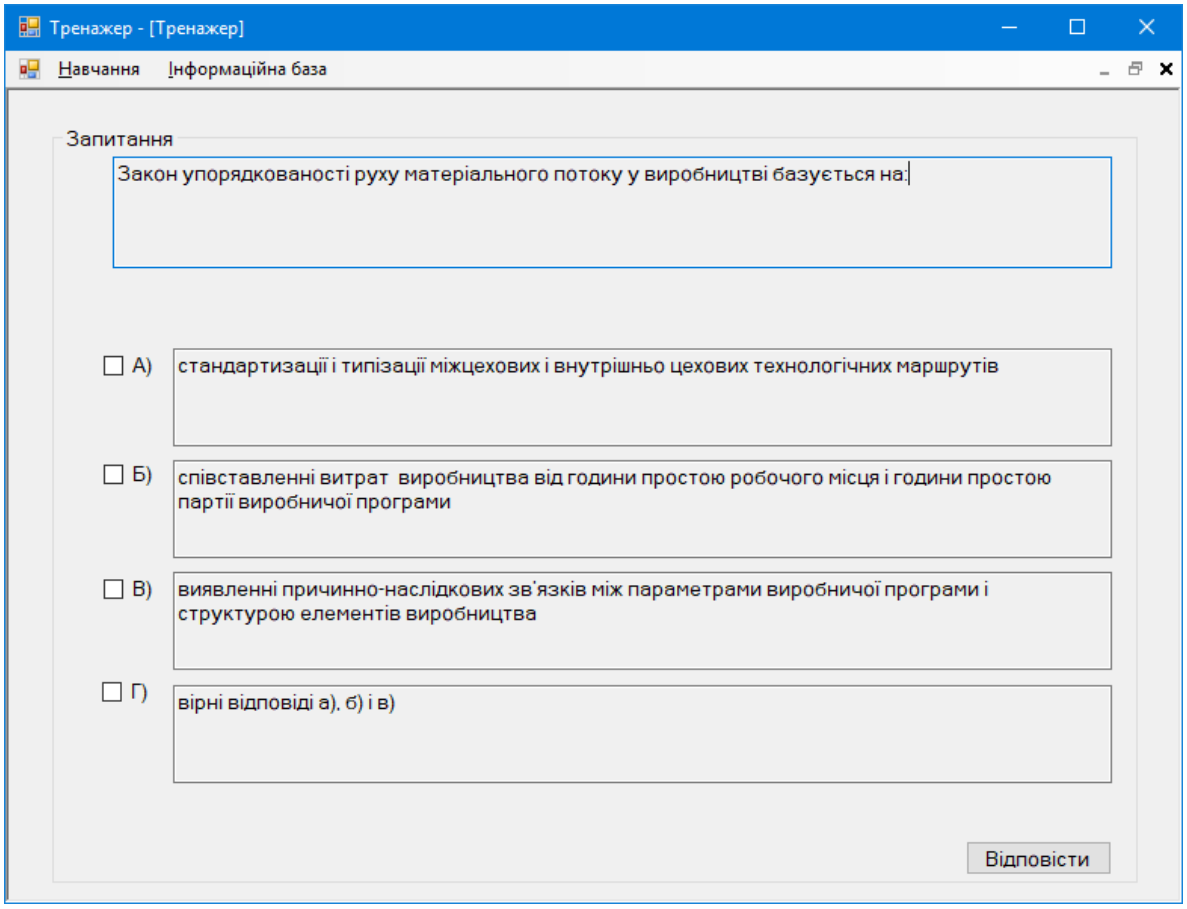

Рисунок 2.6 — Форма роботи тренажера

Для того, щоб надати відповідь необхідно вибрати один із варіантів та натиснути «Відповісти». Якщо відповідь неправильно, виведеться наступне повідомлення, яке показано на рисунку 2.7.

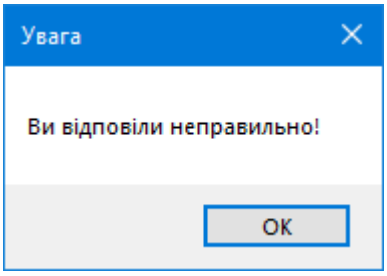

Рисунок 2.7 — Діалогове вікно, якщо відповідь невірна

Якщо ж користувач вибере правильну відповідь, програма перейде до наступного питання і так до того часу, поки не будуть пройдені всі питання.

Так, як було описано вище в системі можна розширяти та міняти питання для тренажеру. Для цього в головному меню програми необхідно перейти «Інформаційна база» –> «Запитання алгоритму тренажера», після чого побачимо форму зі списком всіх питань та можливістю додати нове або редагувати старі.

