МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально науковий інститут комп'ютерних наук та управління проєктами

Кафедра управління проєктами

«Допущений до захисту»
Завідувач кафедри
_____Чернов С.К.
«12» грудня 2022 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти **«магістр»**

на тему:

Удосконалення механізмів управління функціональною логістикою у сфері реалізації товарів

Виконав: студент 6171м групи
_____ Білявська О.В.
(підпис)
Керівник роботи:
к.т.н., доцент_____
(посада, науковий ступень вчене звання)
_____Чернова Лб.С.
(підпис)

Миколаїв – 2022 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально науковий інститут комп'ютерних наук та управління проєктами

Кафедра управління проєктами
Спеціальність 122 Комп'ютерні науки
Освітня програма «Управління проєктами»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____Чернова Л.С.
(підпис)
«29» вересня 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТА
на здобуття ступеня вищої освіти «магістр»

Студенту ***Білявській Олександрі Валентинівні***
(Прізвище, ім'я, по батькові)

1. Тема роботи: Розвиток моделей календарно-мережевого планування в управлінні проєктами
Керівник роботи ***к.т.н., доцент Чернова Любава Сергіївна***
Затверджені наказом ректора № 805-уч. від «29» вересня 2022 року
2. Термін подання роботи: 12.12.2022р.
3. Вихідні дані по роботі: дослідити та удосконалити механізми управління функціональною логістикою у сфері реалізації товарів.
4. Перелік питань, що належать до розробки (найменування розділів):
Розділ 1. Теоретичні засади по інформаційним технологіям та в управленні функціональною логістикою
Розділ 2. Опис програмного додатку
Розділ 3. Етапи розробки та створення програмного додатку
Розділ 4. Охорона праці
Розділ 5. Охорона навколишнього середовища.
5. Перелік презентаційних матеріалів виконаний в програмі Power Point.
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ 4. Охорона праці	Гурець Н.В., старший викладач	06.09.2022 Н.Гурець	06.09.2022
Розділ 5. Охорона навколишнього середовища	Гурець Н.В., старший викладач	06.09.2022 Н.Гурець	06.09.2022

7. Дата видачі завдання 05.09.2022 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Вивчення літературних джерел з предмету дослідження	протягом навчання на 5 курсі	виконано
2	Складання розгорнутого плану магістерської роботи та ознайомлення керівника з планом кваліфікаційної роботи	Вересень 2022	виконано
3	Написання розділу 1	Вересень 2022	виконано
4	Написання розділу 2	Жовтень 2022	виконано
5	Написання розділу 3	Листопад 2022	виконано
6	Написання розділів з Охорони праці та навколишнього середовища (4,5)	Листопад 2022	виконано
7	Оформлення магістерської роботи	Грудень 2022	виконано
8	Передача магістерської роботи рецензенту для рецензування	Грудень 2022	виконано
9	Передача магістерської роботи науковому керівникові для написання відгуку	Грудень 2022	виконано
10	Попередній захист магістерської роботи	12.12.2022	виконано
11	Захист магістерської роботи	19.12.2022	виконано

Студент _____ (підпис) Білявська Олександра Валентинівна

Керівник роботи _____ (підпис) Чернова Любава Сергіївна

Анотація

Білявська О. В. Удосконалення механізмів управління функціональною логістикою у сфері реалізації товарів. 122 «Комп'ютерні науки» освітньої програми «Управління проектами. Миколаїв. 2022 рік. 132 стор.

Зміст роботи:

У цій роботі ми провели систематичний огляд літератури щодо основних моделей прогнозування потреби швидкопсувних товарів на малих та середніх підприємствах, розроблених протягом останніх років. Аналіз наявної інформації дозволив авторам визначити, що методи прогнозування з використанням м'яких обчислювальних методів і часових рядів є найбільш використовуваними в літературі. Були також визначені основні вхідні змінні цих моделей та фактори, що впливають на зміну вимог.

Ключові слова: моделі прогнозування, попит, товари, швидкопсувні товари, бізнес, торгово-складський облік.

Annotation

O. Biliavska. Improvement of functional logistics management mechanisms in the field of goods sales. 122 "Computer Science" of the educational program "Project Management. Mykolaiv; 2022. 132 pages.

Content of work:

In this work, we conducted a systematic review of the literature on the main models for forecasting the demand for perishable goods in small and medium-sized enterprises developed in recent years. Analysis of the available information allowed the authors to determine that forecasting methods using soft computational methods and time series are the most widely used in the literature. The main input variables of these models and the factors influencing the change in requirements were also determined.

Keywords: forecasting models, demand, goods, perishable goods, business, trade and warehouse accounting.

Зміст

ВСТУП

У багатьох галузях промисловості підприємства стикаються зі все більш невизначеним попитом. Численні фактори викликають це явище; проте головною зміною, що поширюється між різними секторами, є постійно зростаюча увага до клієнтів. Компанії визначили, що клієнти мають вирішальне значення не тільки тому, що вони безпосередньо впливають на успіх конкретних продуктів чи фірм, а й тому, що вони відіграють фундаментальну роль у багатьох внутрішніх процесах. Хоча роль клієнтів у бізнес-процесах глибоко проаналізована, питання прогнозування попиту та роль клієнтів до кінця не вивчені. Це дослідження має на меті вивчити вплив неоднорідності запитів клієнтів на підходи прогнозування попиту на основі трьох випадків дослідження дій. На основі аналізу поведінки споживачів розробляється відповідна методологія для кожного випадку на основі кластеризації клієнтів відповідно до їхніх моделей попиту.

Точні прогнози попиту мають важливе значення для компаній, включаючи виробників і дистриб'юторів, оскільки вони забезпечать більш чуйне обслуговування клієнтів із меншими запасами та зменшенням морального старіння. Залежні від ціни моделі попиту є найбільш поширеними, можливо, тому, що цінова стратегія є найефективнішим інструментом, який використовувався для впливу на попит фірми. При розгляді інтегрованого управління попитом і ланцюгом поставок прогнозування і ціноутворення пов'язані один з одним. Тому, очевидно, слід враховувати вплив його ціни при прогнозуванні попиту споживачів на певний продукт. Відмінною особливістю цієї роботи є те, що вона залежить від описових моделей поведінки споживачів, щоб передбачити попит клієнтів і вивести стратегії ціноутворення в динамічних умовах. Зокрема, під час моделювання нашої функції попиту ми враховуємо як ментальний облік споживачів, так і вплив референтної ціни.

Актуальність обраної теми полягає у необхідності осмислення прогнозування, оскільки зараз, все більше люди перекладають обов'язки на машини, та розробляють програми безпосередньо пов'язані з прогнозуванням.

Сучасні тренди програмування тісно пов'язані з не лише з візуальною складовою, а й з досвідом користування користувачів ресурсу, їх задоволеності зрозумілим та лаконічним дизайном, інтуїтивними налаштуваннями. Все це робить роботу з прогнозуванням важливою, адже потребує врахування поєднання візуальної ідентифікації та програмного забезпечення, використовуючи та проектуючи стильову модель для кожного проекту окремо.

Метою роботи є аналіз структурних ефектів та властивостей на основі розробки власного продукту для прогнозування майбутнього попиту, які можуть бути отримані під час виконання роботи.

Задачі дослідження

Для реалізації мети дослідження були поставлені наступні завдання:

- Аналіз актуальної теоретичної бази з теми прогнозування
- Визначення основних етапів розробки
- Відокремлення відмінностей між розробкою різними методами
- Оцінка моделей розрахунку різними методами
- Розробка власного програмного додатку з урахуванням функціональності та стильового наповнення

Об'єктом дослідження є процес розробки програмного продукту на основі прогнозування.

Предметом дослідження є процес проектування образів та елементів програми.

Серед основних ***методів дослідження*** у даній роботі були використані метод аналізу, синтезу та моделювання. Метод аналізу був використаний щодо наявної інформації та попередніх досліджень. Окрім того, інформація зібрана та проаналізована, в подальшому була синтезована в загальну структуру розуміння моделі. Моделювання власного програми дозволило підтвердити твердження про використання моделі як засобу знаходження майбутнього попиту на товар.

Наукова новизна отриманих результатів полягає в тому, що в результаті проведення дослідження підтверджено залежності структури знаходження майбутнього попиту на товар, за допомогою методів та засобу візуальної ідентифікації ресурсу та позитивного досвіду користування у цільової аудиторії.

Практичне значення отриманих результатів полягає в тому, що в межах даної роботи розроблено різні структури прогнозування попиту, які відрізняються між собою. Окрім того, проаналізовано та систематизована теоретична інформація щодо наявних підходів до розуміння прогнозування. Отримані результати властивостей допоможуть прогнозувати попит на будь-який товар на етапі його проектування та враховувати в подільшому досвід користування ресурсами.

Структура та обсяг роботи

Магістерська робота складається зі вступу, п’яти розділів з висновками, загальних висновків та списку використаної літератури, а також додатків. Основна частина роботи викладена на 94 сторінках друкованого тексту, включає 35 рисунків. Список використаної літератури з 27 найменувань поданий на 2 сторінках. Повний обсяг роботи складає 132 сторінки.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ЗАСАДИ ПО ІНФОРМАЦІЙНИМ ТЕХНОЛОГІЯМ ТА В УПРАВЛІННІ ФУНКЦІОНАЛЬНОЮ ЛОГІСТИКОЮ

1.1 Теоретичні засади

Інформаційні технології (ІТ) зростали та розвивалися за останні 50 років; Ви не можете думати та планувати проект, бізнес чи іншу ініціативу без використання цієї технології. Коли ми говоримо про інформаційні технології, то маємо на увазі не лише персональні комп’ютери чи смартфони, а й сучасне обладнання на фабриках, автомобільній промисловості, авіаційній промисловості, різноманітну побутову техніку тощо, це так чи інакше не тільки полегшило наше повсякденне життя, але й також зменшується вартість та час загалом.

Дослідження показують, що чверть працівників країни, значну частину року працюють вдома, а ще чверть працюють «мобільними» - на ходу. Це відображає великі можливості, які надають інформаційні технології та Інтернет як важливий інструмент для впровадження в організаціях та державних установах. Економісти високо цінують важливість інформаційних технологій для зростання бізнесу, зниження витрат і просування найкращих продуктів.

Протягом останніх років глобалізація та комп’ютеризація змінили промисловість, політику, культуру та соціальний порядок. Глобалізація означає остаточну інтеграцію економічних і культурних інститутів. Ця інтеграція відбувається в результаті використання інформаційних технологій. технологічна революція передбачає глобальні комп’ютеризовані мережі та вільний рух товарів, інформації та людей через національні кордони. Отже, Інтернет та глобальні комп’ютерні мережі роблять можливим глобалізацію, створюючи технологічну інфраструктуру для світової економіки. Комп'ютеризовані мережі, системи супутникового зв'язку, програмне та апаратне забезпечення пов'язують разом і сприяють розвитку світової економіки.

Метою цієї роботи є програма розробки, яка допомагає з прогнозуванням попиту на товар. Для розробки програми ми будемо використовувати мову програмування C++.

Інформаційні технології (IT) зростали та розвивалися за останні 50 років; Ви не можете думати та планувати проект, бізнес чи іншу ініціативу без використання цієї технології. Коли ми говоримо про інформаційні технології, то мають на увазі не лише персональні комп'ютери чи смартфони, а й сучасне обладнання на фабриках, автомобільній промисловості, авіаційній промисловості, різноманітну побутову техніку тощо, це так чи інакше не тільки полегшило наше повсякденне життя, але й також зменшується вартість та час загалом.

Дослідження показують, що чверть працівників країни, значну частину року працюють вдома, а ще чверть працюють «мобільними» - на ходу. Це відображає великі можливості, які надають інформаційні технології та Інтернет як важливий інструмент для впровадження в організаціях та державних установах. Економісти високо цінують важливість інформаційних технологій для зростання бізнесу, зниження витрат і просування найкращих продуктів.

Протягом останніх років глобалізація та комп'ютеризація змінили промисловість, політику, культуру та соціальний порядок. Глобалізація означає остаточну інтеграцію економічних і культурних інститутів. Ця інтеграція відбувається в результаті використання інформаційних технологій. технологічна революція передбачає глобальні комп'ютеризовані мережі та вільний рух товарів, інформації та людей через національні кордони. Отже, Інтернет та глобальні комп'ютерні мережі роблять можливим глобалізацію, створюючи технологічну інфраструктуру для світової економіки. Комп'ютеризовані мережі, системи супутникового зв'язку, програмне та апаратне забезпечення пов'язують разом і сприяють розвитку світової економіки.

Опис процесу діяльності

Предметною областю для якої розробляється автоматизованої інформаційної системи(АІС) в рамках даної магістерської роботи є діяльність оптово-роздрібного підприємства з методом партійного складського обліку. Даний метод дуже широко застосовується в аналітичних програмах, службовців для ведення складського обліку.

Партія - це операція (документ), який є необхідним при оприбуткуванні товару на склад, так як містить важливу для обліку товару інформацію: номер партії, дату приходу, номенклатуру і місце зберігання.

Таким документом може бути прибуткова накладна, рахунок-фактура, акт прийому-передачі. Партійний облік використовується з тією метою, що система, побудована використанням даного виду обліку, надає можливість деталізувати до партій складські залишки, а також аналізувати будь-які операції по списування товарів з урахуванням партій.

Виділяються такі типи партійного обліку:

- FIFO;
- LIFO;
- ручної облік по партіях;
- комбінований облік по партіях.

Методи «FIFO» і «LIFO» є автоматичними, функціонують завдяки розробленому для ведення обліку алгоритму, не вимагають участі користувача. Розрізняються в черговості списання товарів, зазвичай спираючись на дату надходження. У разі ручного обліку, користувачеві необхідно самому вказувати, з якої партії будуть списуватися товари.

При комбінованому методі користувач має можливість налаштовувати автоматичні методи списання, наприклад, вказувати бажану для списання партію.

У розробці АІС підприємства оптово-роздрібної торгівлі буде реалізований метод партійного обліку FIFO. Він є найбільш популярним, так як в умовах постійних складських залишків списує товари з партій, оприбуткованих раніше

інших. Такий метод є не дуже чутливим до порядку введення документів, що дозволяє вводити інформацію заднім числом.

Незаперечні переваги партійного обліку:

- дозволяє бачити таку інформацію, як дата, час,
- постачальник, товар, реальна кількість товару на складі, яка буде актуальна для менеджерів при подальшому плануванні закупівель;
- є можливість вести аналіз обороту і прибутку товарів від
- різних постачальників, що, в свою чергу, може сприяти налагодженню ефективності співпраці та реалізації продукції;
- дозволяє обчислити собівартість списання товарів.

Для виконання даної кваліфікаційної роботи слід провести аналіз предметної області. Основна діяльність співробітників підприємства складається з планування закупівель і продажів, вибору реалізованого асортименту товарів, складання необхідних для купівлі й продажу продукції документів, ведення звітності.

Підприємство має кілька складів поставка продукції здійснюється від декількох фірм. При занесенні товару в номенклатуру, товарний елемент проходить категоризацію і отримує дві цінових характеристики: планова ціна закупівлі і планова ціна реалізації.

Номенклатура містить наступні дані про товар: найменування, категорія, країна виробник, ставка ПДВ, планова ціна покупки і планова ціна реалізації.

1.2. Методи прогнозування управління. Основні етапи розробки

Практично в кожному рішенні, яке вони приймають, сьогодні керівники враховують якийсь прогноз. Обґрунтовані прогнози попитів і тенденцій – це вже не предмети розкоші, а необхідність, якщо менеджери хочуть впоратися з сезонністю, різкими змінами рівня попиту, маневрами конкурентного зниження цін, страйками та великими коливаннями економіки. Прогнозування може допомогти їм впоратися з цими проблемами; але це може допомогти їм більше,

чим більше вони знають про загальні принципи прогнозування, що воно може, а що не може зробити для них зараз, і які методи відповідають їхнім потребам на даний момент. Тут автори намагаються пояснити менеджерам потенціал прогнозування, приділяючи особливу увагу прогнозуванню збуту продукції Corning Glass Works, оскільки вона дозріває протягом життєвого циклу продукту. Також міститься короткий опис методів прогнозування (рис. 1.1).

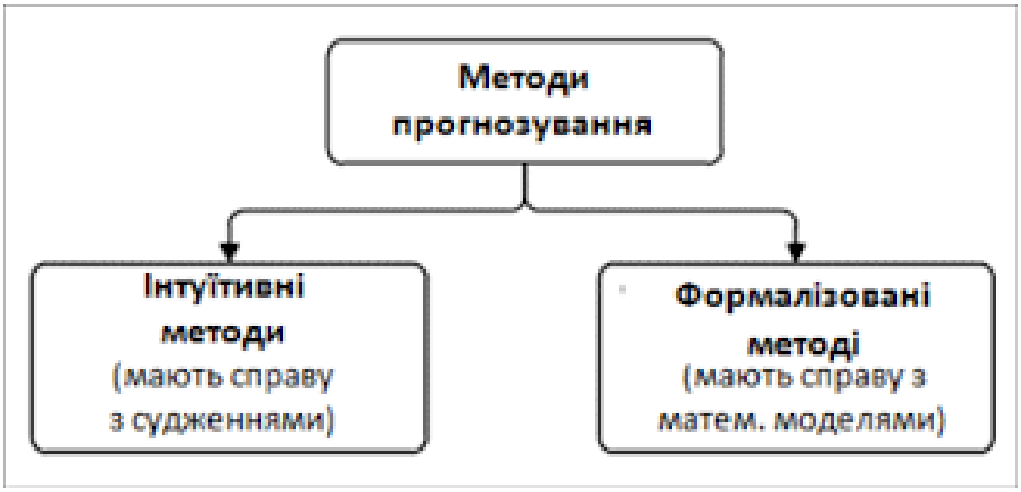


Рис. 1.1 Методи прогнозування

Для вирішення зростаючого різноманіття та складності проблем управлінського прогнозування в останні роки було розроблено багато методик прогнозування. Кожен з них має своє особливе застосування, і потрібно уважно вибирати правильну техніку для конкретного застосування. Менеджер, а також прогнозист відіграють роль у виборі техніки; і чим краще вони розуміють діапазон можливостей прогнозування, тим більша ймовірність, що зусилля компанії з прогнозування принесуть результати.

Вибір методу залежить від багатьох факторів — контексту прогнозу, релевантності та доступності історичних даних, бажаного ступеня точності, періоду часу, який потрібно прогнозувати, вартості/вигоди (або цінності) прогнозу для компанії та час, доступний для проведення аналізу.

Ці фактори необхідно зважувати постійно і на різних рівнях. Загалом, наприклад, синоптик повинен вибрати методику, яка найкраще використовує наявні дані. Якщо прогнозист може легко застосувати одну методику з

прийнятною точністю, він або вона не повинні намагатися «золотою пластиною» використовувати більш досконалу техніку, яка пропонує потенційно більшу точність, але яка вимагає неіснуючої інформації або інформації, отримання якої є дорогим. Такий компроміс відносно легко зробити, але інші, як ми побачимо, вимагають значно більше уваги.

Крім того, якщо компанія бажає зробити прогноз щодо певного продукту, вона повинна враховувати етап життєвого циклу продукту, для якого вона робить прогноз. Доступність даних і можливість встановлення взаємозв'язків між факторами безпосередньо залежать від зрілості продукту, і, отже, етап життєвого циклу є головним визначальним фактором методу прогнозування, який буде використовуватися.

Наша мета — представити огляд цієї галузі, обговоривши, як компанія повинна підійти до проблеми прогнозування, описати доступні методи та пояснити, як узгодити метод із проблемою. Ми проілюструємо використання різних методів з нашого досвіду роботи з ними в Corning, а потім завершимо нашим власним прогнозом щодо майбутнього прогнозування.

Так само різні продукти можуть вимагати різного роду прогнозів. Два продукти CGW, які обробляються по-різному, є основними скляними компонентами для кольорових телевізійних трубок, основним постачальником яких є Corning, і кухонний посуд Corning Ware, власна лінійка споживчих товарів. Ми простежимо методи прогнозування, які використовуються на кожному з чотирьох різних етапів зрілості цих продуктів, щоб дати певне уявлення про вибір і застосування деяких основних методів, доступних сьогодні.

Перш ніж ми розпочнемо, давайте звернемо увагу на те, чим відрізняються ситуації для двох видів продуктів:

Для споживчого продукту, такого як посуд, контроль виробника над розподільним трубопроводом поширюється принаймні через рівень дистриб’ютора. Таким чином, виробник може безпосередньо впливати або контролювати продажі споживачам, а також безпосередньо контролювати деякі елементи трубопроводу.

Таким чином, багато змін у ставках відвантаження та загальної прибутковості пов'язані з діями, вжитими самими виробниками. Тактичні рішення щодо акцій, акцій та цін зазвичай також приймаються на їхній розсуд. Тому методика, обрана прогнозистом для прогнозування продажів, повинна дозволяти включати таку «особливу інформацію». Можливо, доведеться почати з простих технік і перейти до більш складних, які охоплюють такі можливості, але кінцева мета є.

Якщо компанія менеджера постачає компонент OEM, як Corning для виробників труб, компанія не має такого прямого впливу або контролю ні над елементами трубопроводу, ні з кінцевим споживачем. Для компанії може бути неможливим отримати хорошу інформацію про те, що відбувається в точках далі вздовж системи потоку (як у верхньому сегменті Ілюстрації II), і, як наслідок, прогнозист обов'язково буде використовувати інший жанр прогнозування від того, що використовується для споживчого продукту.

Між цими двома прикладами наша дискусія охопить майже весь спектр методів прогнозування. Однак у міру необхідності ми торкнемося інших продуктів та інших методів прогнозування.

Розробка продукту

На ранніх етапах розробки продукту менеджер хоче отримати відповіді на такі запитання:

- Які є альтернативні можливості для зростання, ніж продукт X?
- Як попрацювали відомі продукти, подібні до X?
- Чи варто вступати в цей бізнес; і якщо так, то в яких сегментах?
- Як ми повинні розподілити зусилля та кошти на дослідження та розробки?
- Наскільки успішними будуть різні концепції продукту?
- Як продукт X впишеться на ринки через п'ять чи десять років?

Прогнози, які допомагають відповісти на ці довгострокові питання, обов'язково мають самі довгі горизонти.

Загальне заперечення проти довгострокового прогнозування полягає в тому, що практично неможливо передбачити з точністю, що станеться через кілька років у майбутньому. Ми згодні з тим, що невизначеність збільшується, коли прогноз робиться на період більше двох років. Проте, щонайменше, прогноз і міра його точності дозволяють менеджеру знати ризики при здійсненні обраної стратегії і на цих знаннях вибрати відповідну стратегію з наявних.