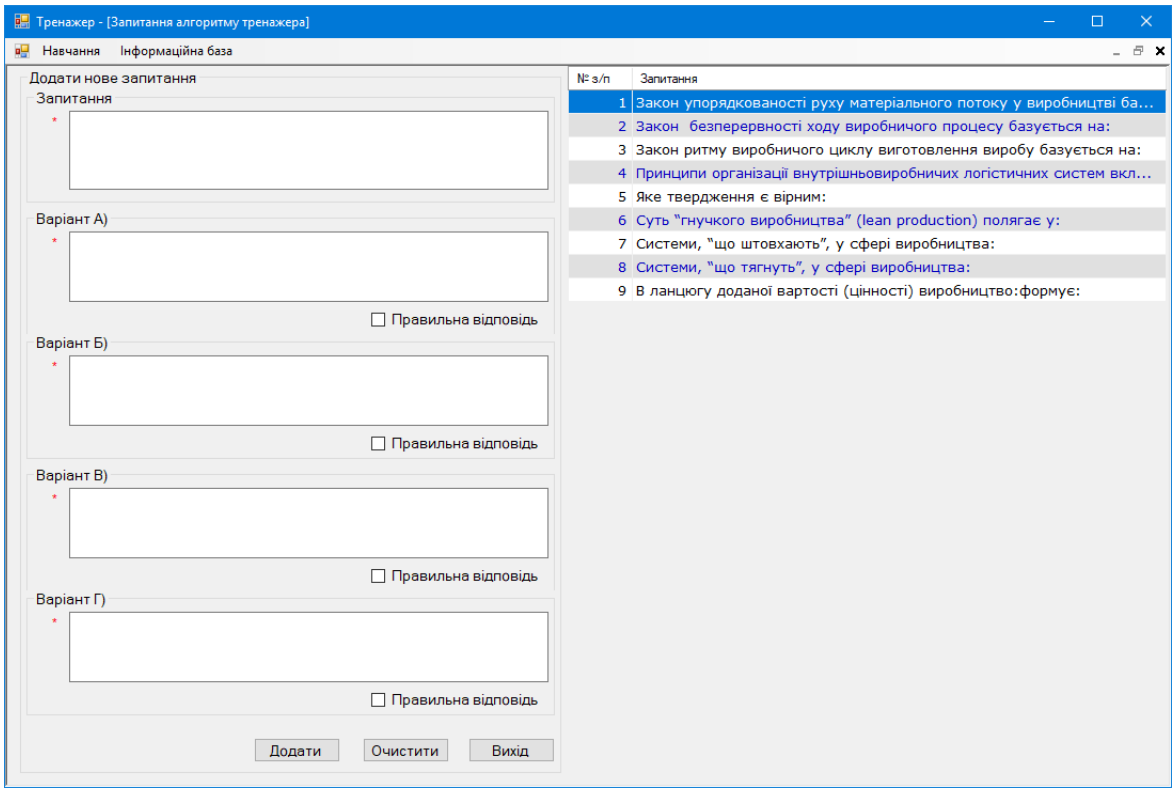

Рисунок 2.8 — Форма для розширення можливостей тренажера

Для того, щоб додати нове питання необхідно:

- Вказати запитання.
- Вказати варіанти відповіді на нього.
- Вказати вірну відповідь.

Після заповнення всіх даних необхідно натиснути кнопку «Додати», після чого в системі з’явиться нове питання.

Також, якщо в якомусь питанні допущена помилка, реалізована можливість редагування вибраного із списку питання. Для цього необхідно клікнути лівою кнопкою мишки по необхідному питанню, після чого з’явиться можливість його редагування чи видалення.

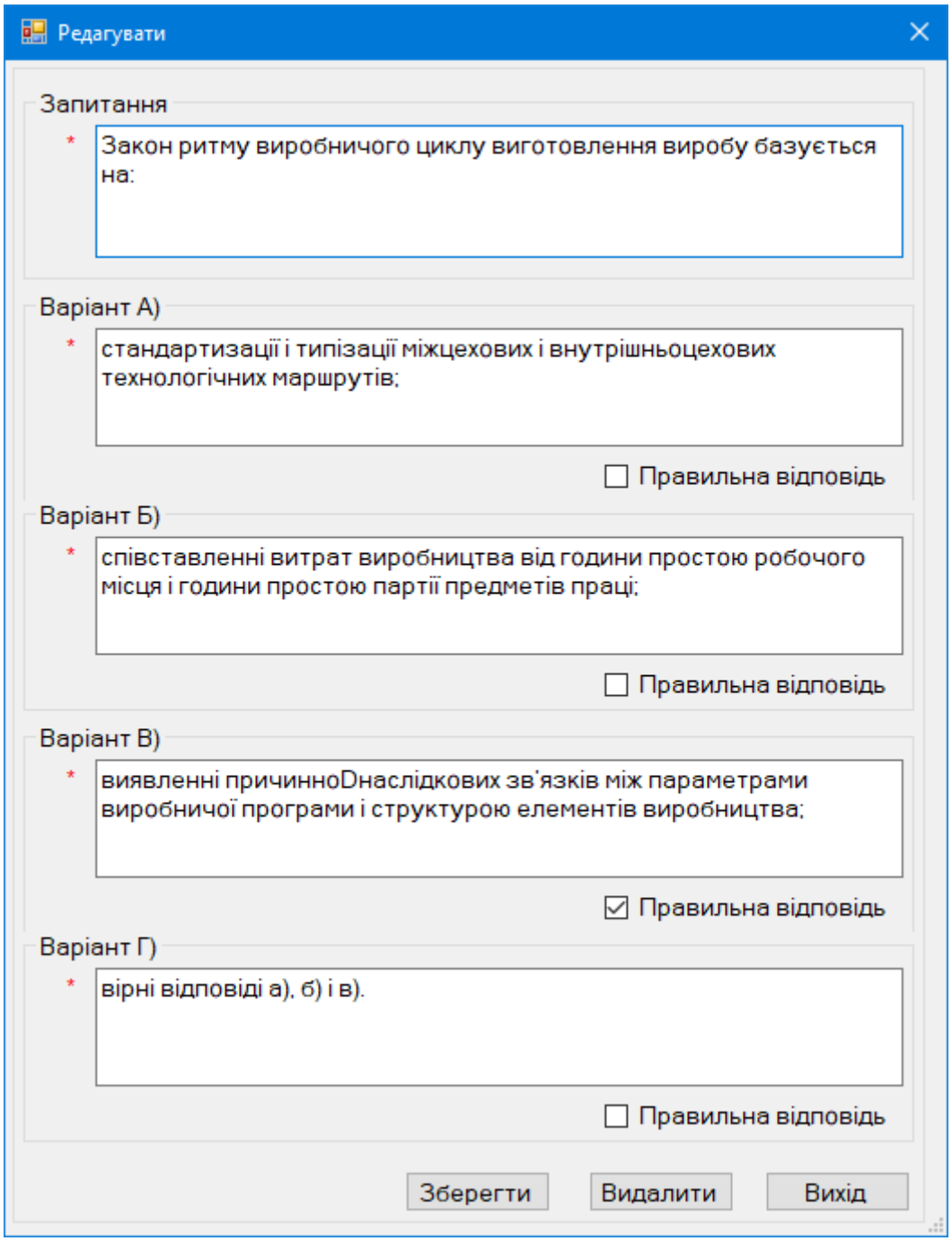

Рисунок 2.9 — Форма для редагування запитання тренажера

ВИСНОВОК

При виконанні дипломної роботи було розроблено алгоритм навчального тренажера оптимізації добових обсягів виробництва деталей та створено програму навчального тренажеру з даної теми.

Створений навчальний тренажер планується впровадити в дистанційний курс для навчання студентів.

Поставлена мета і завдання в ході виконання дипломної роботи досягнута.

СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1.Ємець О. О. Методи оптимізації та дослідження операцій: навчально-методичний посібник / О.О. Ємець. – Полтава: РВВ ПУСКУ, 2009. – 76 с.

2.Ємець О. О. Навчально методичний посібник з курсом лекцій для самостійного вивчення дисципліни «Методи оптимізації та дослідження операцій» за кредитно-модульною системою організації навчального процесу / О.О. Ємець, Т.О. Парфьонова. – Полтава: РВВ ПУЕТ, 2013. – 492 с.

3.Ємець О. О. Методичні рекомендації щодо оформлення пояснювальних записок до курсових проектів (робіт) для студентів за освітньою програмою «Комп'ютерні науки» — Полтава: РВВ ПУЕТ, 2017. — 69 с.

4. Вікіпедія. Режим доступу: https://uk.wikipedia.org/wiki/ Microsoft _Visual_Studio

5.Гальчун А.М. програмна реалізація тренажера з побудови математичної моделі задачі вибору плану обслуговування клієнтів фінансового ринку з дисципліни «Методи оптимізації та дослідження операцій» / А.М. Гальчун, О. О. Ємець. – Полтава: Кафедра математичне моделювання та соціальна інформатика ПУЕТ, 2019. – 3с. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/7013

6.Методи оптимізації та дослідження операцій (Частина 2) 2017-2018н.р. : [дистанційний курс] / О. О. Ємець // Головний центр дистанційного навчання, Полтава: ПУЕТ- Режим доступу: http://www2.el.puet.edu.ua/st/mod/url/view.php?id=73464Кристофидес Н. Теория графов. Алгоритмический подход / Кристофидес Н. – М: Мир, 1978. – С. 75 – 94.