Систематичні дослідження ринку, безумовно, є основою в цій сфері. Наприклад, аналіз пріоритетних моделей може описати переваги споживачів і ймовірність того, що вони куплять продукт, і, таким чином, має велике значення для прогнозування (і оновлення) рівнів і показників проникнення. Але є й інші інструменти, залежно від стану ринку та концепції продукту.

1.3. Ознайомлення з середовищами для розробки програмного додатку

Оскільки C++ одна із найпоширеніших мов програмування, йому існує велика кількість інтегрованих середовищ розробки з різним функціоналом. Оскільки висока продуктивність є однією з основних вимог до розробки програми.

Важливим критерієм вибору середовища розробки програми є наявність профільника. Як об'єкти дослідження були обрані інтегровані середовища розробки MS Visual Studio, JetBrains Clion, Qt Creator, NetBeans та Eclipse SDK , як найбільш функціональні та поширені.

а) MS Visual Studio

Microsoft Visual Studio - це інтегроване середовище, призначене для розробки програмного забезпечення, що має низку додаткових інструментів.

Продукти MS Visual Studio (рис. 1.2) дозволяють розробляти консольні програми, програми з графічним інтерфейсом, веб-сайти, веб-програми та веб-сервіси для платформ.

Рис. 1.2 –Microsoft Visual Studio

MS Visual Studio містить вбудований інструмент для редагування вихідного коду програми та дозволяє проводити рефакторинг коду. Відладчик, вбудований у MS Visual Studio, має два рівні роботи: рівень налагодження вихідного коду та рівень налагодження машинного коду. Додаткові вбудовані інструменти включають форми GUI та зручний редактор для створення програм з інтерфейсами, містять веб-редактор та редактор класів. Це середовище розробки підтримує можливість створення та підключення плагінів (надбудов), що використовуються для розширення функціональності. MS Visual Studio підтримує системи контролю версій вихідного коду (наприклад, Visual SourceSafe або Subversion), містить безліч інструментів, що дозволяють редагувати та проектувати код предметно-орієнтованими мовами програмування.

б) JetBrains Clion

JetBrains CLion — інтелектуальне інтегроване середовище розробки програмного забезпечення, призначене для розробки додатків мовами C і C++ [28]. JetBrains Clion (рис. 1.3) дозволяє розробляти програмне забезпечення на платформах Windows, OS X та Linux.

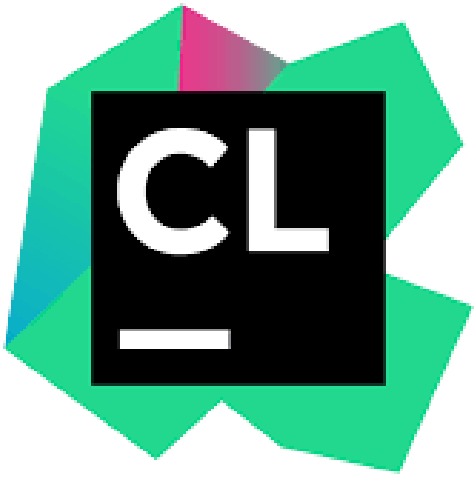


Рис. 1.3 - JetBrains CLion

Багатофункціональне середовище розробки - JetBrains CLion включає вбудований налагоджувач, безліч шаблонів коду та інтерфейс для популярних систем управління версіями (таких як Subversion, Git, GitHub, Mercurial, CVS, Perforce та TFS). CLion надає можливості автодоповнення та автоформатування коду, виконує аналіз коду на стрічці з виділенням потенційних проблем та пропонує способи їх усунення, підтримує систему складання для кросплатформових проектів CMake, а також різні рефакторинги коду. JetBrains CLion має великий репозиторій модулів (доповнень) для розширення існуючої функціональності.

в) Creator Qt

Qt Creator (рис. 1.4) – це повністю інтегроване середовище розробки програмного забезпечення, що містить інструменти для проектування та розробки складних програм у різних операційних системах.

Рис. 1.4 - Qt Creator

Qt Creator призначений для розробки програмного забезпечення мовою програмування C++. Також є PyQt для програмування на Python та PHP-Qt для PHP.

Qt Creator інтегрований з кросплатформовими системами автоматизації складання: qmake і CMake, містить редактор коду Qt Designer, що дозволяє проектувати і створювати графічні інтерфейси користувача (GUI) з віджетів Qt. Це середовище містить багато корисних інструментів, таких як підтримка систем контролю версій (Subversion, Git, GitHub, Mercurial та CVS) та емулятор Qt. Qt Creator містить внутрішній налагоджувач для налагодження звичайних програм C++, а також включає можливість підключення мобільних пристроїв до вашого комп'ютера та налагодження програм, що працюють на них.

г) NetBeans

IDE NetBeans – це безкоштовне середовище IDE з відкритим вихідним кодом для розробників програмного забезпечення [30]. NetBeans(рис. 1.5) надає безліч різних інструментів для створення професійного програмного забезпечення мовами Java, C/C++, PHP, Python, JavaScriot, Groovy, Ruby та інших.

Рис. 1.5 - IDE NetBeans

Середовище розробки NetBeans включає такі функції:

– рефакторинг коду;
– профільування;
– виділення кольором різних синтаксичних конструкцій;
– автоматичне завершення при вході до різних споруд;
– велика кількість шаблонів коду.

У середовищі IDE NetBeans спочатку має бути J2EE SDK або потрібна версія Sun JDK.

NetBeans надає засоби управління системами контролю версій з готовими інтеграціями для Subversion, Mercurial і Git. Редактори середовища IDE та функція перетягування також дозволяють швидко та ефективно розробляти графічні інтерфейси програм.

г) Eclipse SDK

Eclipse SDK – це безкоштовне інтегроване середовище розробки програмного забезпечення з відкритим кодом. Eclipse SDK (рис. 1.6) має велику і розгалужену систему модулів (надбудов), що підключаються, яка забезпечує робоче середовище з такими мовами програмування, як C, C++, PHP, Perl, Python, Ruby, Ada або COBOL.

Рис. 1.6 - Eclipse SDK

Eclipse SDK дозволяє створювати кросплатформне програмне забезпечення для

Microsoft Windows, Mac OS X, дистрибутиви Linux та навіть Solaris. Це середовище включає засоби розробки Eclipse Java (JDT), які містять компілятор Java. У програмі використовуються інструменти, що входять до системи Eclipse Rich Client Platform.

Така система допомагає розробникам створювати потужні функціональні користувацькі програми з гарним графічним інтерфейсом на основі CSS (Cascading Style Sheets – каскадні таблиці стилів). Інтегроване середовище розробки

Eclipse SDK надає Java IDE з усіма інструментами, необхідні для розробки якісних програм.

Через війну вивчення розглянутих середовищ розробки програмування мовою C++ було створено порівняльна таблиця 1.1.

Параметри	MS Visual Studio	JetBrains Clion	Qt Creator	Eclipse SDK	NetBeans
Редактор користувальницького інтерфейсу	+	-	+	+	+
Вбудований профілювальник	+	-	-	-	-
Вбудований відгадчик	+	+	+	+	+

Підтримка C++11	+	+	+	+	+
Встановлене ПЗ на підприємстві, наявність ліцензії	+	-	+	-	-
Опис використання	+	-	+	+	-
Сумісність з колишніми проектами	+	-	+	-	-

Для розробки програми вибрали середовище розробки Microsoft Visual Studio. Середовище розробки Microsoft Visual Studio було обрано тому, що він має ряд необхідних для розробки функцій, таких як підтримка профілювання коду мовою програмування C++, налагодження та сумісність з успадкованими проектами.

Ознайомлення з сучасними системами програмування

1995 рік був особливо цікавим роком у світі комп'ютерного програмування. Саме в цьому році було випущено чотири нові мови програмування, які вплинули на світове співтовариство програмування таким чином, що не передбачалося на момент їх офіційного оголошення. До речі, це був також час, коли Інтернет тільки починав робити хвилі, а Інтернет був на межі вибуху в основну цифрову культуру.

Чотири мови, які дебютували в 1995 році, були Java, JavaScript, PHP і Ruby. Незважаючи на те, що їхні відповідні релізи не супроводжували особливої фанфари, ці мови з часом виростуть і стануть повсюдними інструментами програмування для більшості розробників програмного забезпечення. До цього часу домінуючими мовами були C і C++. Незважаючи на те, що ці двоє були дуже потужними, вони за своєю суттю не підходили для всесвітньої мережі. Крім того, для початківців програмістів (особливо C++) вони часто вважалися дещо складними та залякуючими.

Серед чотирьох, Java виявилася шаленим успіхом. З його часто цитованим гаслом «Напишіть один раз, запустіть будь-де» Java стала миттєвим хітом,

оскільки її було набагато легше вивчити та опанувати (порівняно з C++). Java також представила нову ідею віртуальної машини (JVM), яка дозволила писати програми, які запускалися б на різних платформах без необхідності перекомпіляції.

В останні роки JavaScript витіснив Java як провідну мову програмування у світі. Постійне зростання JavaScript значною мірою пов'язано з впровадженням Node.js, технології, яка зробила можливим запуск JavaScript на стороні сервера.

PHP став домінуючою силою у сфері веб-програмування, особливо в поєднанні з іншими популярними технологіями з відкритим вихідним кодом, такими як Linux, Apache та MySQL. Разом вони утворили те, що зазвичай називають стеком LAMP.

Ruby здобув популярність серед веб-розробників після випуску шалено популярного веб-фреймворку Ruby on Rails у 2005 році датським програмістом Девідом Хайнемайєром Ханссоном (DHH).

Для великих (або командних) проектів в середу розробки повинні бути включені файловий менеджер, інтегроване середовище розробки програмного забезпечення, PlSql (використовується і для роботи з Системою Управління БД і як інструмент звітів), Cristal Reports (створення звітів), StarTeam (ведення журналу версій продукту, що розробляється).

Підводячи підсумки потрібно сказати про те, що інтегровані середовища розробки ПО дозволяють програмістам скоротити час на написання додатків, знизити затратність на написання (розробку), підвищити зручність розробки - що і є однією з основних цілей програмної інженерії.

Інтегровані середовища розробки зручні для командних проектів, остільки оскільки в таких середовищах можна виробляти весь цикл створення програмного забезпечення.

Інтегровані середовища зручні в написанні програм.

Структура інформаційної системи торгово-складського обліку

При виконанні роботи необхідно розробити АІС підприємства оптово-роздрібної торгівлі, які відповідають наступним вимогам, пред'являти до інформаційних систем:

1) функціональність:

- точність;
- захищеність;
- здатність до взаємодії;

2) продуктивність:

- ефективність користування ресурсами;

3) надійність:

- відмовостійкість;
- здатність до відновлення;

4) зручність використання:

- Зрозумілість;
- зручність навчання;
- зручність роботи;

5) переносимість:

- зручність установки;

6) зручність супроводу:

- зручність перевірки;
- зручність внесення змін.

В ході реалізації своєї виробничої і господарської діяльності більшість підприємств і організацій стикаються з необхідністю пошуку найбільш оптимальних способів зберігання різноманітних видів товарних і матеріальних цінностей. Збереження товарно-матеріальних цінностей кожного підприємства може бути забезпечена шляхом організації спеціально обладнаних складів, або комор на базі приміщень даного підприємства.

В рамках виробничої діяльності існують спеціальні складські виробничі приміщення, вони виступають в якості певної частини складського приміщення,

основним призначенням якого є здійснення процедур типу приймання товару, сортування та зберігання, процеси комплектацій, процедури відпуску та відвантаження товарно-матеріальних цінностей.

Підсобні складські приміщення використовуються з метою розміщення в них підсобних господарств і реалізації заходів, пов'язаних з необхідністю обслуговування основних функцій технологічного процесу.

Підсобні складські приміщення - виступають в якості спеціальних приміщень, основним призначенням яких є зберігання упаковки, вузлів технологічного обладнання та механізмів, інвентарю, тари, спеціальних збиральних апаратів, і пакувальних відходів.

Процедурі контролю над рухом і забезпеченням збереженості товарних і матеріальних запасів на підприємстві реалізується за допомогою використання методів складання та підтвердження:

- схематичного плану реалізації документообігу на підприємстві. У плані регламентуються основні форми документів, які можуть бути використані при обліку матеріальних запасів, в даному випадку дозволяється використання форм документів розроблених силами підприємства, регламентуються посадові особи, яким делеговані обов'язки за належне забезпечення руху матеріальних запасів на кожній стадії виробничого циклу також за проходження всіх етапів супроводжуючих документів, вказуються терміни подання документів до служби бухгалтерії, наводяться також зразки підписів відповідальних осіб. У складі плану схеми вказується також обраний на підприємстві спосіб здійснення обліку товарно-матеріальних цінностей в складських службах, він може бути кількісним або якісним-сумовим;

Процедури передачі товарної продукції покупцям реалізуються з оформленням пакету спеціальних супровідних документів, які регламентовані у вимогах, присутніх в договорах поставки та перевезення товарної продукції. Сюди слід віднести накладні, ТТН, ж/д накладні, рахунки або рахунки-фактури.

Товарна продукція, яку підприємство торгівлі закуповує в цілях здійснення наступного перепродажу, може поставлятися на склад цього підприємства, а також торгове підприємство може набирати і її за межами власного складу.

Накладна містить в собі такі реквізити:

- дата виписки товару і номер документа;
- повне найменування постачальника і покупця;
- повне і короткий опис товарної продукції;
- обсяги та кількість товарної продукції;
- ціна за одиницю товарної продукції сукупна вартість усього товару відпускаються зі складу (з урахуванням ПДВ), ПДВ потрібно відображати в документі окремим рядком.

Складський облік бере свій початок з моменту здійснення процедури приймання матеріалів і компонентів від постачальників підприємства і його підрядників. Спочатку перевіряються представлені супровідні документи, які поставляються одночасно зі вступниками на підприємство матеріалами і цінностями. Потім здійснюється процедура перевірки матеріалів шляхом огляду матеріалів і елементів від постачальника, реалізується також процедура перевірки супровідної документації, якщо здійснюється повернення цінностей і матеріалів від клієнта. Дані заносяться ручним способом. Матеріали передаються на склад тільки після того як ретельно буде перевірена вся необхідна інформація.

Потім товари і цінності відправляються на зберігання. У всіх прийнятих на зберігання матеріалів і елементів відповідальний співробітник сканує штрих-коди, і вносить кількість просканувати матеріалу в спеціальний термінал, після чого дані вносяться в програму.

У відповідності до сплаченого рахунком, що надійшли від клієнта, відповідальний співробітник реалізує процедуру відвантаження йому необхідних елементів, і на підставі реалізації даної процедури оформляють рахунок. При надходженні на склад заявки на повернення цінностей, реалізується процедура їх приймання та перевірки з метою перевірки відповідності всім необхідним

умовам здійснення процедури повернення, здійснюється формування даних по поверненню, і відповідно до них формується акт списання цінностей, а також рахунок для здійснення їх оплати.

Були проведені відповідні аналізи предметної сфери, на підставі яких зроблено розробка функціональної моделі, в ній наводиться детальний опис реалізованої в даний час діяльності співробітників складського обліку, що відповідають за облік матеріалів і компонентів.

Огляд наявних алгоритмів

При плануванні виробництва та розробці маркетингової стратегії на коротко- та середньостроковий періоди першими міркуваннями менеджера зазвичай є точна оцінка поточного рівня продажів і точна оцінка швидкості, з якою цей рівень змінюється (рис. 1.7).

Рис. 1.7.- Алгоритм прогнозування

Таким чином, на цьому етапі прогнозист покликаний зробити два пов’язаних внески:

Для Corning Ware, де рівні системи розподілу організовані відносно просто, ми використовуємо статистичні методи для прогнозування відвантажень і

інформацію про місця для прогнозування змін у ставках відвантажень. Зараз ми знаходимося в процесі включення спеціальної інформації — маркетингових стратегій, економічних прогнозів тощо — безпосередньо в прогнози відвантажень. Це веде нас у напрямку причинно-наслідкової моделі прогнозування.

З іншого боку, постачальник компонентів може бути в змозі спрогнозувати загальний обсяг продажів з достатньою точністю для планування виробництва з широким навантаженням, але середовище конвеєра може бути настільки складним, що найкращий засіб для короткострокових прогнозів – покладатися насамперед на оцінки продавців. . Ми вважаємо це вірним, наприклад, при оцінці попиту на скло для телевізора за розміром і покупцем. У таких випадках найкращою роллю статистичних методів є надання посібників і перевірок для прогнозів продавців.

Загалом, однак, на цьому етапі життєвого циклу доступно достатньо даних часових рядів і достатньо причинно-наслідкових зв’язків відомо з безпосереднього досвіду та ринкових досліджень, щоб прогнозист дійсно міг застосувати ці два потужні набори інструментів. Повинні бути доступні історичні дані принаймні за останні кілька років. Синоптик так чи інакше використає все це.

На цьому етапі можна згадати загальну критику. Люди часто заперечують проти використання кількох останніх даних (наприклад, даних про продажі в найближчому минулому) для побудови прогнозів, оскільки, за їхніми словами, поточна ситуація завжди настільки динамічна, а умови змінюються настільки радикально та швидко, що історичні дані з більш глибокої давнини мають незначну цінність або не мають жодної цінності.

На практиці, як ми виявили, загальні закономірності мають тенденцію зберігатися щонайменше протягом одного-двох кварталів у майбутнє, навіть якщо особливі умови викликають коливання продажів протягом одного-двох (місячних) періодів у найближчому майбутньому.

Для короткострокового прогнозування на один-три місяці вперед вплив таких факторів, як загальна економічна ситуація, мінімальний і не викликає радикальних зрушень у структурі попиту. І оскільки тенденції мають тенденцію змінюватися поступово, а не раптово, статистичні та інші кількісні методи відмінно підходять для короткострокового прогнозування. Використання однієї або лише кількох найновіших точок даних призведе до недостатнього врахування природи тенденцій, циклів та сезонних коливань продажів.

Сортування тенденцій та сезонів.

Очевидно, що тенденція і сезонність – це дві абсолютно різні речі, і їх потрібно розглядати окремо в прогнозуванні.

Поміркуйте, що станеться, наприклад, якби прогнозист просто взяв середнє значення останніх точок даних вздовж кривої, об’єднав це з іншими подібними середніми точками, що тягнуться назад у безпосереднє минуле, і використав їх як основу для проекція. Синоптик може легко реагувати на випадкові зміни, приймаючи їх за ознаки переважаючої тенденції, приймаючи зміну темпів зростання за сезонні тощо.

Щоб уникнути саме такого роду помилок, техніка ковзної середньої, яка подібна до щойно описаної гіпотетичної, використовує точки даних таким чином, що вплив сезонних факторів (і нерівностей) усувається.

Крім того, керівникові потрібні точні оцінки тенденцій і точні оцінки сезонності, щоб планувати виробництво з широким навантаженням, визначити маркетингові зусилля та розподіл, а також підтримувати належні запаси, тобто запаси, які відповідають попиту споживачів, але не є надмірно дорогими.

Таким чином, мета методики прогнозування, яка використовується тут, полягає в тому, щоб якнайкраще визначити тенденції та сезонність. На жаль, більшість методів прогнозування проектуються за допомогою процесу згладжування, аналогічного методу ковзної середньої або подібної до гіпотетичної методики, яку ми описали на початку цього розділу, і точніше відокремлення тенденцій і сезонних факторів потребуватиме додаткових зусиль і витрат.

І все-таки розбірні підходи зарекомендували себе на практиці. Ми можемо найкраще пояснити причини їхнього успіху, приблизно окресливши спосіб створення прогнозу продажів на основі тенденцій, сезонних факторів та даних, отриманих з них.

У особливих випадках, коли немає сезонних факторів, які слід враховувати, звичайно, цей процес значно спрощується, і може бути достатнім менше даних і простіші методи.

Ми виявили, що аналіз закономірностей зміни темпів зростання дає нам більшу точність у прогнозуванні поворотних моментів (і, отже, змін від позитивного до негативного зростання, і навпаки), ніж коли ми використовуємо лише цикл тренда.

Основна перевага розгляду змін зростання насправді полягає в тому, що часто можна передбачити раніше, коли відбудеться ситуація без зростання. Таким чином, графік зміни зростання забезпечує чудову візуальну базу для прогнозування та визначення моменту повороту.

Висновки до розділу 1

Логістика – один із нових наукових напрямків у теорії та практиці маркетингу, що характеризується раціональною організацією взаємодії постачання, виробництва, розподілу, транспортування та споживання готової продукції. Він розглядає вищевказані функції з точки зору системи. Слід зазначити, що логістичний підхід управління логістикою підприємства сприяє максимальній оптимізації всієї логістичної операції.

Логістичні витрати нерозривно пов’язані з функціонуванням логістичної системи підприємства і формуються в різних сферах: постачання, виробництва та розподілу, що ускладнює можливість ефективного управління ними.

Логістична вартість – це результат логістичної операції, який становить значну частку загальних витрат підприємства, визначає вартість продукції та

послуг праці, визначає кінцевий результат виробничої діяльності, а потім визначає загальну економічну ефективність підприємства. .

Можна встановити, що логістична вартість – це грошовий вираз сукупності матеріальних, трудових, фінансових та інформаційних ресурсів, спожитих підприємством щодо забезпечення бізнес-процесів та операцій з руху матеріальних потоків у логістичній системі.

Управління логістичними витратами – це процес цілеспрямованого формування оптимального рівня логістичних витрат підприємства. Основним принципом управління логістичними витратами є концепція загальної (загальної) вартості.

На основі існуючого визначення та на основі власних досліджень пропонується більш конкретне пояснення суті цього поняття для чіткого розуміння та практичного використання: управління витратами на логістику полягає у прийнятті логістичних рішень на основі отриманих облікових даних. процес. Керуйте всім набором витрат на логістику, інформаційний потік і потік капіталу в усій логістичній системі, щоб зменшити витрати на логістичну діяльність.

В цьому розділі ми ознайомилися з мовами програмування, з середовищами програмування. З усіх мов та середовищ, ми обрали мову C++ та середовище Microsoft Visual Studio 2019.

Дізналися про методи прогнозування та про основні етапи розробки.

- Метод експертних оцінок.
- Метод екстраполяції.
- Методи моделювання.
- Метод економічного прогнозування (економічний аналіз)
- Балансовий метод.
- Нормативний метод
- Програмно-цільовий метод (ПЦМ).

Для виконання даної кваліфікаційної роботи слід провести аналіз предметної області. Основна діяльність співробітників підприємства складається

з планування закупівель і продажів, вибору реалізованого асортименту товарів, складання необхідних для купівлі й продажу продукції документів, ведення звітності.

Були проведені відповідні аналізи предметної сфери, на підставі яких зроблено розробка функціональної моделі, в ній наводиться детальний опис реалізованої в даний час діяльності співробітників складського обліку, що відповідають за облік матеріалів і компонентів.