7.Abramson D. A Simulated Annealing Code for General Integer Linear Programs / D. Abramson, M. Randall // Annals of Operations Research. – 1999. – V. 86, P. 3 – 24.

8. Ємець О.О. Методчині рекомендації до виконання дипломної роботи для

студентів ступені магістра спеціальності 122 «Комп'ютерні науки» / О.О. Ємець, Т.О. Парфьонова – Полтава: РВВ ПУЕТ, 2018.–35с.

9. Кристиан Нейгел и др. C# 5.0 и платформа .NET 4.5 для профессионалов = Proffessional C# 5.0 and .NET 4.5. – М.: «Диалектика», 2013. – 1440с.

10. Економічний ризик: методи оцінки та управління : Навч. посіб. / Т.А. Васильєва, С.В. Лєонов, Я.М. Кривич та ін.; під заг. ред. д-ра екон. наук, проф. Т.А. Васильєвої, канд. екон. наук Я.М. Кривич. – Суми : ДВНЗ "УАБС НБУ", 2015. – 208 с.

11. Останкова Л.А. Аналіз, моделювання та управління економічними ризиками / Л.А. Останкова, Н.Ю. Шевченко : Навч. посіб. – К.: Центр учбової літератури, 2011. – 256 с.

12. Томченко О.В. Тренажер з теми "Перший алгоритм Гоморі"/ О.В. Томченко //Ємець О.О. Методи оптимізації та дослідження операцій (Частина 2) 2017-2018н.р. : [дистанційний курс] / О. О. Ємець // Головний центр дистанційного навчання, Полтава: ПУЕТ- Режим доступу: http://www2.el.puet.edu.ua/st/mod/url/view.php?id=73464

13. Івахова Ю. С. Програмне забезпечення для тренажера з теми: «Матриця суміжності а інцидентості» дистанційного навчального курсу «Дискретна математика» / Ю. С. Івахова // Інформатика та системні науки (ІСН-2013): матеріали IV Всеукр. наук. – практ. конф., (м. Полтава, 21-23 берез. 2013 р.). – Полтава: ПУЕТ, 2013. – С. 136-139. – Режим доступу: http://dspace.puet.edu.ua//handle/123456789/1552

ДОДАТОК А
ПРОГРАМНИЙ КОД

Код реалізації класу «SimulatorAlgorithmMDI»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SimulatorAlgorithm {
  public partial class SimulatorAlgorithmMDI : Form {

    public SimulatorAlgorithmMDI() {
      InitializeComponent();
    }

    private void testSetupToolStripMenuItem_Click(object sender, EventArgs e) {
      CloseAllWindows();
      TestForm testForm = new TestForm();
      testForm.MdiParent = this;
      testForm.WindowState = FormWindowState.Maximized;
      testForm.Show();
    }

    private void exitToolStripMenuItem_Click(object sender, EventArgs e) {
      this.Close();
```

```
    }

    private void запитанняДоТестівToolStripMenuItem_Click(object sender, EventArgs
e) {
      CloseAllWindows();
      QuestionForm questionForm = new QuestionForm();
      questionForm.MdiParent = this;
      questionForm.WindowState = FormWindowState.Maximized;
      questionForm.Show();
    }

    public void CloseAllWindows() {
      Form[] childArray = this.MdiChildren;
      foreach (Form childForm in childArray) {
        childForm.Close();
      }
    }
  }
}
```

Код реалізації класу «TestForm»

```
using SimulatorAlgorithm.DAL;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

```
namespace SimulatorAlgorithm {
  public partial class TestForm : Form {
    int QuestionNumber = 1;
    int MaxQuestion = 0;
    Question _selectedQuestion = new Question();
    private QuestionProvider _QuestionProvider = new QuestionProvider();
    private List<Question> _QuestionList = new List<Question>();

    public TestForm() {
      InitializeComponent();
      DataLoad();
      GetQuestionByQuestionNumber(QuestionNumber);
    }

    private void NextBtn_Click(object sender, EventArgs e) {
      if (((AnswerABoolCBox.Checked) && (_selectedQuestion.AnswerABool))
        || ((AnswerBBoolCBox.Checked) && (_selectedQuestion.AnswerBBool))
        || ((AnswerCBoolCBox.Checked) && (_selectedQuestion.AnswerCBool))
        || ((AnswerDBoolCBox.Checked) && (_selectedQuestion.AnswerDBool))) {
        QuestionNumber++;
        if (QuestionNumber <= MaxQuestion) {
          GetQuestionByQuestionNumber(QuestionNumber);
        } else {
          MessageBox.Show("Запитання закінчилися!", "Увага",
MessageBoxButtons.OK);
          this.Close();
        }
      } else {
        MessageBox.Show("Ви відповіли неправильно!", "Увага",
```

```
MessageBoxButtons.OK);
      }
    }
    private void DataLoad() {
       _QuestionList = _QuestionProvider.GetAllQuestion();
       MaxQuestion = _QuestionList.Count();
    }

    private void GetQuestionByQuestionNumber(int QuestionNumber) {
      for (int i = 0; i < _QuestionList.Count(); i++) {
       if (QuestionNumber == _QuestionList[i].Number) {
         _selectedQuestion = _QuestionList[i];
         QuestionNameTBox.Text = _selectedQuestion.QuestionName;
         AnswerATBox.Text = _selectedQuestion.AnswerA;
         AnswerBTBox.Text = _selectedQuestion.AnswerB;
         AnswerCTBox.Text = _selectedQuestion.AnswerC;
         AnswerDTBox.Text = _selectedQuestion.AnswerD;
         AnswerABoolCBox.Checked = false;
         AnswerBBoolCBox.Checked = false;
         AnswerCBoolCBox.Checked = false;
         AnswerDBoolCBox.Checked = false;
       }
      }
    }


    private void AnswerABoolCBox_CheckedChanged(object sender, EventArgs e) {
      AnswerBBoolCBox.Checked = false;
      AnswerCBoolCBox.Checked = false;
      AnswerDBoolCBox.Checked = false;
```

```
    }

    private void AnswerBBoolCBox_CheckedChanged(object sender, EventArgs e) {
      AnswerABoolCBox.Checked = false;
      AnswerCBoolCBox.Checked = false;
      AnswerDBoolCBox.Checked = false;
    }

    private void AnswerCBoolCBox_CheckedChanged(object sender, EventArgs e) {
      AnswerABoolCBox.Checked = false;
      AnswerBBoolCBox.Checked = false;
      AnswerDBoolCBox.Checked = false;
    }

    private void AnswerDBoolCBox_CheckedChanged(object sender, EventArgs e) {
      AnswerABoolCBox.Checked = false;
      AnswerBBoolCBox.Checked = false;
      AnswerCBoolCBox.Checked = false;
    }
  }
}
```