РОЗДІЛ 2. ОПИС ПРОГРАМНОГО ДОДАТКУ

2.1. Опис технічного завдання

Розробити програму, яка реалізує систему товарно-складського обліку з функцією прогнозування попиту на товар. Реалізувати прогнозування різними методами. Розробити можливість взаємодії з користувачем. Надати можливість вибору введення даних вручну, автоматично або зчитуванням із файлу. Також додати можливість зберігати отриманні данні до файлу.

Оцінка ефективність ланцюга поставок у впровадженні пакетів програмного забезпечення.

Щоб оцінити прийняті логістичні рішення, необхідно проаналізувати ефективність бізнес-процесів у ланцюзі поставок.

Для досягнення цілей логістики необхідно вимірювати результати управління ланцюгами поставок. Такі показники є кількісними і показують ступінь ефективності логістичних операцій і функціональної реалізації.

Вимірювання результатів логістичної діяльності залежить від:

- спеціалізовані бізнес-процеси;
- завдання управління Контроль тривалості виконання логістичних операцій.

Збалансована система показників складається з п’яти основних груп показників:

- тривалість логістичного циклу;
- ефективність інвестицій;
- продуктивність людей та інфраструктури;
- задоволеності клієнтів;
- логістичні витрати.

Щоб оцінити ефективність запропонованих заходів, необхідно визначити покращення показників часу. Використання робочого паралелізму, щоб скоротити час на виконання підготовки, буде:

– 60 днів генерального планування та проектування (дизайнерське агентство);
– 10 днів на складання плану завантаження та оформлення маршруту доставки;
– 30 днів - на планування виробництва та документацію;
– 25 днів - розробити план монтажних робіт;
– 10 днів - підготувати комерційну пропозицію;
– 8 днів - Підписати договір з партнером.

Разом: Термін виконання буде скорочено на 143 дні, скорочення на 20,6%, позитивний результат. Це показує, що організація також може збільшити свій дохід на 20,6% порівняно з існуючою системою управління, коли попит на її продукцію стабільний протягом того ж періоду, тобто збільшення продажів становить:

$$ПРОр = Орф \cdot ПО \qquad (2.1)$$

де ПРОр - приріст продажів, тис. грн.,

Орф - фактичний обсяг продажів, тис. грн.,

ПО – розрахунковий відсоток зростання доходу.

ПРОр = 120176 х 20,6% = 24756 тис. грн.

Також необхідно розрахувати скорочення витрат на запропоновану діяльність. Це дозволить знизити витрати на оплату праці в контексті кооперацийних угод з проектними та транспортними організаціями та виробниками:

– Проектно-технічне обслуговування – 80 000 грн./міс.;
– Логістичне обслуговування - 50 000 грн./міс.;
– Відділ матеріально-технічного забезпечення – 100 000 грн./міс.

Разом: 230 000 грн./міс або 2760 000 грн./рік.

Проте запропоновані заходи також передбачають певні витрати.

Постійні витрати включають:

– відділ АСУ - 140 000 грн./міс.;

– єдина Логістична Служба – 270 000 грн./міс.

Разом 410 000 грн./міс

Інвестиції складатимуть:

– Ліцензійна програма T-FLEX – 4800 000 грн.;
– Придбання комп'ютерної техніки – 2400 грн./міс.

Таблиця 2.1 – Вихідні дані, використані для розрахунку ефективності впроваджених заходів

Показник	Умовне позначення	Значення
Обсяг реалізації у 2022 році, тис. грн	Орф	120176
Приріст обсягу реалізації, тис. грн	ПРОр	24756
Середньооблікова чисельність працюючих на підприємстві у 2022 році, чол.	Ч	180
Середньорічне вироблення 1 працюючого у 2022 році, тис. грн.	Врф	667,64
Середньорічна заробітна плата одного працюючого у 2022 році, тис. грн.	З/Пгод	324
Відсоток відрахувань соціального податку, %	СОТЧ	31,5%
Умовно-постійні витрати у 2022 році, грн.	РУ-П1	78997
Витрати на захід, грн.: - поточні; - капітальні	ЗТЕК ЗКАП	4920 (410*12) 7200

Розрахуємо ефективність виконання подій, зазначених у таблиці 2.2 (впровадження програмних продуктів на всіх ланках ланцюга поставок).

Таблиця 2.2 - Розрахунок ефективності події

№ п/п	Показник	Методика розрахунку	Розрахунок показника
1	Запланований обсяг реалізації, тис.грн	$Op_{п} = O_{р_ф} + ПР_{ор}$	$Op_{п} = 120176 + 24756 = 144932$
2	Заплановане вироблення 1-го працюючого, тис. грн.	$Вр_{п} = Op_{п} / Ч$	$Вр_{п} = 144932/180 = 805{,}18$
3	Приріст продуктивності праці по підприємству, %	$ПТ \uparrow = (Вр_{п} - Вр_{ф}) / / Вр_{ф} * 100$	$ПТ = (805.18 - 667.64) / / 667.64 * 100 = 20.60$
4	Вихідна чисельність працюючих, чол.	$Ч_{іск} = Ч_{отч} * I_{ПТ}$	$Ч_{іск} = 180 * (1 + 0.206) = = 217.08$
5	Умовна економія чисельності працюючих, чол.	$Еч = (Ч_{іск} * ПТ \uparrow) / /(100 + ПТ \uparrow)$	$Еч = (217.08 * 20.6) / /(100 + 20.6) = 37.08$
6	Умовно-річна економія із заробітної плати, тис. грн.	$E_{зп} = E_{ч} * З / П_{год}$	$E_{зп} = 37.08 * 324 = 12013.92$
7	Умовно-річна економія щодо відрахувань на соціальні потреби, тис. грн.	$E_{СН} = E_{ЗП} * C_{отч} / 100$	$E_{СН} = 37.08 * 324 = 12013.92$
8	Умовно-річна економія щодо умовно-постійних витрат, тис. грн.	$E_{у-п} = (P_{у-п_1} / Орф - P_{у-п_2} / Орп) * Орп$	$E_{у-п} = (78998/120176 - 83917/144932) * 144932 = (0.66 - 0.58) * 144932 = 11594.56$
9	Умовно-річна економія, тис. грн	$E_{у-г} = E_{ЗП} + E_{СН} + + E_{у-п} - З_{ТЕК}$	$E_{у-г} = 12013.92 + 3784.38 + 11594.56 - 4920 = 22472.86$
10	Річний економічний	$E_{г} = E_{у-г-Ен} * Зкап$	$E_{г} = 22472.86 - 0.2 * 7200 = 21032.86$

	ефект, тис. грн.		
11	Термін окупності, років	$Т_{ОК} = З_{КАП} / Е_{У-Г}$	$Т_{ОК} = 7200 / 22472.86 = 0.32$

З аналізу таблиці видно, що заходи результативні, річна економія після виконання умовних заходів 22,47286 млн грн., річний економічний ефект 21,03286 млн грн., продуктивність праці зросла на 20,6%, обсяг продажів збільшився на 24 756 грн., а термін окупності діяльності менше одного року, тобто 3,84 мес.

Амортизаційні відрахування визначаються виходячи з норми амортизації.

Норма амортизації визначається виходячи з економічно доцільного строку корисного використання і має компенсувати зношеність основних фондів при можливому психічному та фізичному зносі та створити економічну основу для заміни [11].

Амортизаційні відрахування за кожну годину роботи системи розраховуються за формулою (2.1).

$$А_{ч} = Ф_{перв} / F_{д} \qquad (2.1)$$

Де $Ф_{перв}$ початкова вартість системи або окремих елементів;

$F_{д}$ - Річний фонд годин роботи (2500 годин)

Таблиця 2.3 - Розрахунок амортизаційних

п/п	Елемент КТС	$Ф_{перв}$	$F_{д}$	$А_{ч}$	Кількість годин	Загальна вартість (грн.)
1	Aspire	24999	2500	1,9999	398	795,9602
Загалом						795,9602

Підсумовуючи, вартість придбання, обслуговування та експлуатації програмного та апаратного забезпечення, яке ми отримали, становить 26 415,14 грн.

2.2. Опис середовища та системи для управління функціональною логістикою

Стале управління ланцюгом поставок — це «управління сировиною та послугами від постачальників до виробника / постачальника послуг до клієнта і назад із чітко врахованим покращенням соціального та екологічного впливу».

Зменшення бар'єрів у торгівлі та вдосконалення технологій дозволили підприємствам і ланцюгам поставок розширюватися між регіонами та країнами, ще більше посилюючи потребу в сталому управлінні ланцюгами поставок. Крім того, для бізнесу недостатньо просувати стійкість лише в межах своєї компанії, цілі ланцюги поставок мають управлятися стійким способом, щоб бізнес залишався конкурентоспроможним. Переваги сталого управління ланцюгом поставок включають підвищення привабливості бренду, лояльності збуту та задоволення зацікавлених сторін, а також зменшення негативного впливу на суспільство та навколишнє середовище. Наявність стійкого ланцюга поставок також покращить прозорість та видимість уздовж ланцюга, що дозволить компаніям швидко реагувати на зміни на ринку та інші ситуації.

Одним з ключових компонентів ланцюга поставок є складське господарство. Більшість складських і транспортних компаній мало звертають увагу на вплив своїх дій на навколишнє середовище і не розуміють соціальних наслідків своєї діяльності. Ці компанії розглядають такі фактори, як економічність та задоволеність клієнтів, як основні показники ефективності. Лінтон і Куарігуасі Фрота Нето стверджують, що, будучи залученими до зберігання та транспортування товарів, ці компанії повинні усвідомити важливість трансформації своєї поточної бізнес-моделі на стійку. Додатковим стимулом для аргументації є те, що викиди транспортних засобів є одним з основних джерел забруднення.

Це дослідження реалізує реальний проект сталого управління складом в Окленді для сертифікованої ISO складської компанії. Компанія забезпечує складські приміщення для хімічних та харчових продуктів в Окленді та

Крайстчерчі для різних клієнтів і доставляє товари різним виробничим і роздрібним підприємствам по всій Новій Зеландії. Різні малі та великі транспортні компанії також працюють у співпраці з компанією в процесі транспортування та доставки. Складська компанія стрімко розвивається, тому хоче збільшити потужність, створюючи більше складів. Процеси управління операцією компанії чітко визначені, але через нестачу кваліфікованої робочої сили, останні зміни в галузі охорони праці та екологічних норм, а також тиск з боку свідомих членів ланцюга поставок, клієнтів та кінцевих користувачів, компанія вирішили перепроектувати бізнес-процеси для нових складів, які дотримуються підходу до управління стійкістю та досягають конкурентних переваг за короткий період.

Основні проблеми, такі як утримання співробітників, дотримання вимог і вплив на навколишнє середовище, а також фінансова віддача, пов’язані з розташуванням складу. Тому вибір місця розташування складів є важливим стратегічним питанням, яке матиме великий вплив на економічний, екологічний та соціальний аспекти всього управління бізнесом. Деякі з цих проблем ми висвітлюємо в іншій частині цього розділу.

Для реалізації поставленої задачі був створений окремий клас, в якому зберігається вся інформація.

```
class Products
{
public:
  Products();

  void Create(int count_tov, int count_day, int num);
  void Print();
  void Method1();
  void Method2();
  void Method3();
  void save();
```

```
  ~Products();

private:
  vector<vector<int>> arr;
  vector<string> product_name;
  vector<double> prognoz;
};
```

В цьому класі і зберігається вся інформація стосовно складу.

2.3 Засоби розробки та опис архітектури програмного забезпечення

Написання безпечного коду, що забезпечує безпеку без уразливостей, є критичним викликом для кібербезпеки. Написання коду без уразливостей вже давно принаймні так само складно, як написання коду без помилок. Хоча в програмному забезпеченні є багато інших потенційних джерел ризику безпеки, розробка коду без відомих класів уразливостей завжди здавалася посильною метою. Він покладається на людей-розробників, які використовують інструменти, методи та процеси для створення програмного забезпечення, яке не має особливо відомих типів дефектів.

Один з найефективніших підходів — дослідження мов та інструментів програмування — приніс технології, які, як показано, протистоять категоріям уразливостей, в основному, не допускаючи їх. Безпечні мови пам'яті, які керують виділенням та звільненням пам'яті, замість того, щоб вимагати від програміста це робити, унеможливлюють для розробників створення вразливостей переповнення буфера та деяких інших типів впливу через відсутні перевірки меж масиву, використання нульового покажчика та даних. витік через повторне використання пам'яті. Потокобезпечні мови можуть вирішувати випадки, коли

умови гонки можуть бути використані, щоб підірвати перевірки, пов’язані з безпекою в програмі.

У межах спільноти розробників програмного забезпечення групи та організації, які мають на меті безпечну розробку програмного забезпечення, включили інструменти та методи у свій життєвий цикл розробки програмного забезпечення, щоб включити життєвий цикл безпечної розробки. Раннє програмне забезпечення з високою надійністю використовувало формальні методи для визначення властивостей безпеки системи, а також перегляд коду, щоб використовувати людину для пошуку таких недоліків на рівні кодування.2 Корпорація Майкрософт створила свій життєвий цикл розвитку безпеки, додавши аналіз першопричин, навчання безпеки, моделювання загроз, конкретні вимоги до безпечного кодування та тестування безпеки, яке включало тестування на проникнення та нечіткое тестування. Практика, як правило, впроваджується на основі потреб бізнесу, відчутного впливу на безпеку та відповідає усталеній або розвивається практиці розвитку.

Необхідні дослідження, які впливають на те, що працює, а що може працювати для безпечного розвитку. Здається, що нинішні дослідження відіграють, на жаль, обмежену роль у створенні, пропонуванні, оцінці та підтвердженні інструментів, методів і процесів, які використовуються на практиці для безпечного розвитку. Зокрема, дослідження рідко стосуються безпосередньо інструментів і методів, оскільки вони використовуються, у контексті, в якому вони використовуються. Нам потрібні додаткові дослідження ефективності та результатів безпечних інструментів, методів і процесів розробки. Це дослідження можна судити про його вплив на те, як розробка програмного забезпечення працює на практиці. Властивості дослідження впливають на ймовірність такого впливу.

Суворість дослідницького наукового експериментування вимагає ряду вимог до процесу, включаючи формулювання гіпотези, що перевіряється, контроль змінних експерименту, щоб переконатися, що експеримент фактично перевіряє гіпотезу, і аналіз експериментальних даних і результатів для

математичного підтвердження гіпотези (або спростувати нульову гіпотезу). Хоча ці процеси можуть стати основою важливих фундаментальних досліджень безпечної розробки, вони часто уникають безладних реалій, пов'язаних із впровадженням техніки на практиці, саме тому, що ці безладні реалії ускладнюють проектування експериментів.

Негативні результати дослідження, які не підтверджують, що безпечна техніка розробки підвищує безпеку, хоча й важливі для галузі досліджень, навряд чи вплинуть на безпечний розвиток на практиці. Перший урок розробника безпеки у великій технологічній компанії полягав у тому, що вказувати розробникам не робити чогось майже завжди було неефективним, якщо це не було поєднане з альтернативою, яку вони могли б використати для досягнення мети застарілої практики. Крім того, виявити, що інструмент або техніка експериментально неефективні для забезпечення безпеки, не доводить, що вони неефективні за межами контрольованого експерименту, у більш широкому, безладному та різноманітнішому контексті розробки програмного забезпечення.

Які з тих речей, які робляться в дослідженні, сподіваються на практичне перенесення в безпечний розвиток? Дві сучасні тенденції досліджень безпеки дають певну надію на безпечний розвиток. Одна з них полягає в тому, що безпечна розробка стала темою на конференціях з дослідження безпеки, охоплюючи такі теми, як оцінка здатності розробників безпечно та належним чином використовувати криптовалюту, оцінка інструментів, які допомагають розробникам уникати вразливостей, і вимірювання здатності розробників кодувати функціональні можливості, що стосуються безпеки. .

Інша обнадійлива тенденція — оцінка артефактів. Багато розробок програмного забезпечення базується на існуючому програмному забезпеченні, використовуючи фреймворки, бібліотеки та відкритий код. Пропонування артефакту, який використовується для встановлення та підтвердження ідеї дослідження, зменшує бар'єри для передачі цієї ідеї в розробку програмного забезпечення. Доступ до коду з відкритим кодом з умовами ліцензії, зручними

для повторного використання, може збільшити його потенціал для використання. Деякі дослідницькі стимули змінюються, щоб заохочувати подання артефактів у рамках процесу подання та публікації наукової роботи на конференціях з безпеки, таких як USENIX Security і ACSAC.

Вимоги до технічного забезпечення

Технічна підтримка системи має максимально ефективно використовувати існуючі технічні засоби.

До складу комплексу повинні входити такі технічні засоби:

1) сервер бази даних;

2) персональні комп'ютери (ПК) користувачів.

Мінімальні вимоги до характеристик апаратних компонентів, за яких значення тимчасових параметрів Системи повинні відповідати вимогам, представленим у ТЗ:

для сервера бази даних:

- процесор - 2 x IntelXeon3 ГГц;
- обсяг оперативної пам'яті – 16 ГБ;
- дискова підсистема – 4 x 146 ГБ;
- дисковод компакт-дисків (DVD-ROM);
- мережевий адаптер - 100 Мбіт/с.

для ПК користувача:

- процесор - Intel Pentium 1,5 ГГц;
- обсяг оперативної пам'яті – 256 МБ;
- дискова пам'ять – 40 ГБ;
- мережевий адаптер - 100 Мбіт/с.

Опис архітектури програмного забезпечення

Архітектура програмного забезпечення дає пояснення того, як ваші системи поводяться на структурному рівні. Системи, які ви використовуєте, мають набір компонентів, розроблених для виконання певного завдання або набору завдань. Архітектура програмного забезпечення забезпечує фундамент, на якому все

програмне забезпечення, яке є у компанії, можна змінити, створити або вилучити з експлуатації.

Архітектура програмного забезпечення впливає на якість, продуктивність, обслуговування та успіх системи на основі дизайну. Не розглядаючи архітектуру програмного забезпечення на регулярній основі, компанія відкриває себе для довгострокових наслідків, і проблеми, які можуть поставити їх системи під загрозу поломки, злому або низької продуктивності.

У сучасних системах існують загальні шаблони в архітектурі програмного забезпечення, які називаються архітектурними системами для програмного забезпечення. У більшості випадків для створення цілісної системи використовується кілька різних архітектурних систем, особливо для систем, які створювалися роками або працювали, або тих, які були побудовані різними розробниками.

Опис класів

Клас у C++ є будівельним блоком, який веде до об'єктно-орієнтованого програмування. Це визначений користувачем тип даних, який містить власні члени даних і функції-члени, до яких можна отримати доступ і використовувати, створивши екземпляр цього класу. Клас C++ схожий на план для об'єкта.

Наприклад: розглянемо клас автомобілів. Може бути багато автомобілів з різними назвами та марками, але всі вони будуть мати деякі спільні властивості, як-от усі вони матимуть 4 колеса, обмеження швидкості, діапазон пробігу тощо. Отже, автомобіль – це клас і колеса, обмеження швидкості, пробіг. їх властивості.

Клас — це визначений користувачем тип даних, який має члени даних і функції-члени.

Члени даних — це змінні даних, а функції-члени — це функції, які використовуються для маніпулювання цими змінними, і разом ці члени даних і функції-члени визначають властивості та поведінку об’єктів у класі.

На прикладі класу Car членом даних буде обмеження швидкості, пробіг тощо, а функціями члена можуть бути гальмування, збільшення швидкості тощо.

Об'єкт - це екземпляр класу. Коли клас визначено, пам’ять не виділяється, але коли він створюється (тобто створюється об’єкт), пам’ять виділяється.

Визначення класу та оголошення об'єктів

Клас визначається в C++ за допомогою ключового слова class, за яким слідує ім'я класу. Тіло класу визначається всередині фігурних дужок і закінчується крапкою з комою в кінці.

Оголошення об'єктів: коли визначено клас, визначається лише специфікація об'єкта; пам'ять або сховище не виділено. Для використання даних і функцій доступу, визначених у класі, потрібно створити об’єкти.

Синтаксис:

ClassName ObjectName;

Доступ до членів даних і функцій-членів: До членів даних і функцій-членів класу можна отримати доступ за допомогою оператора dot(‘.’) з об’єктом. Наприклад, якщо ім’я об’єкта – obj, і ви хочете отримати доступ до функції-члена з ім’ям printName(), тоді вам доведеться написати obj.printName() .

Доступ до членів даних

Доступ до загальнодоступних членів даних також здійснюється таким же чином, але об’єкт не може мати прямий доступ до членів приватних даних. Доступ до члена даних залежить виключно від контролю доступу цього члена даних.

Цей контроль доступу надається модифікаторами доступу в C++. Існує три модифікатори доступу: відкритий, приватний і захищений.

Клас - це шаблон або план, який зв'язує властивості та функції сутності. Ви можете помістити всі сутності або об’єкти, що мають подібні атрибути, під одним дахом, відомим як клас. Класи далі реалізують основні концепції, такі як інкапсуляція, приховування даних та абстракція. У C++ клас діє як тип даних, який може мати кілька об’єктів або екземплярів типу класу.

Розглянемо в якості прикладу створений нами клас Products:

```
class Products
{
public:
  Products();

  void Create(int count_tov, int count_day, int num);
  void Print();
  void Method1();
  void Method2();
  void Method3();
  void save();

  ~Products();

private:
  vector<vector<int>> arr;
  vector<string> product_name;
  vector<double> prognoz;
};
```

В ньому є конструктор, деструктор. У private частині зберігаються змінні. Також в ньому є декілька функцій які зберігаються у public секції.

Опис методів

Існує багато методів прогнозування

Рис. 2.1 – прогнозування різними методами

У роботі було використано чотири метода прогнозування

1. Балансовий метод
2. Екстраполяційний метод
3. Нормативний метод
4. Трігг-лінча

Далі про кожен метод окремо:

Балансовий метод.

У світовій практиці сформувалися дві методології планування та прогнозування економічного та соціального розвитку. Перша з них заснована на марксистській теорії розширеного відтворення, друга – на кейнсіанській, монетаристській та інших теоріях. За першою методологією планування ґрунтувалося на командно-адміністративній системі управління. Друга методологія є основою планування та прогнозування у країнах з ринковою

економікою. У зв'язку з переходом колишніх соціалістичних країн до ринкових відносин формується єдина методологія.

Методика планування та прогнозування розвитку економіки визначає основні засади, підходи та методи проведення прогнозних та планових розрахунків, виявляє та характеризує логіку формування прогнозів, планів та їх реалізації.

Принципи - це основні правила планування та прогнозування, тобто вихідні положення формування прогнозів та обґрунтування планів з точки зору їх цілеспрямованості, послідовності, структури, логіки та організації розвитку. Іншими словами, це основні вимоги, яких необхідно дотримуватись при розробці прогнозів та планів.