Код реалізації класу «QuestionForm»

```
using SimulatorAlgorithm.AppCode;
using SimulatorAlgorithm.DAL;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SimulatorAlgorithm {
  public partial class QuestionForm : Form {
    private int _selectedRowIndex = 0;
    private Validation _validation = new Validation();
    private QuestionProvider _QuestionProvider = new QuestionProvider();
    private List<Question> _QuestionList = new List<Question>();



    public QuestionForm() {
      InitializeComponent();
      DataLoad();
    }

    private void AddBtn_Click(object sender, EventArgs e) {
      if (IsDataEnteringCorrect()) {
        _QuestionProvider.InsertQuestion(QuestionNameTBox.Text, AnswerATBox.Text,
AnswerABoolCBox.Checked, AnswerBTBox.Text, AnswerBBoolCBox.Checked,
          AnswerCTBox.Text, AnswerCBoolCBox.Checked, AnswerDTBox.Text,
AnswerDBoolCBox.Checked);
        DataLoad();
        ClearAllControls();
      }
    }
    private void ClearBtn_Click(object sender, EventArgs e) {
      ClearAllControls();
    }
```

```
    private void ExitBtn_Click(object sender, EventArgs e) {
      this.Close();
    }
    private void QuestionGridView_CellClick(object sender,
DataGridViewCellEventArgs e) {
      if (e.RowIndex >= 0) {
        _selectedRowIndex = e.RowIndex;
        UpdateQuestionForm _updateQuestionForm = new
UpdateQuestionForm(Convert.ToInt32(QuestionGridView[0,
e.RowIndex].Value.ToString()));
        _updateQuestionForm.ShowDialog();
        DataLoad();
      }
    }

    private void QuestionGridView_MouseHover(object sender, EventArgs e) {
      QuestionGridView.Focus();
    }

    private void DataLoad() {
      int firstRowIndex = 0;
      if (QuestionGridView.FirstDisplayedScrollingRowIndex > 0) {
        firstRowIndex = QuestionGridView.FirstDisplayedScrollingRowIndex;
      }
      try {
        _QuestionList = _QuestionProvider.GetAllQuestion();
        LoadDataInQuestionGridView(_QuestionList);
        if (_selectedRowIndex == QuestionGridView.Rows.Count) {
          _selectedRowIndex = QuestionGridView.Rows.Count - 1;
        }
```

```
      if (_selectedRowIndex >= 0) {
        QuestionGridView.FirstDisplayedScrollingRowIndex = firstRowIndex;
        QuestionGridView.Rows[_selectedRowIndex].Selected = true;
      }
    } catch { }
  }

  private void LoadDataInQuestionGridView(List<Question> QuestionList) {
    QuestionGridView.DataSource = null;
    QuestionGridView.Columns.Clear();
    QuestionGridView.AutoGenerateColumns = false;
    QuestionGridView.RowHeadersVisible = false;

    QuestionGridView.DataSource = QuestionList;

    if (QuestionList.Count > 0) {
      if (QuestionList[0].Message == Names.NoDataNames.NoDataInQuestion) {
        DataGridViewColumn messageColumn = new DataGridViewTextBoxColumn();
        messageColumn.DataPropertyName = "Message";
        messageColumn.Width = QuestionGridView.Width -
Names.SizeOptins.MinusSizePanel;
        QuestionGridView.Columns.Add(messageColumn);
      } else {
        DataGridViewColumn AvtoIdColumn = new DataGridViewTextBoxColumn();
        AvtoIdColumn.DataPropertyName = "QuestionID";
        QuestionGridView.Columns.Add(AvtoIdColumn);
        QuestionGridView.Columns[0].Visible = false;

        DataGridViewColumn numberColumn = new DataGridViewTextBoxColumn();
        numberColumn.HeaderText = "№ з/п";
```

```
            numberColumn.DataPropertyName = "Number";
            numberColumn.DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight;
            numberColumn.Width = Names.SizeOptins.NumberSize;
            QuestionGridView.Columns.Add(numberColumn);

            DataGridViewColumn QuestionNameColumn = new
DataGridViewTextBoxColumn();
            QuestionNameColumn.HeaderText = "Запитання";
            QuestionNameColumn.DataPropertyName = "QuestionName";
            QuestionNameColumn.Width = Names.SizeOptins.Title;
            QuestionGridView.Columns.Add(QuestionNameColumn);

        }
        for (int i = 0; i < QuestionGridView.Columns.Count; i++) {
            QuestionGridView.Columns[i].HeaderCell.Style.BackColor = Color.LightGray;
        }
    }
}

private bool IsDataEnteringCorrect() {
    bool isCorrect = true;
    if (_validation.IsDataEntering(QuestionNameTBox.Text)) {
        QuestionNameValiadtionLbl.Text = Names.ProgramButtons.RequiredValidation;
    } else {
        QuestionNameValiadtionLbl.Text = Names.ProgramButtons.ErrorValidation;
        isCorrect = false;
    }
    if (_validation.IsDataEntering(AnswerATBox.Text)) {
        AnswerAValidationLbl.Text = Names.ProgramButtons.RequiredValidation;
```

```
  } else {
    AnswerAValidationLbl.Text = Names.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  if (_validation.IsDataEntering(AnswerBTBox.Text)) {
    AnswerBValidationLbl.Text = Names.ProgramButtons.RequiredValidation;
  } else {
    AnswerBValidationLbl.Text = Names.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  if (_validation.IsDataEntering(AnswerCTBox.Text)) {
    AnswerCValidationLbl.Text = Names.ProgramButtons.RequiredValidation;
  } else {
    AnswerCValidationLbl.Text = Names.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  if (_validation.IsDataEntering(AnswerDTBox.Text)) {
    AnswerDValidationLbl.Text = Names.ProgramButtons.RequiredValidation;
  } else {
    AnswerDValidationLbl.Text = Names.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  return isCorrect;
}

private void ClearAllControls() {
  QuestionNameTBox.Text = "";
  AnswerATBox.Text = "";
  AnswerBTBox.Text = "";
  AnswerCTBox.Text = "";
```

```
      AnswerDTBox.Text = "";
      AnswerABoolCBox.Checked = false;
      AnswerBBoolCBox.Checked = false;
      AnswerCBoolCBox.Checked = false;
      AnswerDBoolCBox.Checked = false;
    }
  }
}
```