Методи – це методи, прийоми, використовувані розробки прогнозів, планів, програм. Вони є інструментом реалізації методологічних принципів планування і прогнозування.

Логіка - упорядкована послідовність дій при проведенні прогнозних розрахунків та обґрунтуванні запланованих рішень.

Методологія є складовою методології. Вона має приватний характері і підпорядкована методиці. Методологія являє собою сукупність конкретних методів та прийомів, що використовуються для проведення конкретних прогнозних чи планових розрахунків.

Науковою основою методології планування та прогнозування є закони у суспільному розвиткові та економічна теорія. Прогностичні функції виконують закони діалектики: закон єдності та боротьби протилежностей, закон взаємопереходу кількісних та якісних змін, закон заперечення заперечення.

У перехідний період до ринкових відносин зростає роль прогностичних балансів, розроблених на макрорівні: платіжного балансу, балансу доходів та витрат держави, зведеного балансу трудових ресурсів, балансу попиту та пропозиції. Результати балансових розрахунків є основою для формування структурної, соціальної, фінансово-бюджетної та грошово-кредитної політики, а також політики зайнятості та зовнішньоекономічної діяльності. Баланс також

використовується для виявлення диспропорцій у поточному періоді, відкриття невикористаних резервів та обґрунтування нових пропорцій.

За допомогою балансового методу реалізується принцип збалансованості та пропорційності. Він використовується при розробці прогнозів, планів та програм. Суть її полягає у ув'язці потреб країни у різних видах продукції, матеріальних, трудових та фінансових ресурсах з можливостями виробництва та джерелами ресурсів.

Балансовий метод передбачає розробку балансів, що являють собою систему показників, в якій одна частина, що характеризує ресурси за джерелами надходження, дорівнює іншій, що показує розподіл (використання) у всіх напрямках їх витрачання.

Система балансів, що використовується при плануванні та прогнозуванні, включає матеріальний, трудовий та фінансовий баланси. Кожна з цих груп включає низку балансів.

Суть балансового методу (підходу) полягає у виявленні та кількісній оцінці відносин між сторонами будь-якої діяльності, які врівноважують одна одну.

Економічні баланси поділяються на:

- матеріальні, службовці встановлення матеріальних і матеріальних показателей. Вони розробляються у відповідних фізичних одиницях (га, тонни, м, од.) для найважливіших видів продукції; виробничі потужності, основні фонди, фонди споживання та накопичення;

- вартісні (фінансові) - у яких економічні величини виражені у вартісної (грошової) формі, що полегшує зіставлення різноманітних ресурсів. Вони застосовуються розробки державних документів: державного бюджету, економічної оцінки виробничих планів, грошових потоків населення;

- для забезпечення процесу виробництва робочої сили, розвитку невиробничої сфери складаються баланси праці, що відображають наявність ресурсів, їх склад та розподіл;

- натурально-вартісні, де всі розрахунки ведуться паралельно у натуральному та вартісному вираженні.

Баланси диференціюються за: призначення та використання продукції - балансу засобів виробництва та балансу товарів народного споживання; період дії - оперативний, середньостроковий та довгостроковий; за охопленням об'єкта - міжгалузеві, галузеві, територіальні, місцеві; за одиницями виміру - на натуральні, вартісні, натурально-вартісні; за видами використання моделей - одно- та багатотоварні.

Екстраполяційний метод

Методи екстраполяції тренду, мабуть, найпоширеніші та найбільш розроблені серед усієї сукупності методів прогнозування. Використання екстраполяції в прогнозуванні засноване на припущенні, що процес зміни аналізованої змінної є поєднанням двох складових - регулярної і випадкової:

$$y(x) = f(\bar{a}, x) + n(x)$$

Вважається, що регулярна складова f(a, x) є гладкою функцією аргументу (в більшості випадків часу), що описується кінцевим вектором параметрів a, що зберігають свої значення протягом періоду випередження прогнозу. Цю складову ще називають трендом, рівнем, який визначається основою процесу, трендом. Під усіма цими термінами ховається інтуїтивне уявлення про якусь очищену від перешкод сутність аналізованого процесу. Інтуїтивно, тому що для більшості економічних, технічних, природних процесів неможливо однозначно відокремити тренд від випадкової складової. Все залежить від того, яку мету переслідує цей поділ і з якою точністю воно здійснюється.

Випадкова складова n(x) зазвичай розглядається як некорельований випадковий процес із нульовим середнім. Його оцінки необхідні подальшого визначення точності характеристик прогнозу.

Методи екстраполяційного прогнозування зосереджені на виявленні найкращого опису тренда у певному сенсі та на визначенні прогнозних значень

шляхом його екстраполяції. Методи екстраполяції багато в чому перетинаються з методами прогнозування, що базуються на регресійних моделях. Іноді їх відмінності зводяться лише до розбіжностей у термінології, позначення або написання формул. Проте сама прогностична екстраполяція має низку специфічних особливостей та прийомів, що дозволяють віднести її до певного самостійного типу методів прогнозування.

До особливостей прогнозної екстраполяції відносяться методи попередньої обробки числового ряду з метою приведення його до зручного для прогнозування виду, а також аналіз логіки та фізики прогнозованого процесу, що істотно впливає як на вибір вид екстраполюючої функції та визначення меж зміни її параметрів.

Нормативній метод

Основне завдання, що стоїть перед аналітиками та проектувальниками великих систем, взагалі, як правило, полягає у пошуку найбільш оптимальних шляхів створення більш ефективних систем – як новостворених, так і модернізованих. Складність розв'язання цього завдання полягає, насамперед, у цьому, що зазвичай не вдається знайти рішення суто математичними методами, оскільки, зазвичай, не вдається точно визначити величини (функціонали), які підлягають оптимізації (екстремалізація) в математичному сенсі. Це пов'язано не лише зі складністю опису функціонування великих систем, але й з такими фундаментальними типами, як, наприклад, конкретні цілі, для яких призначена система. По-перше, у системи може бути не одна мета, а їхня сукупність, що відразу призводить до проблеми векторної оптимізації. По-друге, набір цілей, поставлених перед системою, може містити чисто якісні цілі, що не піддаються практичним кількісним вимірам. Це призводить, з одного боку, до проблеми оцінки ступеня досягнення якісної мети, а з іншого боку, до проблеми виміру важливості якісних та кількісних цілей та ступеня їх досягнення.

Аналогічна ситуація виникає і в оцінці наслідків запропонованого способу досягнення мети. Вкажемо, наприклад, що ці наслідки можуть одночасно мати економічний, політичний, соціальний чи інший характер.

У умовах рішення системної проблеми перебуває з допомогою нормативних методів з допомогою дуже складного математичного апарату і полягає у видачі обґрунтованих рекомендацій, достатніх розробки рішення.

Метод нормативного прогнозування – це метод отримання та спеціалізованої обробки прогнозних оцінок об'єкта шляхом систематичного опитування висококваліфікованих спеціалістів (експертів) у вузькій галузі науки, техніки чи виробництва. Прогнозні експертні оцінки відображають індивідуальне судження спеціаліста щодо перспектив розвитку своєї галузі та засновані на мобілізації професійного досвіду та інтуїції.

Метод нормативного прогнозування аналогічний методу Дельфі, методу колективної генерації ідей та методу колективного рецензування у тому сенсі, що одним з його елементів є збір та обробка експертних суджень, висловлених на основі професійного досвіду та інтуїції. Однак від цих методів він відрізняється більшою ясністю теоретичних основ, прийомів формування анкет та таблиць, порядку роботи з експертами та алгоритму обробки отриманої інформації. Цей метод називається евристичним у зв'язку з однорідністю форм розумової діяльності експерта при вирішенні наукового завдання та оцінки перспектив розвитку об'єкта прогнозування, а також у зв'язку з використанням експертами конкретних прийомів, які призвести до правдоподібних висновків.

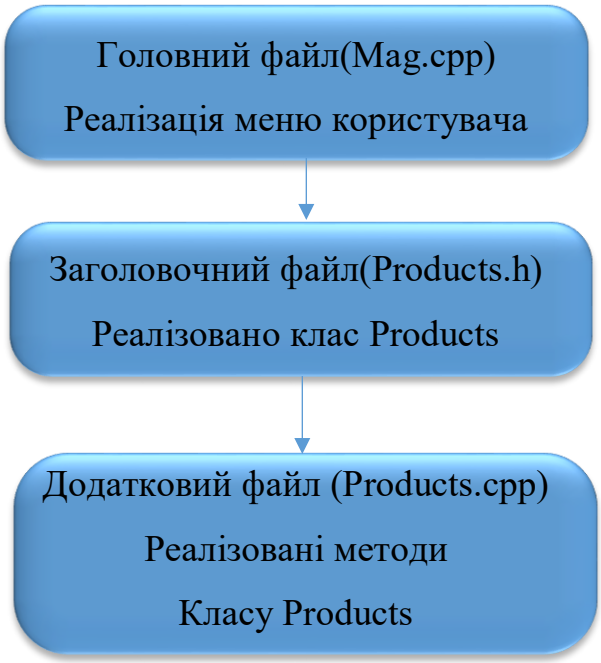
Метод Трігг-Лінча

Модель Тригга-Лінча відноситься до моделей з адаптивними параметрами адаптації, тобто модель з підвищеною здатністю до самонавчання.

Тригг-Лінча запропонували модифікувати прогностичні системи за допомогою експоненційного згладжування шляхом зміни швидкості реакції залежно від величини керуючого сигналу. У найпростішій моделі це

еквівалентно настроюванню параметра згладжування α. Найочевидніший спосіб змусити систему автоматично реагувати на розбіжності між прогнозами та фактичними даними — збільшити α, щоб надати більшу вагу свіжим даним і таким чином дозволити моделі швидше адаптуватися до нової ситуації. Як тільки система адаптується, необхідно знову зменшити значення α, щоб відфільтрувати шум.

Діаграма компонентів



Висновки до розділу 2

У другому розділі розповідається про технічне завдання. Також описані середовище та системи які потрібні для створення власного складу. Розповідається про засоби розробки та які необхідні вимоги до технічного забезпечення.

У процесі дослідження системи управління логістичними витратами слід зазначити, що формування логістичної собівартості є невід'ємною частиною загальної логістичної системи підприємства. У роботі проведено внутрішній аналіз факторів, що формують собівартість логістичної діяльності. Зокрема, до

внутрішніх факторів, що формують собівартість логістичної діяльності, відносяться такі підсистеми: обслуговування клієнтів; управління матеріальними потоками; транспортування вантажів; складування; управління запасами.

Також розповідається про класи які використовувалися у роботі та методи якими користувалися для прогнозування товарів. Реалізована діаграма компонентів програми.

Основні проблеми, такі як утримання співробітників, дотримання вимог і вплив на навколишнє середовище, а також фінансова віддача, пов’язані з розташуванням складу. Тому вибір місця розташування складів є важливим стратегічним питанням, яке матиме великий вплив на економічний, екологічний та соціальний аспекти всього управління бізнесом. Деякі з цих проблем ми висвітлюємо в іншій частині цього розділу.

В роботі були використані чотири метода прогнозування

- Балансовий метод
- Екстраполяційний метод
- Нормативний метод
- Трігг-лінча

РОЗДІЛ 3. ЕТАПИ РОЗРОБКИ ТА СТВОРЕННЯ ПРОГРАМНОГО ДОДАТКУ

3.1 Опис та розробка продукту

Тестування та введення

Перш ніж продукт зможе увійти в свою (сподіваюся) стадію швидкого проникнення, необхідно випробувати ринковий потенціал і запровадити продукт, і тоді може бути доцільно провести додаткове тестування ринку. На цьому етапі керівництво потребує відповідей на такі запитання:

- Яким має бути наш маркетинговий план — на які ринки ми повинні вийти і з якими обсягами виробництва?
- Скільки виробничих потужностей знадобиться на ранніх етапах виробництва?
- У міру зростання попиту де ми повинні будувати цю потужність?
- Як ми будемо розподіляти наші науково-дослідні ресурси з часом?

Іноді, звичайно, можна бути впевненим, що новий продукт буде з ентузіазмом прийнятий. Ринкові тести та початкова реакція клієнтів дали зрозуміти, що для посуду Corning Ware буде великий ринок. Оскільки система розподілу вже існувала, час, необхідний для швидкого зростання лінії, в першу чергу залежав від нашої здатності її виготовити. Іноді прогнозування полягає лише в розрахунку потенціалу компанії, але не зазвичай.

Швидке зростання

Коли продукт переходить на цю стадію, найважливіші рішення стосуються розширення потужностей. Ці рішення зазвичай передбачають найбільші витрати в циклі (за винятком основних рішень НДДКР), і відповідні зусилля щодо прогнозування та відстеження є виправданими.

Прогнозування та відстеження на цьому етапі повинні надавати керівнику три види даних:

Тверда перевірка прогнозу швидкого зростання, зробленого раніше.

Важка дата, коли продажі досягнуть «нормального», стабільного зростання.

Для компонентних продуктів відхилення кривої зростання, яке може бути викликано характерними умовами вздовж трубопроводу, наприклад, закупорками запасів.

Прогнозування темпів зростання

Середньо- та довгострокове прогнозування темпів зростання ринку та досягнення стабільних продажів вимагає тих самих заходів, що й етап впровадження продукту,— детальні маркетингові дослідження (особливо опитування щодо намірів купити) та порівняння продуктів.

З іншого боку, коли продукт починає швидко розвиватися, зазвичай є достатньо даних для побудови статистичних і, можливо, навіть причинно-наслідкових моделей зростання (хоча остання обов'язково міститиме припущення, які потрібно перевірити пізніше).

Ми оцінили темпи зростання та стабільну швидкість кольорового ТБ за грубою економетричною моделлю маркетингу на основі даних, доступних на початку цього етапу. Ми також часто проводили маркетингові дослідження.

Темпи зростання посуду Corning Ware, як ми пояснили, були обмежені насамперед нашими виробничими можливостями; і, отже, основною інформацією, яку слід передбачити в цьому випадку, була дата зростання рівня. Оскільки значні запаси буферизували інформацію про споживчі продажі по всій лінії, хороших даних на місцях бракувало, що ускладнювало оцінку цієї дати. Зрештою ми визнали необхідним створити кращу (більш пряму) польову інформаційну систему.

Окрім простого буферування інформації, у випадку компонентного продукту конвеєр справляє певний спотворюючий вплив на попит виробника; ці ефекти, хоча й дуже важливі, часто нелогічно нехтують під час планування виробництва чи потужності.

Стійкий стан

Рішення керівника на цьому етапі значно відрізняються від тих, які приймалися раніше. Більшість планів об'єктів було зведено в квадрат, а тенденції та темпи зростання стали досить стабільними. Можливо, коливання

попиту та прибутку відбудуться через зміну економічних умов, появу нових і конкурентоспроможних продуктів, динаміку конвеєрів тощо, і менеджеру доведеться підтримувати заходи відстеження і навіть вводити нові. Однак, за великим рахунком, менеджер концентрує увагу на прогнозуванні на таких напрямках:

Довго- та короткострокове планування виробництва.

Встановлення стандартів для перевірки ефективності маркетингових стратегій.

Прогнози, розроблені для сприяння плануванню прибутку.

Менеджеру також знадобиться хороша система відстеження та попередження, щоб визначити, що попит на продукт значно скорочується (але, сподіваюся, до цього ще далеко).

Безумовно, менеджеру потрібні прогнози маржі та прибутку, а також довгострокові прогнози, які допомагають планувати на корпоративному рівні. Однак коротко- та середньострокові прогнози продажів є основними для цих більш детальних заходів, і ми зосередимося на прогнозах продажів.

Напишемо блок схему до нашої програми

Розробка власного продукту

Розробка структури даних

Структура даних – це спосіб збору та організації даних таким чином, щоб ми могли ефективно виконувати операції з цими даними. Структури даних – це відтворення елементів даних у термінах певного зв’язку для кращої організації та зберігання.

Ми можемо організувати ці дані у вигляді запису на зразок Запису гравця, у якому буде вказано ім’я гравця та вік. Тепер ми можемо збирати та зберігати записи гравця у файлі чи базі даних як структуру даних.

Наприклад:

Ім’я	Прізвище	Вік	Поштовий індекс
Марк	Мартин	43	12345
Андрій	Мироненко	32	97453
Марина	Вернадська	56	64829

Чи

Код товару	Колір	Ціна
1	Жовтий	15
2	Червоний	13
3	Синій	11

Структура даних —спосіб організації та зберігання даних, щоб ефективно виконувати операції. Доступ, вставка, видалення, пошук і сортування даних є одними з основних операцій, які можна виконувати за допомогою структур даних. Не всі структури даних можуть ефективно виконувати ці операції, це призвело до розробки різних структур даних. Скажімо, вам потрібно знайти конкретну книгу в неорганізованій бібліотеці, це завдання займе величезну кількість часу. Подібно до того, як бібліотека організовує свої книги, нам потрібно організувати наші дані так, щоб операції можна було виконувати ефективно.

Для створення структури даних використовується попередньо визначений тип даних, наприклад Integer, Stings, Boolean. Використання типу даних залежить від вимог користувача та типу даних, які вони хочуть зберігати. Тепер давайте зануримося в структури даних, які використовуються в нашому повсякденному програмуванні, а також розглянемо підтримку структур даних для різних мов програмування.

Одновимірні масиви

Базовою структурою даних, яку використовують у повсякденному програмуванні, є масив. Масив може містити фіксовану кількість контейнерів для зберігання даних, і над цими даними можна виконувати операції відповідно до потреб користувача.

Багатовимірні масиви

Багатовимірний масив — це просто розширення звичайного масиву, де ми визначаємо масив із рядком контейнерів. У випадку багатовимірного масиву ми маємо рядки, а також стовпці. Щоб отримати доступ до індексу, нам потрібні два числа.

Динамічні масиви

Більшість мов програмування дозволяють визначати динамічні масиви, що означає, що пам’ять виділяється під час виконання програми. Розглянемо це таким чином, ви визначили масив з 10 індексами, але вам потрібно лише 3 індекси, отже, решта 7 індексів знищені, що споживає додаткову пам’ять. Динамічні масиви дають можливість розподілити пам'ять відповідно до вимог програми.

Підтримка масивів для різних мов програмування:

Java: Є кілька класів для масивів, але найвідомішим є клас ArrayList.

C++: - Вектори використовуються для представлення динамічних масивів у C++. Його можна реалізувати за допомогою std::vector.

Python: Python має новий тип даних, відомий як список, як і Boolean і Integers. Тип даних списку можна використовувати для динамічного визначення масивів.

Однозв’язаний список

Пов’язаний список — це набір вузлів, які з’єднані за допомогою посилань. Пов’язаний список містить вузол, який зберігає елементи даних і адресу наступного вузла. Перший вузол зазвичай називають головним вузлом, а останній — хвостовим. Покажчик головного вузла вказує на наступний вузол, а вказівник хвостового вузла вказує на Null.

Динаміка цієї структури даних полегшує додавання або видалення вузлів. Щоб додати/видалити вузол, вам просто потрібно відстежити попередній вузол і вузол після нього і відповідно налаштувати покажчики.

Двозв’язаний список

Двозв’язаний список мало чим відрізняється від однозв’язаного списку, єдине, що їх відрізняє, — це вказівник на попередній вузол. Зображення нижче може дати вам коротке уявлення про те, як повинна виглядати структура.

У випадку двозв’язаного списку попередній вказівник головного вузла вказує на Null, а наступний покажчик хвостового — на Null. Попередній покажчик полегшує перехід в будь-якому напрямку. Отже, додавання та видалення вузлів стає дуже простим, все, що вам потрібно зробити, це відстежувати попередній і наступний вузли та відповідно налаштовувати вказівник.

Круговий зв’язаний список

Круговий зв’язаний список – це зв’язаний список, де всі вузли з’єднані, щоб утворити коло. Головного чи хвостового вузла немає. Перевага такого типу зв’язаного списку полягає в тому, що будь-який вузол може бути використаний як відправна точка.

Порівняння між масивами та зв'язаним списком

Список — це відмінний від масиву тип структури даних. Замість того, щоб зберігати дані в шматках пам'яті, масив потребує безперервного простору пам'яті. Якщо додати елемент до масиву, це може змінити розташування пам’яті всього масиву, але дані залишаться неушкодженими. Масив дає вам номер індексу, отже, ви можете отримати до нього прямий або послідовний доступ. Тоді як для доступу до будь-якого члена даних потрібно пройти через весь список.

3.2. Написання програмного продукту

Для початку роботи програми були підключені наступні бібліотеки

```
#include <vector>
```

```
#include <string>
#include <fstream>
#include <sstream>
#include <algorithm>
#include <cmath>
#include <msclr\marshal_cppstd.h>
```

Бібліотека iostream була підключена для відображення інформації у консолі.

Бібліотека vector була підключена для зберігання інформації масивів даних.

Також було використано простір імен std

```
using namespace std;
```

Для зчитування даних із файл було написано наступну функцію

```
vector<vector<string>> getDataFromCSV(string fname) {
	vector<vector<string>> values;
	ifstream ifs(fname.c_str());
	if (!ifs)
	{

		return values;
	}
	string line;
	int count = 0;
	while (ifs.good())
	{
		getline(ifs, line);
		vector<string> temp;
		count++;
		string s = "";
		for (int i = 0; i < line.size(); ++i)
		{
			if (line[i] != ';')
				s += line[i];
			else
			{
				temp.push_back(s);
				s = "";
			}
		}
		if (s != "")
			temp.push_back(s);
		if (temp.size() != 0)
			values.push_back(temp);
	}
	ifs.close();

	return values;
 }
```

Та при натисканні на кнопку відбувалося завантаження данних у масив

```
    private:   System::Void   button1_Click(System::Object^   sender,
System::EventArgs^ e) {
          string                          str                          =
msclr::interop::marshal_as<std::string>(textBox1->Text);
          product = getDataFromCSV(str);
          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
```

Для того, щоб додати товар була написана наступна функція

```
    private:   System::Void   button2_Click(System::Object^   sender,
System::EventArgs^ e) {
          string                          type                         =
msclr::interop::marshal_as<std::string>(comboBox1->SelectedItem-
>ToString());
          string                          name                         =
msclr::interop::marshal_as<std::string>(textBox2->Text);
          string                           m1                          =
msclr::interop::marshal_as<std::string>(textBox3->Text);
          string                           m2                          =
msclr::interop::marshal_as<std::string>(textBox4->Text);
          string                           m3                          =
msclr::interop::marshal_as<std::string>(textBox5->Text);
          string                           m4                          =
msclr::interop::marshal_as<std::string>(textBox6->Text);
          string                           m5                          =
msclr::interop::marshal_as<std::string>(textBox7->Text);
          string                           m6                          =
msclr::interop::marshal_as<std::string>(textBox8->Text);
          string                           m7                          =
msclr::interop::marshal_as<std::string>(textBox9->Text);
          string                           m8                          =
msclr::interop::marshal_as<std::string>(textBox10->Text);
          string                           m9                          =
msclr::interop::marshal_as<std::string>(textBox11->Text);
```

```
          string                          m10                          =
msclr::interop::marshal_as<std::string>(textBox12->Text);
          string                          m11                          =
msclr::interop::marshal_as<std::string>(textBox13->Text);
          string                          m12                          =
msclr::interop::marshal_as<std::string>(textBox14->Text);


          product.push_back({               type,               name,
m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12 });

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
```

Для видалення відповідного товару за назвою було створено наступий код програми

```
     vector<vector<string>> v;
          for (int i = 0; i < product.size(); ++i)
               if                  (product[i][1]                  !=
msclr::interop::marshal_as<std::string>(textBox15->Text))
                    v.push_back(product[i]);
          product = v;

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
```

```
                listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
                for (int j = 0; j < product[i].size(); ++j)
                     listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
                // добавляем созданный элемент listViewItem (строку)
в listView1
                listView1->Items->Add(listViewItem);
           }
     }
```

Для збереження інформаці написано наступний код

```
    private:   System::Void   button5_Click(System::Object^   sender,
System::EventArgs^ e) {
          ofstream myFile;
          string                              str                              =
msclr::interop::marshal_as<std::string>(textBox16->Text);
          myFile.open(str);
          for (int i = 0; i < product.size(); ++i)
          {
                for (int j = 0; j < product[i].size(); ++j)
                     myFile << product[i][j] << ";";
                myFile << endl;
          }
     }
```

Також було обрано бульбашкове сортування

```
    private: System::Void button4_Click(System::Object^ sender, System::EventArgs^ e) {
          for (int i = 0; i < product.size() - 1; ++i)
                for (int j = i + 1; j < product.size(); ++j)
                     if (comboBox2->SelectedIndex < 2) {
                          if          (product[i][comboBox2->SelectedIndex]          >
product[j][comboBox2->SelectedIndex])
                          {
                               vector<string> s = product[i];
                               product[i] = product[j];
                               product[j] = s;
                          }
                     }
                     else
                     {
                          if        (stoi(product[i][comboBox2->SelectedIndex])        >
stoi(product[j][comboBox2->SelectedIndex]))
                          {
                               vector<string> s = product[i];
                               product[i] = product[j];
                               product[j] = s;
                          }
                     }
          index = 0;
```

```
            this->listView1->Items->Clear();

            // добавляем строки в listView1
            for (int i = 0; i < product.size(); i++)
            {
                  index++;
                  // создадим объект ListViewItem (строку) для listView1
                  ListViewItem^ listViewItem = gcnew ListViewItem();
                  listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
                  for (int j = 0; j < product[i].size(); ++j)
                        listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
                  // добавляем созданный элемент listViewItem (строку) в listView1
                  listView1->Items->Add(listViewItem);
            }
      }
```

Та реалізовані методи прогнозування

Балансовий метод

```
    private:   System::Void   button6_Click(System::Object^   sender,
System::EventArgs^ e) {
          for (int i = 0; i < product.size(); ++i)
          {
               double sum = 0;
               for (int j = 2; j < 14; ++j)
                    sum += stoi(product[i][j]);
               sum /= 12;
               if (product[i].size() == 15)
                    product[i][14] = to_string(sum);
               else
                    product[i].push_back(to_string(sum));
          }

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
```

```
                // добавляем созданный элемент listViewItem (строку)
в listView1
                listView1->Items->Add(listViewItem);
           }
     }
```

Екстраполяційний метод

```
    private:  System::Void  button7_Click(System::Object^  sender,
System::EventArgs^ e) {
          for (int i = 0; i < product.size(); ++i)
          {
               double sum = 0;
               sum     =     2     *     stoi(product[i][12])     -
stoi(product[i][11]);
               if (product[i].size() == 15)
                    product[i][14] = to_string(sum);
               else
                    product[i].push_back(to_string(sum));
          }

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
```

Та нормативний метод

```
     private:  System::Void  button8_Click(System::Object^  sender,
System::EventArgs^ e) {
          for (int i = 0; i < product.size(); ++i)
          {
               int nx = product[i].size() - 2;
               int nx2 = nx * 2;
               double* am2 = new double[nx2];
```

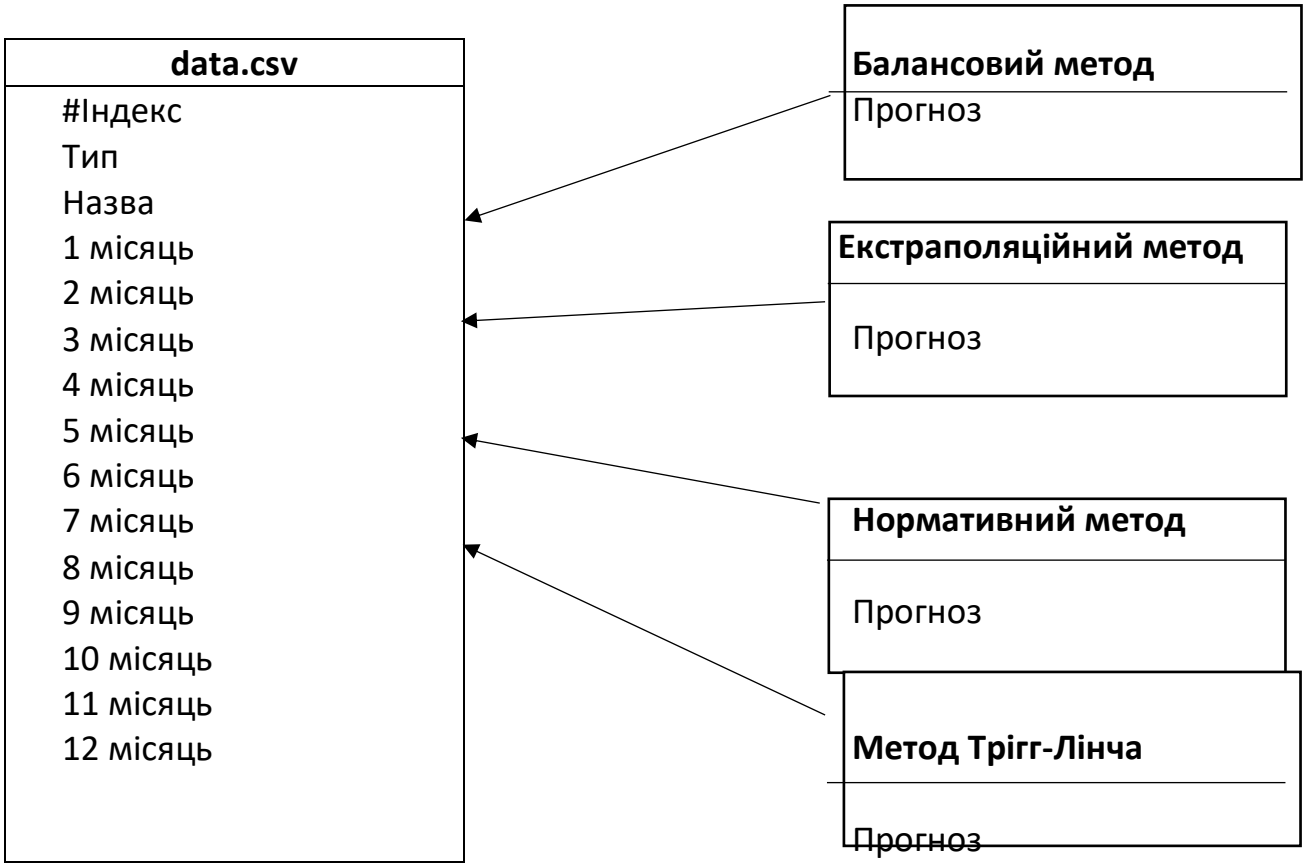
```
                for (int j = 0; j < nx2 - 1; j++)
                {
                     if (j % 2 == 0)
                          am2[j] = (double)stoi(product[i][j / 2 +
2]);
                     if (j % 2 != 0)
                          am2[j] = (double)(stoi(product[i][i / 2 +
2]) + stoi(product[i][j / 2 + 1 + 2])) / 2;
                }
                if (product[i].size() == 15)
                     product[i][14] = to_string(am2[nx / 2 - 1 + 2]);
                else
                     product[i].push_back(to_string(am2[nx / 2 - 1 +
2]));
           }

           index = 0;
           this->listView1->Items->Clear();

           // добавляем строки в listView1
           for (int i = 0; i < product.size(); i++)
           {
                index++;
                // создадим объект ListViewItem (строку) для
listView1
                ListViewItem^ listViewItem = gcnew ListViewItem();
                listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
                for (int j = 0; j < product[i].size(); ++j)
                     listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
                // добавляем созданный элемент listViewItem (строку)
в listView1
                listView1->Items->Add(listViewItem);
           }
     }
```

Та методом Трігг-Ліча

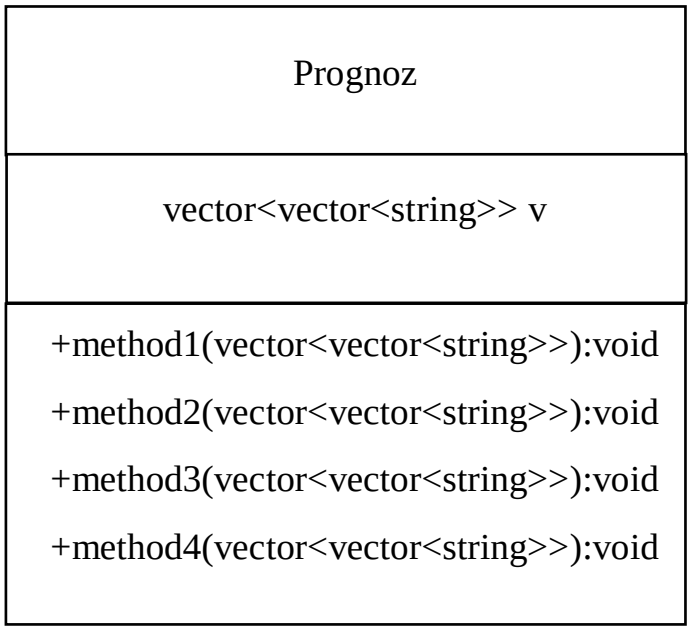
```
    private: System::Void button9_Click(System::Object^ sender,
System::EventArgs^ e) {
     for (int i = 0; i < product.size(); ++i)
     {
           double sum = 0;
           for (int j = 2; j < 14; ++j)
                sum += stoi(product[i][j]);
           sum /= 12;
           double delta = abs(stoi(product[i][12]) -
stoi(product[i][13]));
           double xi = sum * (stoi(product[i][12])+delta*1.005) +
(1.1 - sum) * stoi(product[i][13]);
           if (product[i].size() == 15)
                product[i][14] = to_string(xi);
           else
                product[i].push_back(to_string(xi));
```

```
    }

    index = 0;
    this->listView1->Items->Clear();

    // добавляем строки в listView1
    for (int i = 0; i < product.size(); i++)
    {
        index++;
        // создадим объект ListViewItem (строку) для listView1
        ListViewItem^ listViewItem = gcnew ListViewItem();
        listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
        for (int j = 0; j < product[i].size(); ++j)
            listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
        // добавляем созданный элемент listViewItem (строку) в
listView1
        listView1->Items->Add(listViewItem);
    }
   }
```

Схема бази даних



Діаграма класів



3.3 Опис функціоналу

Запустивши файл ProGnoz.exe (рис. 3.1)

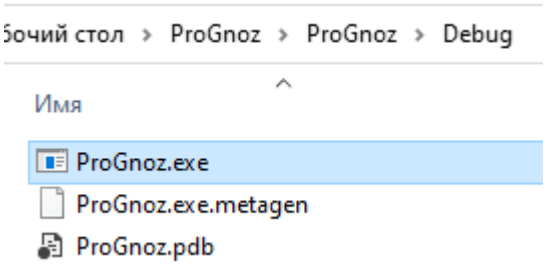


Рис.3.1. – Шлях до файлу

Запускаючи програму перед нами з’явиться наступне вікно: (рис. 3.2)

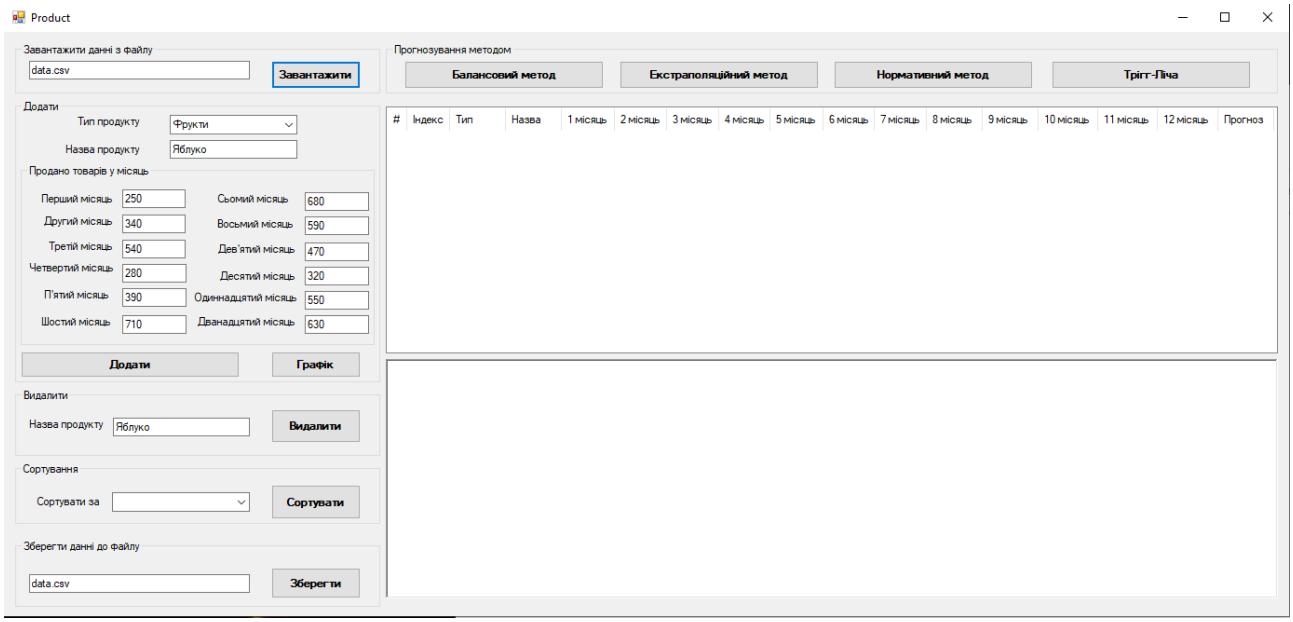


Рис. 3.2. – початковий екран програми

В ньому ми бачимо наступні пункти:

- Завантажити данні із файлу. Вказавши шлях або просто назву файлу, відбудеться завантаження бази даних (рис. 3.3)

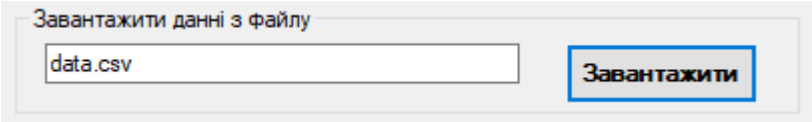


Рис. 3.3. – Завантаження із файлу

- Додати. В ньому можна додати продукт задавши в ньому тип продукту, назву продукту, та кількість проданого товару у кожний місяць.(рис. 3.4)

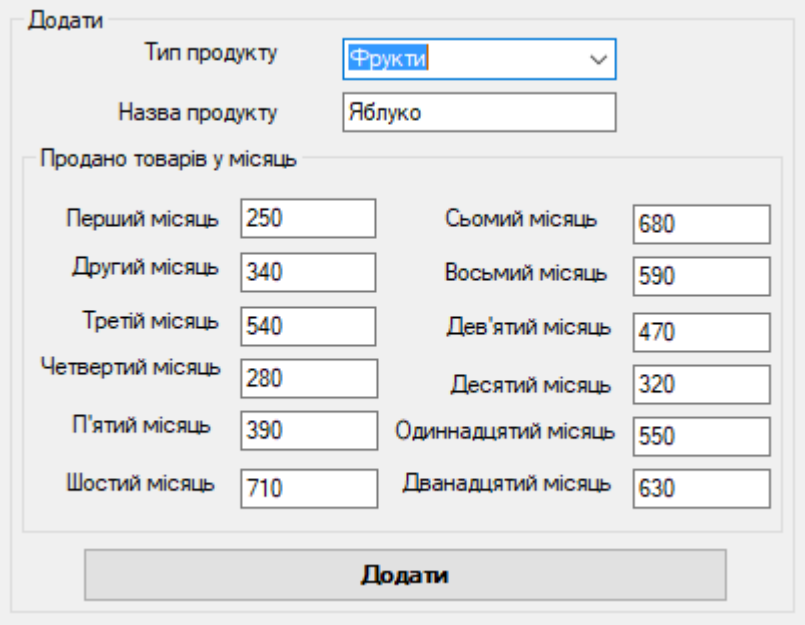


Рис. 3.4. – Додати продукт

- Видалити. В цьому пункті можна видали з таблиці вказану у полі назву продукту (рис. 3.5)

Рис. 3.5. – Видалити продукт

- Сортування продукту. Вибираємо відповідний пункт за яким хочемо відсортувати масив товарів (рис. 3.6)

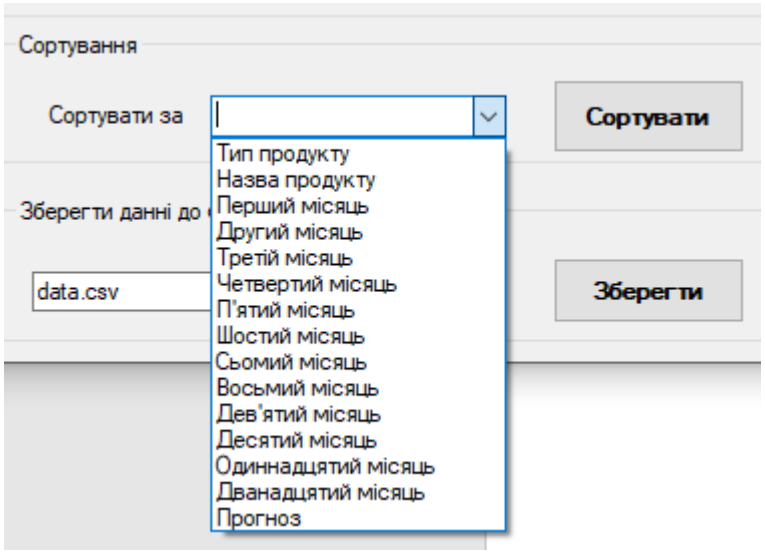


Рис. 3.6. – Сортування товарів

- Зберегти данні до файлу. В цьому пункті вказується назва файлу в якому буде збережена база даних.(рис. 3.7)

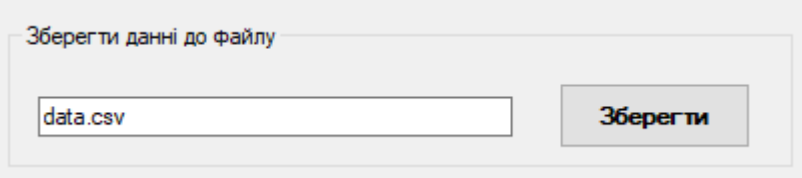


Рис. 3.7. – Збереження даних

- Прогнозування методами. При натисканні кнопок. Відбувається прогнозування відповідним методом.(рис. 3.8)

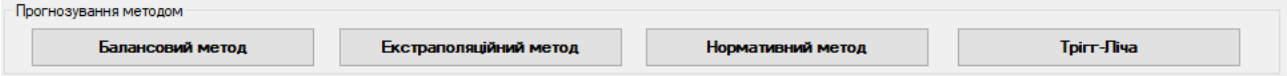


Рис. 3.8. – Прогнозування методами

- Таблиця даних (рис. 3.9)

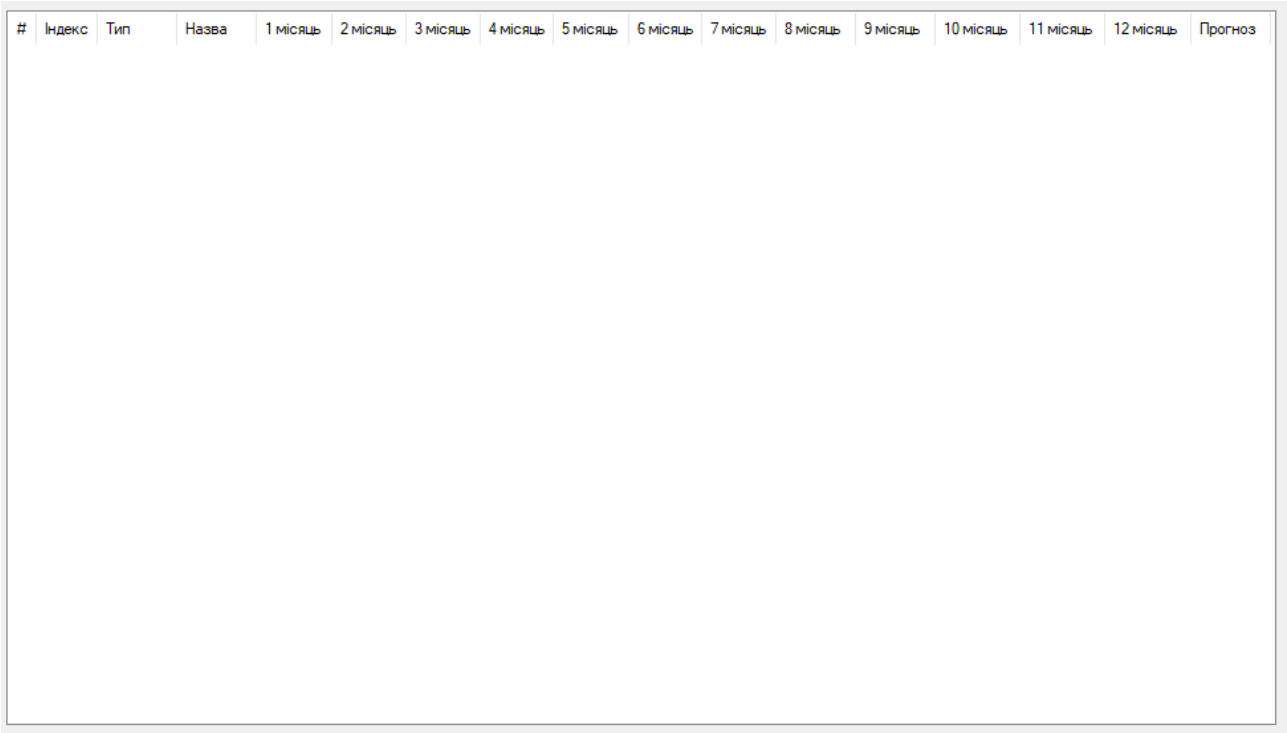


Рис. 3.9. – Таблиця даних

Тестування

Тестування відбувалося багато разів. З різними прикладами і даними (рис. 3.10)