Код реалізації класу «QuestionProvider» та «Question»

```
        using SimulatorAlgorithm.AppCode;
        using System;
        using System.Collections.Generic;
        using System.Data;
        using System.Data.OleDb;
        using System.Linq;
        using System.Text;
        using System.Threading.Tasks;

        namespace SimulatorAlgorithm.DAL {
          class QuestionProvider {
            private string _ConnString =
System.Configuration.ConfigurationSettings.AppSettings["CONNECT"];

            public void InsertQuestion(string QuestionName, string AnswerA, bool
AnswerABool, string AnswerB, bool AnswerBBool,
              string AnswerC, bool AnswerCBool, string AnswerD, bool AnswerDBool) {
              string SqlString = "INSERT INTO Question (QuestionName, AnswerA,
AnswerABool, AnswerB, AnswerBBool, AnswerC, AnswerCBool, AnswerD,
AnswerDBool " +
                ") Values(?, ?, ?, ?, ?, ?, ?, ?, ?)";
```

```
            using (OleDbConnection conn = new OleDbConnection(_ConnString)) {
              using (OleDbCommand cmd = new OleDbCommand(SqlString, conn)) {
                cmd.CommandType = CommandType.Text;
                cmd.Parameters.AddWithValue("QuestionName", QuestionName);
                cmd.Parameters.AddWithValue("AnswerA", AnswerA);
                cmd.Parameters.AddWithValue("AnswerABool", AnswerABool);
                cmd.Parameters.AddWithValue("AnswerB", AnswerB);
                cmd.Parameters.AddWithValue("AnswerBBool", AnswerBBool);
                cmd.Parameters.AddWithValue("AnswerC", AnswerC);
                cmd.Parameters.AddWithValue("AnswerCBool", AnswerCBool);
                cmd.Parameters.AddWithValue("AnswerD", AnswerD);
                cmd.Parameters.AddWithValue("AnswerDBool", AnswerDBool);
                conn.Open();
                cmd.ExecuteNonQuery();
                conn.Close();
              }
            }
          }

          public List<Question> GetAllQuestion() {
            int i = 0;
            string SqlString = "SELECT QuestionID, QuestionName, AnswerA,
AnswerABool, AnswerB, AnswerBBool, AnswerC, AnswerCBool, AnswerD,  " +
              "AnswerDBool FROM Question";

            List<Question> listAllQuestion = new List<Question>();
            using (OleDbConnection conn = new OleDbConnection(_ConnString)) {
              using (OleDbCommand cmd = new OleDbCommand(SqlString, conn)) {
                conn.Open();
```

```
                using (OleDbDataReader reader = cmd.ExecuteReader()) {
                  while (reader.Read()) {
                    Question oneQuestion = new Question();
                    oneQuestion.Number = ++i;
                    oneQuestion.QuestionID =
Convert.ToInt32(reader["QuestionID"].ToString());
                    oneQuestion.QuestionName = reader["QuestionName"].ToString();
                    oneQuestion.AnswerA = reader["AnswerA"].ToString();
                    oneQuestion.AnswerABool =
Convert.ToBoolean(reader["AnswerABool"]);
                    oneQuestion.AnswerB = reader["AnswerB"].ToString();
                    oneQuestion.AnswerBBool =
Convert.ToBoolean(reader["AnswerBBool"]);
                    oneQuestion.AnswerC = reader["AnswerC"].ToString();
                    oneQuestion.AnswerCBool =
Convert.ToBoolean(reader["AnswerCBool"]);
                    oneQuestion.AnswerD = reader["AnswerD"].ToString();
                    oneQuestion.AnswerDBool =
Convert.ToBoolean(reader["AnswerDBool"]);
                    listAllQuestion.Add(oneQuestion);
                  }
                }
                conn.Close();
              }
            }

            if (listAllQuestion.Count == 0) {
              Question noQuestion = new Question();
              noQuestion.QuestionID = 0;
              noQuestion.Message = Names.NoDataNames.NoDataInQuestion;
```

```
            listAllQuestion.Add(noQuestion);
          }
          return listAllQuestion;
        }

        public Question SelectedQuestionByQuestionId(int QuestionId) {
          string SqlString = "SELECT QuestionID, QuestionName, AnswerA,
AnswerABool, AnswerB, AnswerBBool, AnswerC, AnswerCBool, AnswerD, " +
            "AnswerDBool FROM Question Where QuestionID=" +
QuestionId.ToString();

          Question oneQuestion = new Question();
          using (OleDbConnection conn = new OleDbConnection(_ConnString)) {
            using (OleDbCommand cmd = new OleDbCommand(SqlString, conn)) {
              conn.Open();
              using (OleDbDataReader reader = cmd.ExecuteReader()) {
                while (reader.Read()) {