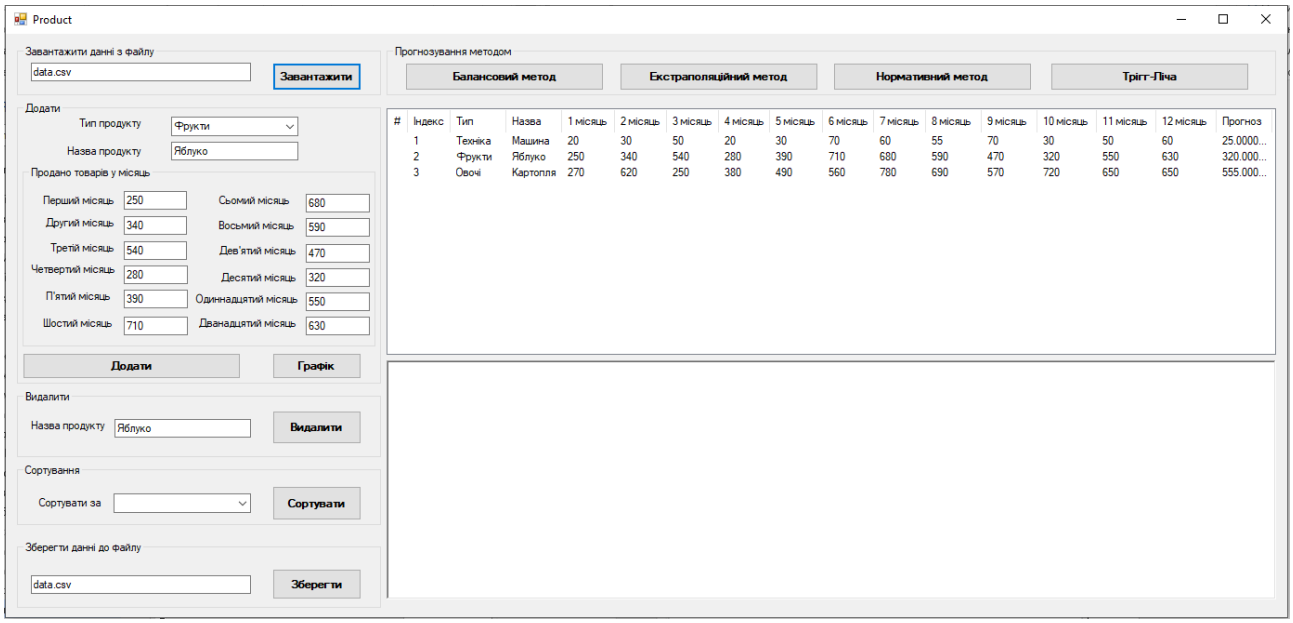


Рис. 3.10. – тестування програми(Завантажити)

Після додавання продукту (рис. 3.11)

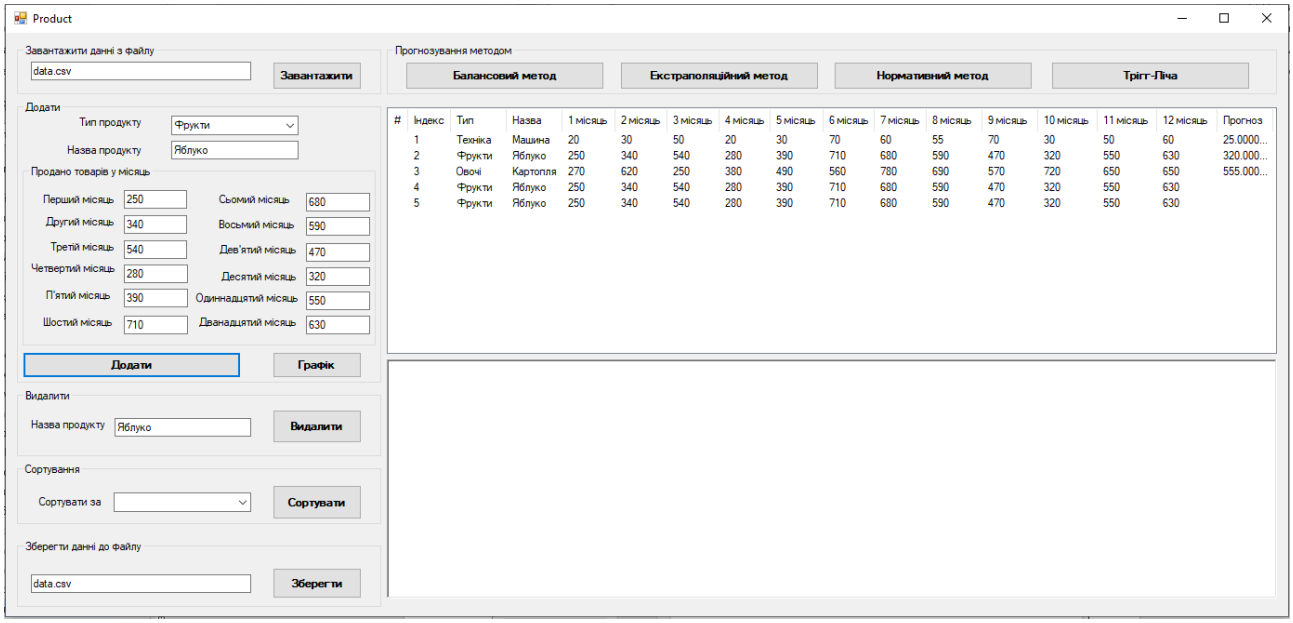


Рис. 3.11. – додавання продукту

Після видалення товару «Картопля» (рис. 3.12)

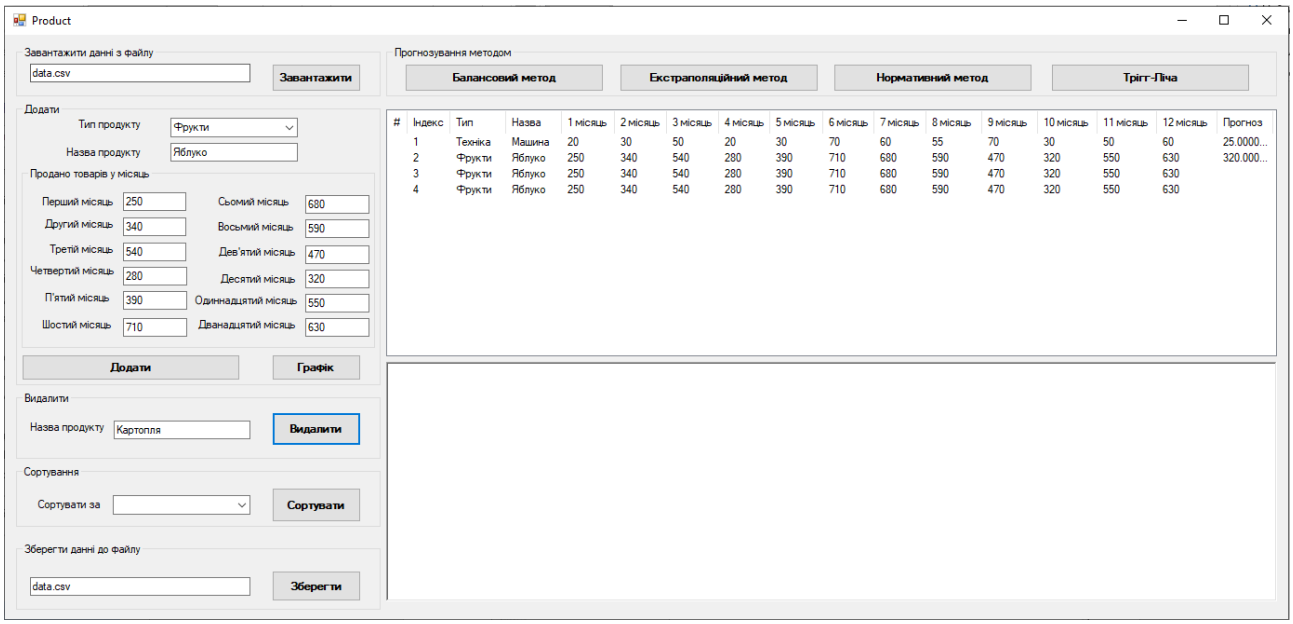


Рис. 3.12. – видалення продукту

Сортування за першим місяцем (рис. 3.13)

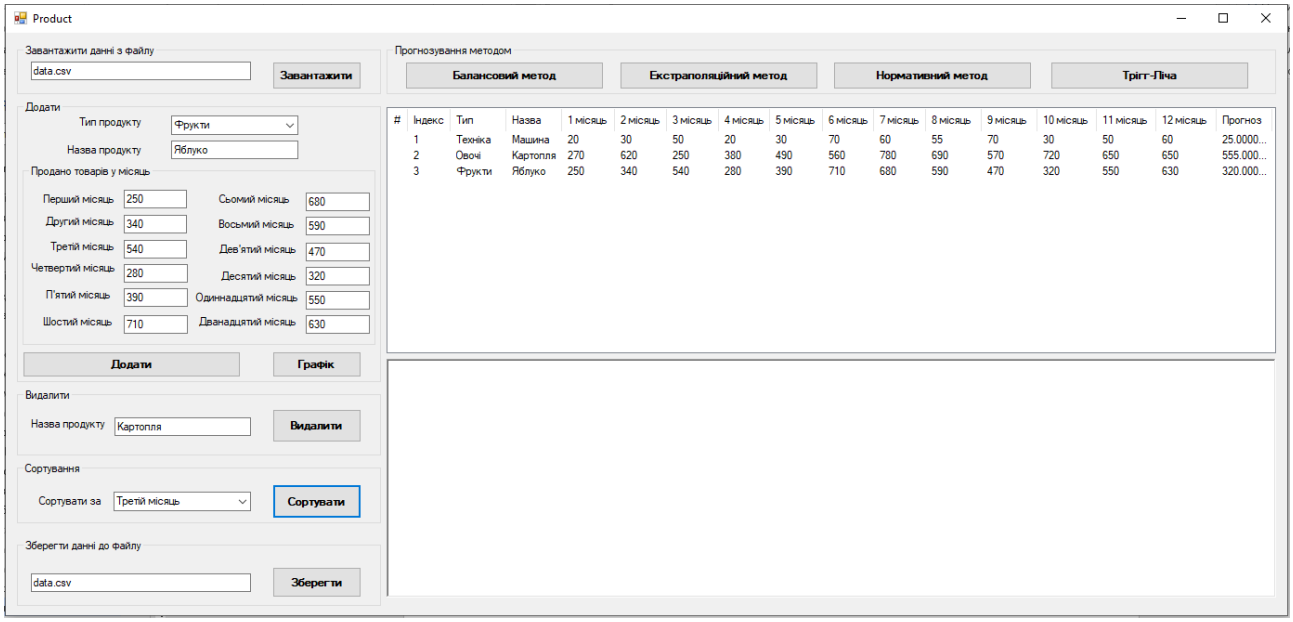


Рис. 3.13. – сортування продукту

Побудова графіку (рис. 3.14)

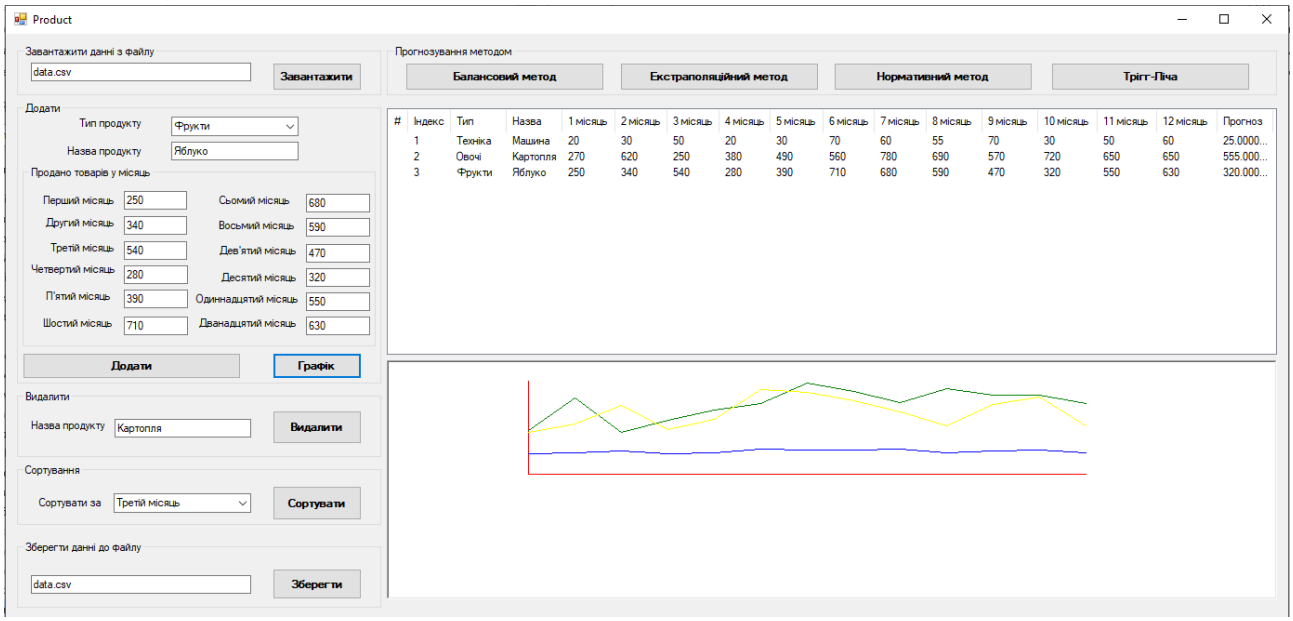


Рис. 3.14. – Побудова графіку

Прогнозування балансовим методом (рис. 3.15)

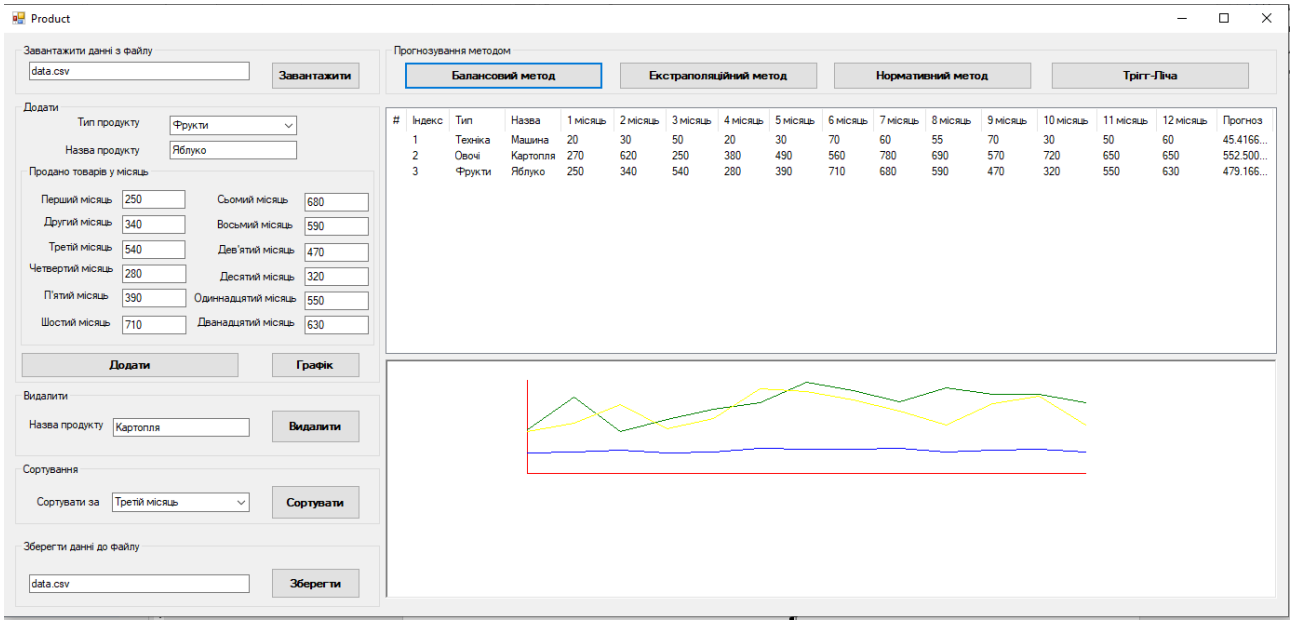


Рис. 3.15. – Прогнозування балансовим методом

Прогнозування екстраполяційним методом (рис. 3.16)

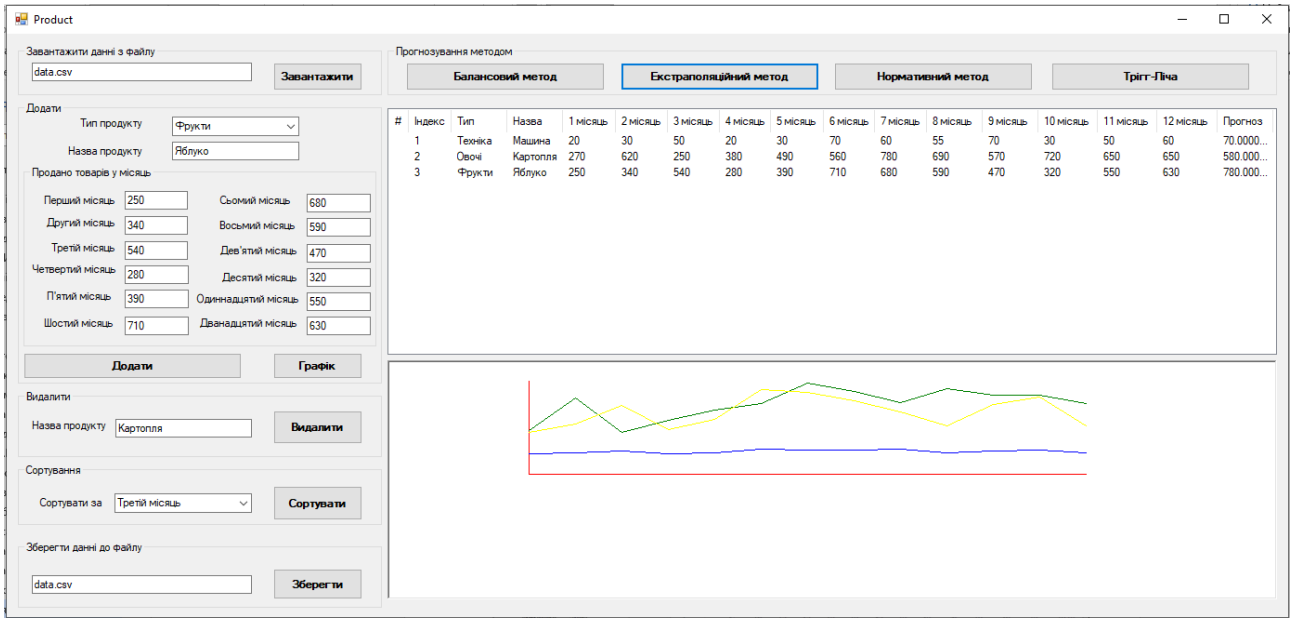


Рис. 3.16. – Прогнозування екстраполяційним методом

Прогнозування нормативним методом(рис. 3.17)

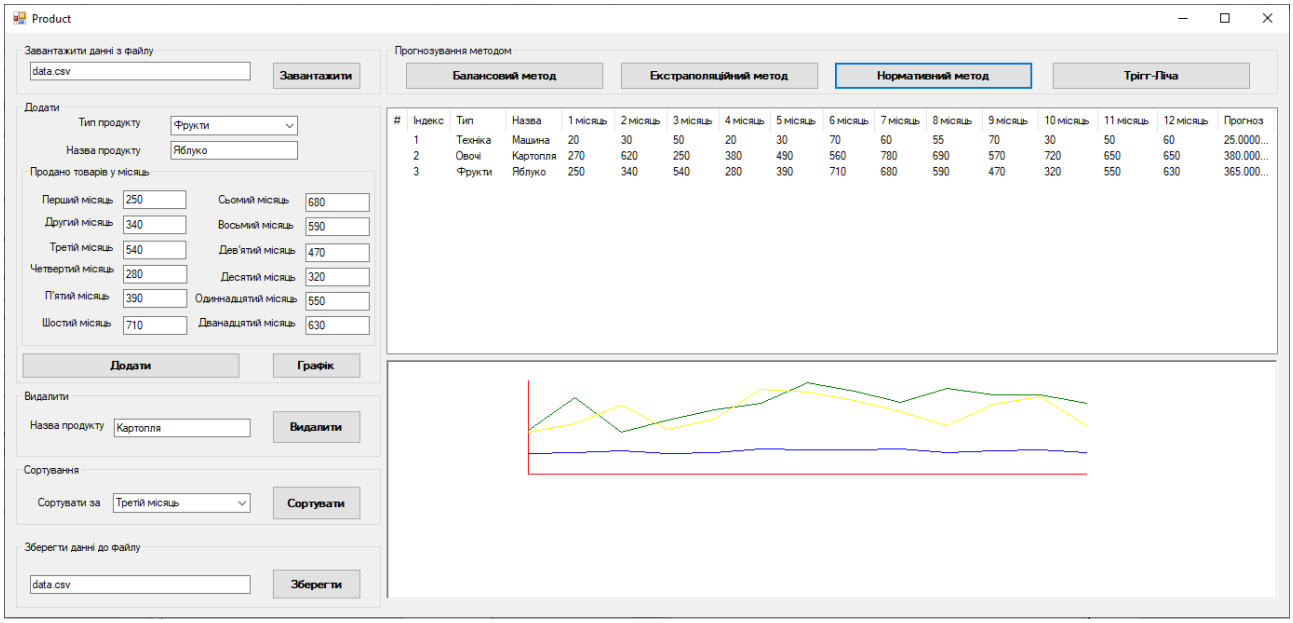


Рис. 3.17. – Прогнозування нормативним методом

Прогнозування методом Трігг-Ліча (3.18)

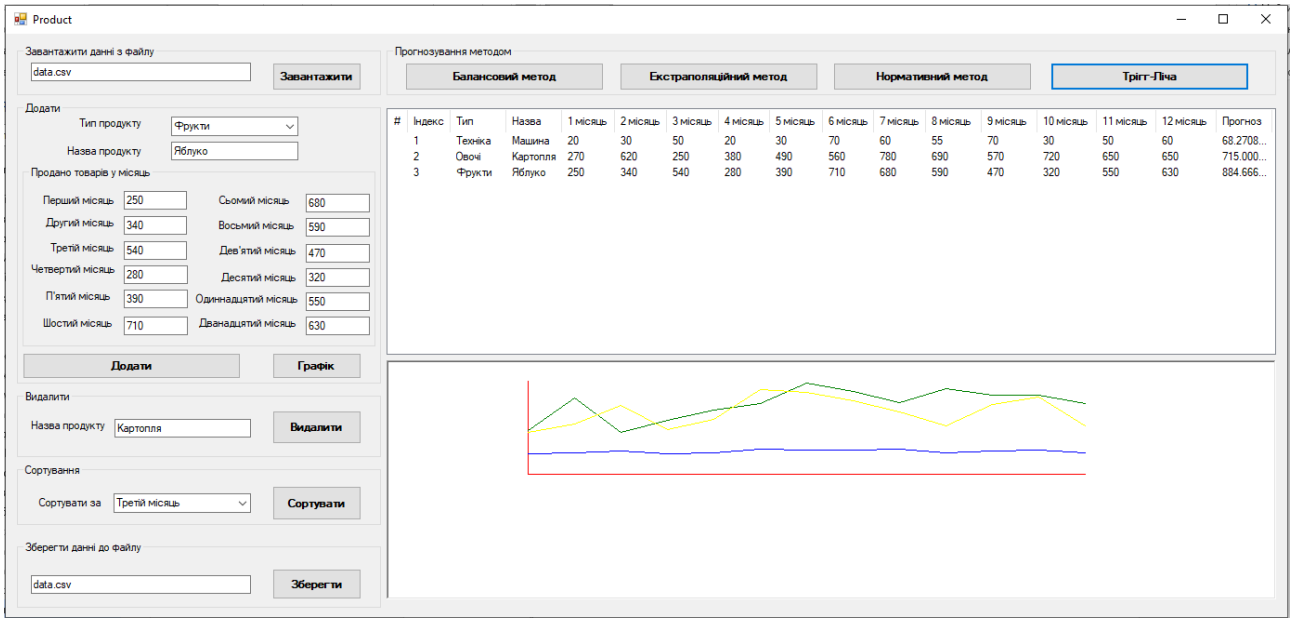


Рис. 3.18. – Прогнозування методом Трігг-Ліча

Висновки до розділу 3

В третьому розділі ідеться опис розроблювального агоритму. Розповідається про розробленні схеми бази даних, про діаграми класів.