                  oneQuestion.QuestionID =
Convert.ToInt32(reader["QuestionID"].ToString());
                  oneQuestion.QuestionName = reader["QuestionName"].ToString();
                  oneQuestion.AnswerA = reader["AnswerA"].ToString();
                  oneQuestion.AnswerABool =
Convert.ToBoolean(reader["AnswerABool"]);
                  oneQuestion.AnswerB = reader["AnswerB"].ToString();
                  oneQuestion.AnswerBBool =
Convert.ToBoolean(reader["AnswerBBool"]);
                  oneQuestion.AnswerC = reader["AnswerC"].ToString();
                  oneQuestion.AnswerCBool =
Convert.ToBoolean(reader["AnswerCBool"]);
                  oneQuestion.AnswerD = reader["AnswerD"].ToString();
```

```
                oneQuestion.AnswerDBool =
Convert.ToBoolean(reader["AnswerDBool"]);
                }
               }
              }
              conn.Close();
             }
             return oneQuestion;
            }

            public void UpdateQuestion(string QuestionName, string AnswerA, bool
AnswerABool, string AnswerB, bool AnswerBBool,
             string AnswerC, bool AnswerCBool, string AnswerD, bool AnswerDBool, int
QuestionId) {
              string SqlString = "UPDATE Question SET QuestionName=?, AnswerA=?,
AnswerABool=?, AnswerB=?, AnswerBBool=?, AnswerC=?, AnswerCBool=?, " +
           "AnswerD=?, AnswerDBool=? WHERE QuestionID=?";

             using (OleDbConnection conn = new OleDbConnection(_ConnString)) {
              using (OleDbCommand cmd = new OleDbCommand(SqlString, conn)) {
               cmd.CommandType = CommandType.Text;
               cmd.Parameters.AddWithValue("QuestionName", QuestionName);
               cmd.Parameters.AddWithValue("AnswerA", AnswerA);
               cmd.Parameters.AddWithValue("AnswerABool", AnswerABool);
               cmd.Parameters.AddWithValue("AnswerB", AnswerB);
               cmd.Parameters.AddWithValue("AnswerBBool", AnswerBBool);
               cmd.Parameters.AddWithValue("AnswerC", AnswerC);
               cmd.Parameters.AddWithValue("AnswerCBool", AnswerCBool);
               cmd.Parameters.AddWithValue("AnswerD", AnswerD);
               cmd.Parameters.AddWithValue("AnswerDBool", AnswerDBool);
```

```
            cmd.Parameters.AddWithValue("QuestionID", QuestionId);
            conn.Open();
            cmd.ExecuteNonQuery();
            conn.Close();
          }
        }
      }

      public void DeleteQuestionByQuestionId(int QuestionId) {
        string SqlString = "DELETE FROM Question WHERE QuestionID=" +
QuestionId.ToString();
        using (OleDbConnection conn = new OleDbConnection(_ConnString)) {
          using (OleDbCommand cmd = new OleDbCommand(SqlString, conn)) {
            conn.Open();
            cmd.ExecuteNonQuery();
            conn.Close();
          }
        }
      }

    }
  }

  public class Question {
    private int _Number;
    private int _QuestionID;
    private string _QuestionName;
    private string _AnswerA;
    private bool _AnswerABool;
    private string _AnswerB;
```

```
private bool _AnswerBBool;
private string _AnswerC;
private bool _AnswerCBool;
private string _AnswerD;
private bool _AnswerDBool;
private string _Message;

public Question() {
  _Number = 0;
  _QuestionID = 0;
  _QuestionName = String.Empty;
  _AnswerA = String.Empty;
  _AnswerABool = false;
  _AnswerB = String.Empty;
  _AnswerBBool = false;
  _AnswerC = String.Empty;
  _AnswerCBool = false;
  _AnswerD = String.Empty;
  _AnswerDBool = false;
  _Message = String.Empty;
}

public int Number {
  set { _Number = value; }
  get { return _Number; }
}
public int QuestionID {
  set { _QuestionID = value; }
  get { return _QuestionID; }
}
```

```
public string QuestionName {
  set { _QuestionName = value; }
  get { return _QuestionName; }
}
public string AnswerA {
  set { _AnswerA = value; }
  get { return _AnswerA; }
}
public bool AnswerABool {
  set { _AnswerABool = value; }
  get { return _AnswerABool; }
}
public string AnswerB {
  set { _AnswerB = value; }
  get { return _AnswerB; }
}
public bool AnswerBBool {
  set { _AnswerBBool = value; }
  get { return _AnswerBBool; }
}
public string AnswerC {
  set { _AnswerC = value; }
  get { return _AnswerC; }
}
public bool AnswerCBool {
  set { _AnswerCBool = value; }
  get { return _AnswerCBool; }
}
public string AnswerD {
  set { _AnswerD = value; }
```

```
    get { return _AnswerD; }
  }
  public bool AnswerDBool {
    set { _AnswerDBool = value; }
    get { return _AnswerDBool; }
  }
  public string Message {
    set { _Message = value; }
    get { return _Message; }
  }
}
```

ДОДАТОК Б
КОМПАКТ ДИСК