Описується функціонал розробленої програми та розповідається про тестування програми.

У ході виконання дисертаційного дослідження розроблено архітектуру та реалізовано інформаційну систему, що забезпечує автоматизоване управління ланцюгами поставок, орієнтоване на прогнозуванні попиту на товар. Для ефективної інтеграції розробленого прикладного рішення запропоновані механізми, що є реалізацією інтеграційної шини на основі патерну та технології з реалізацією підтримки баз даних.

Використання програми дозволило скоротити витрати у забезпеченні виробництва.

Застосовуючи запропонований метод оцінки ефективності управління логістичними витратами, можна прийняти виважене управлінське рішення щодо коригування обсягу та структури логістичних витрат у майбутньому.

Логістична вартість формується у відповідних підсистемах логістичного менеджменту, а саме: підсистемі управління транспортно-заготівельною логістикою, підсистемі управління запасами, підсистемі інформаційного забезпечення управління логістичним забезпеченням.

Враховуючи викладене, вважаємо за доцільне розробити альтернативний метод оптимізації логістичних витрат підприємства у розрізі трьох підсистем логістичного менеджменту.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Вимоги безпеки під час виконання робіт за ПК

В межах даного розділу роботи розглянемо вимоги безпеки під час виконання робіт за ПК, так як система система автоматизованого тестування безпеки веб-додатків розроблюється безпосередньо з застосуванням ПК.

Робота за комп'ютером передбачає ряд шкідливих факторів та загроз. Що пов'язано з можливістю отримання травм та професійних захворювань, то закон України «Про охорону праці» від 21.11.02 р визначає основні положення відносно реалізації конституційного права громадян на охорону положення відносно реалізації конституційного права громадян на охорону їх життя та здоров'я в процесі трудової діяльності, регулює за допомогою відповідних органів державної влади відносини між роботодавцем і працівником з питань безпеки, гігієни працівника виробничого середовища та встановлює єдиний порядок організації охорони праці в Україні.

Дана робота розроблялась на робочому місці, яке знаходиться на третьому поверсі триповерхової адміністративної будівлі.

Площа приміщення складає 50 м^2, що відповідає санітарним нормам, згідно з якими норма на одного працюючого повинна бути не менш 6 м^2.

Висота приміщення 3,0 м. таким чином обсяг приміщення складає 150 м^3, по нормам – не менш 20 м^3 Перелік шкідливих та небезпечних факторів, які діють при роботі на ПЕОМ наведений в табл. 4.1 згідно з та ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [14] та ДСН 33.6.042-99 [15].

Таблиця 4.1 – Шкідливі та небезпечні фактори виробничого середовища

Найменування факторів	Джерела їх виникнення	Параметр, що нормується, та нормативне значення

1	2	3
1. Рівень електромагнітного випромінювання	ЕПТ монітора, системний блок, мережа живлення	Відстань - 50 см, навколо ПК 2-5 кГц – 25 В/м [14]
2. Ультрафіолетове випромінювання	Комп’ютер	Щільність потоку ультрафіолетового випромінювання 10 Вт/м [14]
3. Емоційні перенавантаження, напруга зорового аналізатору	Складність виконання завдань	Зниження реакції користування на звук і світло на 40-50 % [14]
4. Підвищений рівень шуму	Вентилятор, система освітлення, друкувальні прилади	Рівень звуку LA=50 дБ (А) [14]
5. Підвищене значення напруги в електричній мережі	Блок живлення	I=0,6 мА; U=36 В [17]
6. Недолік природнього освітлення	Неправильне планування розташування комп’ютера	КПО, №, Е, лк [14]
7. Вібрація	Вентиляційна система	Віброприскорення, м/с^2, віброшвидкість, м/с або їх рівні LA, LV, дБ [14]
8. Виробничий пил	Статична електрика, накопичена на поверхні комп’ютера	ГДК=4 мг/м^3 [16]
9. Несприятливі температури мікроклімату	Не задовільна робота опалення або вентиляції	Температура (t, °С), вологість (ф, %), швидкість руху повітря (V, м/с). [15]

Оптимальні параметри мікроклімату встановлюються залежно від категорії робіт по фізичному навантаженню [14]. Забезпечення необхідних параметрів мікроклімату досягається у холодний період кондиціюванням та системою опалення, а в теплий період лише системою кондиціювання, згідно з ДБН В.2.5.-67: 2013 [18].

Таблиця 4.2 – Оптимальні параметри мікроклімату

Категорія робіт по вазі	Період року	Температура, °С	Відносна вологість, %	Швидкість руху повітря, м/с

Легка – Іа	теплий	23-25	40-60	0,1-0,2
	холодний	22-24	40-60	0,1

Освітлення виробничих, службових і допоміжних приміщень регламентується ДБН В.2.5.-28-2006.

Штучне та природне освітлення.

В даному випадку використовується комбіноване освітлення: природне бокове вранці та штучне ввечері.

Виконувана робота відноситься до III розряду зорової праці. Мінімальній розмір об'єктів від 0,3 до 0,5 мм, фон світлий, контраст великий, підрозряд зорових робіт – «Г» в приміщенні забезпеченому комбінованим освітленням: у світлий час доби – бокове однобічне природне освітлення – три віконних прорізи, у темний час загальне чи місцеве рівномірне штучне.

В табл. 4.3 наведені норми освітлення для даного розряду і точності зорових робіт.

Таблиця 4.3 – Характеристики виробничого освітлення

Точність зорових робіт	Мінімальний розмір об'єкта розрізнення, мм	Розряд зорових робіт	Характеристика типу фону	Контраст об'єкта розрізнення з фоном	Розряд зорових робіт	Нормативне значення параметрів освітлення	
						Природне, %	Штучне, лк
Високої точності	0,3-0,5	III	Світлий	Великий	Г	1,2	400

Джерелом шуму в кабінеті може слугувати така техніка: телефон, принтер, факс, комп'ютер, кондиціонер. Рівень шуму не повинен перевищувати 50 дБ. Рівень вібрації в умовах комфортної роботи, не повинен перевищувати 75 дБ.

Для зменшення рівня звуку та вібрації застосовуються демпфуючі матеріали.

Основними методами захисту є: зниження шуму та вібрації в джерелі

(підставки, шумопоглинальні корпуси) і на шляху розповсюдження (ширми, шумопоглинальні стійки), застосування індивідуальних засобів та організаційно-профілактичних методів захисту.

Для захисту від електромагнітного випромінювання застосовується спеціальне покриття екрану. Напруга електромагнітних полів у діапазоні 1 – 12 кГц, 60 – 300кГц по магнітній і електричній складовій повинні відповідати вимогам до ДСанПіН 3.3.2-007-98 [14].

Повітря зовнішнього середовища містить (табл. 4.4) [14].

Таблиця 4.4 – Рівень іонізації повітря при роботі на ПЕОМ

Рівні	Кількість іонів в 1 $см^3$ повітря	
	Позитивні	Негативні
Мінімально необхідні	400	600
Оптимальні	1500-3000	3000-5000
Максимально припустимі	50000	50000

Напруга електромагнітних полів у діапазоні 1 – 12 кГц, 60 – 300 кГц по магнітній і електричній складовій повинні відповідати вимогам до ДСанПіН 3.3.2-007-98 [14].

При проектуванні систем електропостачання, монтаж силового електроустаткування й електричного освітлення в будинках і приміщеннях для ЕОМ необхідно дотримуватися вимог нормативно-технічної документації.

Комплекс необхідних заходів щодо техніки безпеки визначається, виходячи з видів електроустановки, її номінальної напруги, умов середовища, типу приміщення й доступності електроустаткування.

Документом ПУЕ 2017 [17] передбачені наступні міри електробезпеки:

1) конструктивні заходи: персональна ЕОМ відноситься як електроустановка до 1000 В закритого виконання, усі рубильники встановлені в закритих корпусах, усі струмоведучі частини розміщені в захисних коробах або

покриті шаром ізоляції, який виключає можливість дотику до них. Комп'ютер має робочу ізоляцію і елементи заземлення;

2) експлуатаційні міри: при роботі на ЕОМ необхідно дотримувати правила техніки безпеки при роботі з високою напругою, не підключати і не відключати кабелі при включеній напрузі мережі, технічне обслуговування і ремонт проводити тільки при вимкнутому живленні [17];

3) схемно – конструктивні міри: в електричних мережах із глухо заземленим нейтральним провідником як схемно – конструктивну міру безпеки застосовують занулення – навмисне з'єднання металевих неструмоведучих частин комп'ютера з нейтральним провідником [17].

Відповідно до вимог ДБН В 1.1.7-2016 [19] пожежна безпека забезпечується наступними заходами, які застосовуються до категорії В: системою пожежного захисту; організаційними заходами щодо пожежної безпеки; системою запобігання пожеж: запобігання утворенню горючого середовища, та запобігання утворення у горючому середовищі джерел запалювання.

Для зменшення небезпеки утворення в стальному середовищі джерел запалювання передбачено:

1) використання устаткування, що відповідає класу пожежобезпечної зони ПІІа: ступінь захисту електроапаратури повинна бути не менш IP-44, ступінь захисту світильників IP-23, відповідно до ДБН В 1.1.7-2016 [19];

2) блискавковідвід будинків, споруджень і устаткування для даного класу пожежонебезпеки, зони П-ІІа і місцевості із середньою грозовою діяльністю 20 і більше грозових годин у рік, тобто встановлена ІІІ категорія блискавко захисту відповідно до ДСТУ EN 62305-1:2012 [20];

3) застосування заземлення захисного екрану для стоку статичної електроенергії; використання для гасіння пожежі вуглекислого вогнегасника ВВ-2.

Організаційними заходами протипожежної профілактики є: навчання виробничого персоналу протипожежним правилам; видання необхідних

інструкцій, плакатів, засобів наочної агітації, плану евакуації персоналу у випадку пожежі.

4.2 Дії персоналу в надзвичайних ситуаціях

Розглянемо дії персоналу у різних надзвичайних ситуаціях. Дії у разі вибуху.

Вибух. Основні вражаючі фактори вибуху: повітряна ударна хвиля та уламкові поля, що утворюються уламками зруйнованих об'єктів, що летять, технологічного обладнання, вибухових пристроїв

При загрозі вибуху слід лягти на живіт, захищаючи голову руками, подалі від вікон, засклених дверей, проходів, сходів.

Якщо стався вибух, вжити заходів щодо недопущення пожежі та паніки; надати першу допомогу постраждалим. Кожен працівник при виявленні вогнища або ознак горіння (задимлення, запах гару, підвищення температури тощо) повинен негайно повідомити про це за телефоном «101».

При цьому назвати найменування об'єкта, місце вибуху, пожежі, а також своє прізвище; вжити заходів щодо евакуації людей, гасіння пожежі та збереження матеріальних цінностей.

Вимоги щодо використання первинних засобів пожежогасіння: Вуглекислотні вогнегасники (ОУ-2, ОУ-3, ОУ-5, ОУ-6, ОУ-7 тощо) призначені для гасіння загорянь різних горючих речовин, виключення, коли горіння яких відбувається без доступу повітря,

Для приведення в дію вуглекислотних вогнегасників необхідно розтруб направити на палаючий предмет, зірвати пломбу, висмикнути чеку, натиснути на важіль (або повернути маховик вентиля вліво вщент), направити струмінь на полум'я. Тримати вогнегасник вертикально, перевертати його не потрібно.

Щоб уникнути обморожування, не торкатися металевої частини розтрубу. оголеними частинами тіла. При гасінні електроустановок, що знаходяться під напругою, не допускається підводити до них розтруб ближче 1м.

Внутрішні пожежні крани призначені для подачі води.

Асбестове полотно, повсть (кошма) використовуються для гасіння невеликих вогнищ загоряння будь-яких речовин та матеріалів, горіння яких не може відбуватись без доступу повітря.

Дії у разі хімічної аварії.

Небезпека хімічної аварії для людей полягає в порушенні нормальної життєдіяльності організму та можливості віддалених генетичних наслідків, а за певних обставин – у летальному результаті при потраплянні АХНР в організм через органи дихання, шкіру, слизові оболонки, рани та разом з їжею.

При отриманні сигналу про хімічну аварію включити радіоприймач для отримання достовірної інформації про аварію та рекомендовані дії.

Закрити вікна, вимкнути електропобутові прилади. Для захисту органів дихання використовувати ватно-марлеву пов'язку або підручні вироби з тканини, змочені у воді, 2-5%-ному розчині харчової соди (для захисту від хлору), 2%-ному розчині лимонної або оцтової кислоти (для захисту від аміаку).

При неможливості залишити зону зараження щільно закрити двері, вікна, вентиляційні отвори та димарі; щілини в них заклеїти папером чи скотчем.

Не ховатися на перших поверхах будівель, у підвалах та напівпідвалах.

При підозрі на поразку АХНР виключити будь-які фізичні навантаження, прийняти рясне питво (молоко, чай) та негайно звернутися до лікаря.

Утримуватися від вживання водопровідної води – до офіційного висновку щодо її безпеки. На зараженій місцевості рухатися швидко, але не бігти, піднімаючи пил, не торкатися навколишніх предметів, не наступати пролиту рідину або порошкоподібні розсипи невідомих речовин.

Дії у разі обвалення будівель, споруд. Повне або часткове раптове обвалення будівлі – це надзвичайна ситуація природного або техногенного характеру, яка також виникає за причини помилок, допущених на етапі проектування.

Руйнування комунально-енергетичних мереж, утворення завалів, травмуванню та загибелі людей. Почувши вибух або виявивши, що будівля втрачає свою стійкість, негайно покинути його.

Залишаючи приміщення, спускатися сходами, а не на ліфті: він у будь-який момент може зупинитись.

Не панікувати, не влаштовувати тисняву у дверях під час евакуації. Зупиняти тих, хто збирається стрибати з балконів (поверхів вище першого) та через засклені вікна. Якщо відсутня можливість покинути будівлю, зайняти безпечне місце: отвори капітальних внутрішніх стін, кути, утворені капітальними внутрішніми стінами, під балконами каркасу (вони захищають від падаючих предметів та уламків). Відкрити двері з приміщення, щоб забезпечити вихід.

Не піддаватися паніці та зберігати спокій. Триматися подалі від вікон, електроприладів.

Якщо виникла пожежа, негайно спробувати загасити її. Телефон використовувати лише для виклику представників правоохоронних органів, пожежної охорони, лікарів, рятувальників

Не користуватися сірниками: існує небезпека вибуху внаслідок витоку газу. Опинившись надворі, не стояти поблизу будинку. Перейти на відкритий простір.

Дії у разі знаходження під завалом:

Дихати глибоко, не піддаватися паніці.

По можливості надати собі першу допомогу. Пристосуватися до обстановки та озирнутися, пошукати вихід. Спробувати визначити, де знаходиться людина, чи немає інших людей: прислухатись, подати голос.

Слід пам'ятати: людина здатна витримати спрагу і голод протягом тривалого часу, якщо не марно витрачати енергію.

Пошукати в кишенях або поблизу предмети, щоб подати світлові або звукові сигнали: ліхтарик або металеві предмети, якими можна постукати по трубі чи стіні (привернути увагу рятувальників).

Якщо єдиним виходом є вузький лаз - протиснутися через нього. Для цього розслабити м'язи та рухатися, притиснувши лікті до тіла.

Також розглянемо дії при електротравматизмі. Згідно з наказом № 398 «Порядок надання домедичної допомоги постраждалим при ураженні електричним струмом та блискавкою» [21], заходи першої допомоги залежать від стану потерпілого після визволення його від електричного струму. Для визначення стану необхідно вжити таких заходів:

- покласти потерпілого спиною на тверду поверхню;
- перевірити наявність у потерпілого дихання;
- перевірити наявність у потерпілого пульсу на сонній артерії;
- з'ясувати стан зіниці, широка зіниця вказує на погіршення кровопостачання.

У всіх випадках ураження електричним струмом виклик лікаря є обов'язковим незалежно від стану потерпілого.

Якщо потерпілий знаходиться при свідомості, його треба покласти у зручне положення і до прибуття лікаря забезпечити спокій, обов'язково спостерігаючи задиханням і пульсом.

Не можна дозволяти потерпілому рухатись, продовжувати роботу. Якщо лікаря швидко викликати не можна, необхідно терміново доставити потерпілого у медичний пункт.

Якщо потерпілий знаходиться у непритомному стані, його необхідно покласти, розстебнути одяг, забезпечити приплив свіжого повітря, дати понюхати нашатирний спирт, бризнути на нього водою і забезпечити спокій. У той же час потрібно викликати лікаря. Якщо потерпілий дихає погано, рідко і судомно, йому необхідно робити штучне дихання і непрямий масаж серця.

У разі відсутності в потерпілого ознак життя не можна вважати його померлим. Якщо в такому стані потерпілому не буде надано негайну першу допомогу у вигляді штучного дихання і зовнішнього масажу серця, то настане смерть.

Оживлення організму, ураженого електричним струмом, може бути проведено кількома способами. Всі вони базуються на штучному диханні. Починати штучне дихання слід негайно після вивільнення потерпілого від електричного струму і проводити безперервно до досягнення позитивного результату.

Штучне дихання необхідно робити безперервно, до прибуття лікаря.

Переносити потерпілого до іншого місця треба тільки в тих випадках, коли йому, чи особі, яка надає допомогу, продовжує загрожувати небезпека.

Ураженого електричним струмом можна визнати померлим тільки за наявності видимих тяжких зовнішніх ушкоджень: роздроблення черепа у разі падіння чи обпалення всього тіла.

В інших випадках констатує смерть лише в лікарні.

Також розглянемо дії при пожежі. Про виникнення пожежі в приміщеннях негайно повідомити пожежну охорону.

При цьому необхідно назвати адресу, зазначити кількість поверхів будівлі, місце виникнення пожежі, обстановку на пожежі, наявність людей, а також повідомити своє прізвище.

Вжити (по можливості) заходи на евакуацію людей, гасіння (локалізацію) пожежі з використанням первинних засобів пожежегасіння та на збереження матеріальних цінностей.

Повідомити про виникнення пожежі керівника (заступників керівника) чи відповідальну компетентну посадову особу та чергового охорони.

У разі необхідності, викликати інші аварійно-рятувальні служби (медичну, газорятувальну тощо).

4.3 Дії працівників в надзвичайних ситуаціях

Забезпечення захисту населення і територій від загроз і надзвичайних ситуацій є одним із найважливіших завдань держави. Актуальність забезпечення природно-техногенної безпеки населення і територій зумовлена тенденцією

зростання людських втрат і територіальних збитків внаслідок небезпечних природних явищ, промислових аварій і катастроф. Зростає ризик виникнення надзвичайних ситуацій природного та техногенного характеру.

Захист населення і територій від надзвичайних ситуацій техногенного і природного характеру – централізовано реалізована система організаційних, технічних, медико-біологічних, фінансово-економічних та інших заходів щодо запобігання і реагування на надзвичайні ситуації техногенного і природного характеру та ліквідації їх наслідків. Відповідні сили і засоби органів виконавчої влади, органів місцевого самоврядування, підприємств, установ і організацій незалежно від форм власності і господарювання, добровільно створених для охорони населення і території, а також матеріальних і культурних цінностей і навколишнє середовище.

Організація та здійснення захисту населення відповідно до вимог Конституції України (1996 р.) та законодавства України:

«Про цивільну оборону в Україні» (1999);

«Про захист організму людини від іонізуючого випромінювання» (1998);

«Про захист населення і територій від надзвичайних ситуацій техногенного і природного характеру» (2000 р.)

«Правовий режим щодо надзвичайного стану»;

Концепція «Захист населення і території у разі загрози та виникнення надзвичайних ситуацій», затверджена Указом Президента України від 26 березня 1999 р., та інші нормативно-правові документи щодо захисту населення у надзвичайних ситуаціях.

Основним завданням цивільної оборони в умовах надзвичайної ситуації є захист населення.

Захист людей – це створення необхідних умов для захисту життя і здоров’я людей у надзвичайних ситуаціях.

Основна мета природоохоронних заходів — уникнути або мінімізувати шкоду популяції.

З метою захисту населення та зменшення втрат і економічних збитків у надзвичайних ситуаціях може бути вжито ряд спеціальних заходів:

– Оповіщення та сповіщення, які досягаються шляхом раннього створення та підтримки постійної готовності загальнодержавної, регіональної та об’єктової системи оповіщення;

– Моніторинг і контроль стану навколишнього природного середовища, продовольства і води забезпечується шляхом створення і підтримки державних і територіальних систем моніторингу і контролю, включаючи існуючі засоби живлення і контролю, незалежно від приналежності;

– Укриття в захисній споруді досягається створенням фонду захисних споруд, в якому захищається все населення за приналежністю (позмінна робота, населення, яке проживає в небезпечних зонах тощо);

– Евакуація - заходи, що вживаються в містах та інших населених пунктах з підвищеною небезпекою об'єктів і у воєнний час.

Основним способом захисту населення є евакуація та поселення в районах за містом;

Здійснювати інженерний захист з метою виконання вимог інженерно-технічного захисту містобудування, розміщення небезпечних об’єктів, будівель, інженерних споруд тощо;

Здійснювати медичний захист для зниження рівня шкоди людям, надання своєчасної допомоги та лікування постраждалим, забезпечення епідемічного здоров’я в надзвичайних ситуаціях;

Біологічний захист включає своєчасне виявлення факторів біологічного забруднення, їх характеру та масштабів, здійснення комплексних адміністративно-господарських, організаційних обмежень та спеціальних протиепідемічних і медичних заходів;

Радіаційний та хімічний захист включає заходи щодо виявлення та оцінки радіаційного та хімічного стану, організацію та проведення дозиметричного та хімічного контролю, розробку типових режимів радіаційного захисту,

забезпечення засобами індивідуального захисту, організацію та проведення спеціального поводження.

5 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

5.1 Поняття охорони навколишнього природного середовища, об’єкти, основні принципи та завдання

Охорона праці - це система законодавчих, організаційно-технічних, соціально-економічних, гігієнічних і лікувально-профілактичних заходів і засобів, призначених для охорони життя, здоров'я і працездатності людини в процесі праці. Завданням охорони праці є мінімізація ймовірності травмування або захворювання працівників під впливом шкідливих виробничих факторів, забезпечення комфортних умов і максимальної продуктивності праці. Закон України "Про охорону праці" роз'яснює основні положення конституційних прав громадян на охорону свого життя і здоров'я в процесі трудової діяльності, регулює відносини між виконавчою владою і працівниками незалежно від форми власності, встановлює єдиний порядок з організації охорони праці в Україні [1].

Завданнями природоохоронного законодавства є регулювання відносин у сфері охорони, використання та відтворення природних ресурсів, забезпечення екологічної безпеки, запобігання та усунення негативного впливу господарської та іншої діяльності на навколишнє природне середовище, охорона природних ресурсів, спадковості та життя. природи, що відноситься до історико-культурної спадщини, ландшафтів та інших природних комплексів, унікальних територій і природних об'єктів [2]. Відповідно до Закону України «Про підприємства України» всі роботодавці повинні у своїй діяльності звертати увагу на дотримання вимог законодавства України щодо охорони праці та охорони навколишнього середовища.

У даній дипломній роботі робота виконується безпосередньо за напрямком диплому та визначеними завданням умовами праці з урахуванням питань охорони праці, що стосуються підприємства.

Система управління гігієною та безпекою праці (СУБП) – це сукупність дій, які приймають, приймають і реалізують рішення щодо здійснення організаційних, технічних, гігієнічних і лікувально-профілактичних заходів.

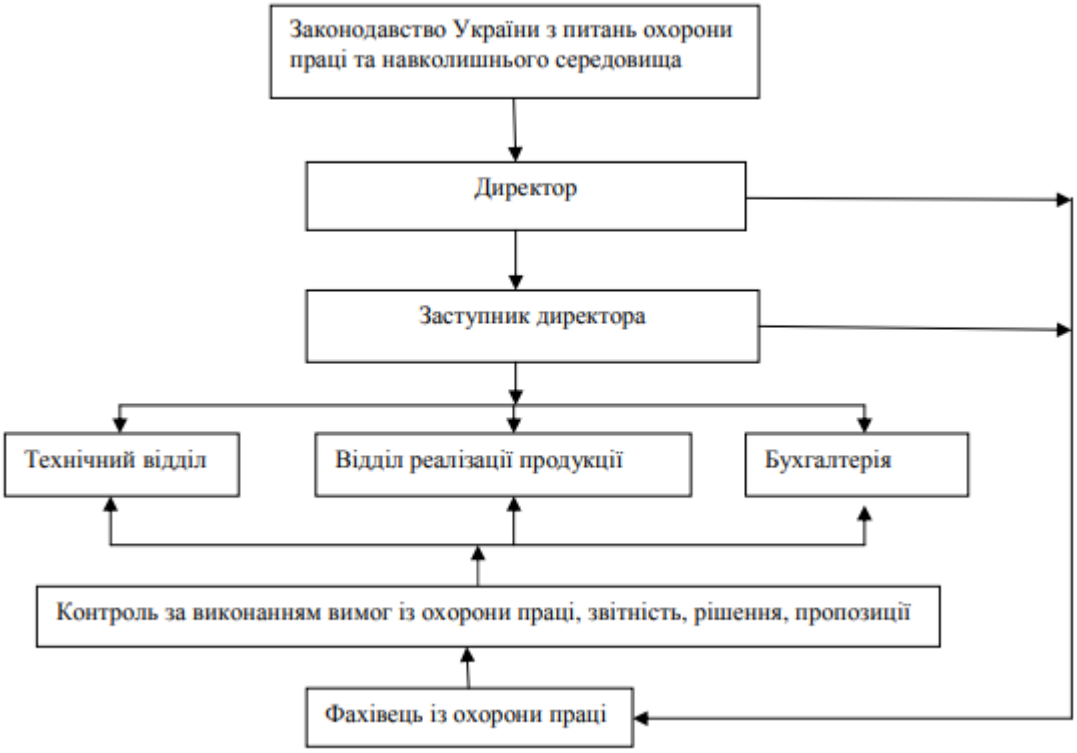


Рисунок 5.1 – структура на підприємстві

Керівництво охороною праці здійснюється: на підприємстві в цілому - безпосередньо відповідальним за підприємство та через його заступників. У сегментах і відділах - за лідерами сегментів. Нагляд за дотриманням вимог охорони праці та навколишнього середовища, підготовку звітів, рішень і рекомендацій щодо поліпшення умов праці здійснюють спеціалісти з охорони праці.

5.2 Права та обов’язки громадян та органів державної влади щодо охорони навколишнього природного середовища

Правові заходи забезпечення екологічної безпеки [7].

Екологічна безпека в Україні забезпечується здійсненням комплексу взаємопов'язаних політичних, економічних, технічних, організаційних, національно-правових та інших заходів. Зміст національних правових заходів неоднаковий. Залежно від спрямованості дії їх можна розділити на кілька видів:

Організація профілактики, нормативного стимулювання, адміністративного примусу, відновлення захисту та забезпечення. Вони утворюють особливий правовий механізм, під яким слід розуміти систему національних правових інструментів, спрямованих на регулювання рівня екологічної безпеки, запобігання погіршенню екологічного стану та діяльності, що є небезпечною для населення та природних систем, а також локалізацію прояви екологічної небезпеки.

Організація та профілактичні заходи. Вони спрямовані на визначення територій, територій, об'єктів і видів діяльності, екологічно небезпечних для навколишнього природного середовища та здоров'я людей, а також здійснення певних заходів щодо запобігання виникненню екологічної небезпеки. До них належать: 1) органи обліку; 2) реєстрація; 3) експертиза; 4) інформація та прогноз. Крім того, в Україні розвиваються екологічний аудит та екологічне страхування.

Облік і встановлення заходів передбачають виявлення, інвентаризацію та класифікацію небезпечних зон, об'єктів, територій і джерел.

Реєстраційні заходи включають сертифікацію екологічно небезпечних речовин, сертифікацію, підтвердження відповідності, ліцензування та реєстрацію екологічно небезпечних джерел. Якщо виробляється продукція, яка є шкідливою для навколишнього середовища, вона повинна бути сертифікована. У процесі сертифікації видається сертифікат відповідності, який підтверджує відповідність продукції українським стандартам. Така продукція маркується як відповідна встановленим зразкам. Обов'язкова сертифікація продукції передбачена безпосередньо Законом України "Про захист прав споживачів" від 15 грудня 1993 року. Закон України «Про підтвердження відповідності» від 17 травня 2001 року визначає закони та організаційні принципи підтвердження відповідності продукції, систем управління якістю, систем управління навколишнім середовищем, персоналу та спрямований на забезпечення єдності підтвердження національної технічної політики. у наступних сферах відповідності.

Послідовний облік екологічно небезпечних джерел відповідно до чинного законодавства. Екологічно шкідливі види діяльності підлягають ліцензуванню, у тому числі заходи, спрямовані на регулювання та обмеження екологічно шкідливих видів діяльності шляхом запровадження ліцензійної системи та встановлення ліцензійних умов провадження такої діяльності. Екологічне ліцензування регулюється Законом України від 1 червня 2000 р. «Ліцензування певних видів господарської діяльності», постановою Кабінету Міністрів України від 10 серпня 1992 р. No 459 «Положення про порядок видачі документів дозвільного характеру» Спец. Використання природних ресурсів» та інші нормативно-правові акти.

Третю групу організаційно-профілактичних заходів щодо забезпечення екологічної безпеки складають заходи експертної оцінки. Це екологічне обстеження об'єктів і комплексів (у тому числі військових і оборонних споруд), що становлять екологічну загрозу навколишньому природному середовищу, життю і здоров'ю людей, попередні оцінки екологічного впливу цих об'єктів, громадські слухання та громадські обговорення екологічних питань. шкідливої діяльності, яка передбачається реалізувати. Екологічні перевірки таких об'єктів регулюються законами України «Про охорону навколишнього природного середовища» (ст. 27), «Екологічні інспекції» (ст. 7).

Остання група – інформаційно-прогностичні заходи. До них належать прогнозування, планування, моніторинг, повідомлення та інші заходи, які вважаються функціями управління в екологічній сфері.

Регуляція і стимул. Вони являють собою систему правових норм і правил, призначених для регулювання відносин і забезпечення дотримання пріоритетів, норм, стандартів, обмежень та інших вимог у сфері екологічної безпеки. Відповідно до положень чинного законодавства розробляються наступні екологічні стандарти (ст. 32 Закону про охорону навколишнього природного середовища України); екологічні стандарти (ст. 33); екологічні обмеження; правила проектування та експлуатації небезпечних установок, поводження з

ними. з речовинами та джерелами, шкідливими для навколишнього середовища Зачекайте.

Існують певні стимули для забезпечення виконання вимог у сфері екологічної безпеки, яка є невід'ємною частиною економічного механізму у сфері охорони навколишнього середовища. Отже, підприємства, установи, організації та громадяни мають право на податкові, кредитні та інші пільги при здійсненні ефективних заходів і дотриманні вимог екологічної безпеки.

адміністративні заходи. Вони передбачають виконання спеціально уповноваженими органами окремих функцій у сфері забезпечення екологічної безпеки.

Найважливіші положення у цій сфері закріплено в Конституції України, згідно з якою органи виконавчої влади, у тому числі Президент України, відповідають за реалізацію політики у сфері екологічної безпеки. Президент України зобов'язаний вживати заходів щодо забезпечення національної безпеки, у тому числі екологічної, оскільки вона є її невід'ємною частиною. Однією з основних функцій у цій сфері є контрольно-наглядова функція державних установ, метою якої є здійснення контролю та перевірки дотримання підприємствами, установами, організаціями та громадянами вимог природоохоронного законодавства та заходів щодо запобігання екологічним правопорушенням.

Консерваційно-реставраційні заходи. Ці заходи спрямовані на локалізацію проявів екологічної небезпеки, проведення робіт з ліквідації наслідків екологічної небезпеки, визначення правового устрою території за рівнем екологічної небезпеки, визначення статусу осіб, які зазнають наслідків екологічної небезпеки. Наприклад, вони передбачають встановлення правових режимів територій надзвичайної екологічної ситуації.

Ліквідація наслідків надзвичайних ситуацій природного та техногенного характеру передбачає виконання комплексу заходів, у тому числі аварійно-рятувальних та інших невідкладних робіт, що проводяться в надзвичайних

ситуаціях з метою припинення дії факторів ризику, порятунку життя та збереження здоров’я людей, у надзвичайних ситуаціях. місцевості.

заходи безпеки. Їх метою є попередження екологічних злочинів у сферах забезпечення екологічної безпеки, захисту прав людини у безпечному для життя і здоров’я навколишньому середовищі та інших пов’язаних із цим екологічних прав, а також застосування національно-правового впливу до винних осіб, які порушують вимоги та стандарти екологічної безпеки.

Екологічне законодавство передбачає можливість судового захисту цивільних прав, порушених невиконанням вимог екологічної безпеки. Не виключається також необхідна оборона, при якій поведінка має бути правомірною, відповідати змісту та характеру злочину та не суперечити вимогам закону. Зокрема, суди розглядають справи про захист прав громадян на життя, здоров'я та безпечне довкілля, справи про відшкодування шкоди, завданої порушенням вимог і правил екологічної безпеки, про відмову у наданні. Надавати своєчасну, повну та достовірну інформацію про стан навколишнього природного середовища та джерела забруднення, приховувати випадки аварійного забруднення навколишнього природного середовища або фальсифікувати інформацію про екологічний стан чи захворюваність населення [7].

5.3 Поняття екологічної безпеки

Екологічна безпека — стан навколишнього середовища, за якого унеможливлено погіршення екологічної обстановки та гарантовано здоров'я людини.

Приклад подібного визначення безпеки середовища виглядає наступним чином [4]:

1. Сукупність дій, станів і процесів, які прямо чи опосередковано не призводять до заподіяння значної шкоди (або загрози такої шкоди) навколишньому природному середовищу, окремим особам і людині [4].

2. Це стан навколишнього природного середовища, який запобігає погіршенню екологічної обстановки та загрозі здоров'ю людей [1, ст.50].

3. Це стан навколишнього середовища, при якому екологічні умови малоймовірні для погіршення та не становлять небезпеки для здоров'я людини [4].

4. Екологічна безпека – це непорушний стан екологічного комфорту проживання, здатність протистояти загрозам життю, здоров'ю всього живого, і насамперед людини, у тому числі її добробуту, праву на безпечне середовище існування, джерела життєзабезпечення, Природні ресурси [5].

5. Екологічна безпека є невід'ємною частиною національної безпеки і є процесом управління системою національної безпеки, у якому державні та недержавні органи забезпечують екологічну рівновагу та охорону середовища існування населення країни та всієї біосфери. Атмосфера, гідросфера, літосфера і космос, видовий склад флори і фауни, збереження природних ресурсів, здоров'я і життєдіяльності людини, а також віддалені наслідки цього впливу для нинішніх і майбутніх поколінь виключаються [6].

Більш широко, змістовно та глибше розкриває суть розглянутої проблематики таке визначення екологічної безпеки [4]:

1. Це елемент суспільної власності, тобто відповідність умов навколишнього середовища завданням збереження здоров'я населення та забезпечення довгострокового сталого соціально-економічного розвитку, що поєднує інтереси природи і суспільства.

Найбільш повне визначення екологічної безпеки як елемента суспільної власності дає Н.П. Федоренко та К.Г. Гофман:

2. Це сукупність дій і сукупність відповідних заходів, процесів, які однаково забезпечують екологічну рівновагу землі та її різних регіонів, до яких людина може фізично, соціально, економічно та політично адаптуватися без значних втрат [4].

ВИСНОВКИ

У цьому дослідженні ми проаналізували наявну літературу в базі даних в інтернеті, посилаючись на основні моделі прогнозування майбутніх потреб у різних продуктах та послугах протягом останніх років. Аналіз зосереджувався на прогнозах попиту на звичайні та швидкопсувних товарів на малих та середніх підприємствах. Ці моделі сприяють покращенню виробничих планів організацій, зменшенню втрат продукції та підвищенню задоволеності споживачів.

У цій роботі розглядаються деякі важливі елементи для прогнозування потреби в швидкопсувних товарів. У бібліографії ми визначили, що кліматичні фактори, рекламні заходи та сезонні моделі є ключовими елементами, які впливають на зміни короткострокового попиту на продукти харчування.

Витрати на використання цих методів значно зменшиться; це сприятиме їх реалізації. Ми очікуємо, що компанії з використанням комп'ютерного таймшерингу нададуть доступ за номінальною вартістю до банків даних введення-виведення, розбитих на більше сегментів бізнесу, ніж доступні сьогодні.

Нарешті, більшість комп'ютеризованих прогнозів буде стосуватися аналітичних методів, описаних у цій статті. Комп'ютерні додатки будуть переважно в усталених і стабільних продуктах бізнесу. Хоча методи прогнозування поки що використовувалися в основному для прогнозування продажів, вони все частіше будуть застосовуватися для прогнозування маржі, капітальних витрат та інших важливих факторів. Це дозволить прогнозисту витрачати більшу частину часу на прогнозування продажів і прибутку нових продуктів.

СПИСОК ЛІТЕРАТУРИ

1. Кормен Т., Лейзерсон Ч., Рівест Р. Алгоритми: побудова та аналіз, 2001.
2. Окулов З. М. Програмування алгоритмах. - М: БІНОМ. 2002 р.
3. Д.Е. Хлист. Мистецтво програмування, том 3. Пров. З англійської. М.: Видавництво Вільямс, 2007. (та ін видання)
4. Бутч Г. Об'єктно-орієнтований аналіз та проектування з прикладами додатків. Видавництво Вільямс, Санкт-Петербург, 2008
5. Бертран Мейєр. Об'єктно-орієнтоване проектування програмних систем. - М: Російське видання, 2005 р.
6. Лафор Р. Об'єктно-орієнтоване програмування на C++. СПб: Пітер, 2006.
7. Синтес А. Освойте власне об'єктно-орієнтоване програмування за 21 день. Москва4 СПб: Вільямс, 2002.
8. Річард Барас: До теорії інновацій у сфері послуг, -1986, с. -183.
9. Хазієва А. М. Сучасна державна статистика // Тенденції та перспективи розвитку статистичної науки та інформаційних технологій. Збірник наукових статей, -2013, -С. 94-98.
10. Стеценко І.В. Системи моделювання: Навч. поб. [Електронний ресурс, текст]/І.В. Стеценка; Міністерство освіти та науки України, Черкаси. Холдинг техн. ун. – Черкаси: ЧДТУ, 2010. – 399 с.
11. Навроцький, А.А. Основи алгоритмізації та програмування у середовищі Visual C++: метод дослідження. посібник/А. А. Навроцький. - Мінськ: БДУІР, 2014. - 160 с.
12. Шільдт, Г. Базовий курс C++, 3-тє вид. / Г. Шільдт. пров. з англійської мови. - М.: Видавництво Вільямс, 2015. - 624 с.
13. МакКоннелл С. Ідеальний код. Майстер-клас/С. Макконнелл. пров. з англійської мови. - М: Видавництво «Російська редакція», 2010. - 896 с.
14. Документація Visual Studio [електронний ресурс]. Режим доступу: https://docs.microsoft.com/en-us/visualstudio/ide/?view=vs-2017.
15. Стандарт кодування C++ Тодда Хоффа [електронний ресурс]. Режим доступу: http://www.possibility.com/Cpp/c++_coding_standards.pdf

16. Посібник із стилю Google C++ [електронний ресурс]. Режим доступу: https://google.github.io/styleguide/cppguide.html 7. ГОСТ 19.701-90 Схеми алгоритмів, програм, даних та систем. 8. Доманов А.Т. Стандарт підприємства СТП 01-2017/А.Т. Доманов, Н.І. Сорока. - Мінськ: БДУІР, 2017. - 169 с.
17. Введення в мови програмування С та С++ [електронний ресурс] / режим доступу: http://www.intuit.ru/studies/courses/1039/231/info
18. Верифікація ПЗ [Електронний ресурс]/режим доступу: http://www.intuit.ru/studies/courses/1040/209/info
19. Інструменти, алгоритми та структури даних [електронний ресурс]/режим доступу: http://www.intuit.ru/studies/courses/683/539/info
20. Мюссер, Д. C++ і STL: довідник / Д. Мюссер, Дж. Дерг, А. Сейні. - М.: Ред. Будинок "Вільямс", 2010 р.
21. Саттер, Г. Стандарти програмування на C++: серія "C++ In-Depth"/Г. Саттер, А. Александреску. - М.: Ред. Будинок "Вільямс", 2008 рік.
22. Седжвік, Р. Алгоритми в C ++ / Р. Седжвік. - М.: Ред. Будинок "Вільямс", 2010 р.
23. Шільдт, Г. C++: Методи програмування Шільда / Г. Шільдт. - М.: Ред. Будинок "Вільямс", 2008 рік.
24. Девіс, С. C ++ для чайників / С. Девіс. - 6-те вид. - М.: Ред. Будинок "Вільямс", 2010 р.
25. Ліберті, Дж. Вивчіть C++ самостійно за 21 день / Дж. Ліберті, Дж. Бредлі. - 5-те вид. - М.: Ред. Будинок "Вільямс", 2010 р.
26. Страуструп, Б. Програмування на прикладах на C++: принципи та практика / Б. Страуструп. - М.: Ред. Будинок "Вільямс", 2010 р.
27. Шильдт, Г. Повний посібник з C++ / Г. Шильдт. - М.: Ред. Будинок "Вільямс", 2010 р.

ДОДАТКИ

Додаток А(MyForm.h)

```
#pragma once
#include <vector>
#include <string>
#include <fstream>
#include <sstream>
#include <algorithm>
#include <cmath>
#include <msclr\marshal_cppstd.h>

int index = 0;
using namespace std;
vector<vector<string>> product;
namespace ProGnoz {

 using namespace System;
 using namespace System::ComponentModel;
 using namespace System::Collections;
 using namespace System::Windows::Forms;
 using namespace System::Data;
 using namespace System::Drawing;


 vector<vector<string>> getDataFromCSV(string fname) {
      vector<vector<string>> values;
      ifstream ifs(fname.c_str());
      if (!ifs)
      {

           return values;
      }
      string line;
      int count = 0;
      while (ifs.good())
      {
           getline(ifs, line);
           vector<string> temp;
           count++;
           string s = "";
           for (int i = 0; i < line.size(); ++i)
           {
                if (line[i] != ';')
                     s += line[i];
                else
                {
                     temp.push_back(s);
                     s = "";
                }
           }
```

```
            if (s != "")
                temp.push_back(s);
            if (temp.size() != 0)
                values.push_back(temp);
        }
        ifs.close();

        return values;
}
/// <summary>
/// Сводка для MyForm
/// </summary>
public ref class MyForm : public System::Windows::Forms::Form
{
public:
        MyForm(void)
        {
            InitializeComponent();
            //
            //TODO: добавьте код конструктора
            //
        }

protected:
        /// <summary>
        /// Освободить все используемые ресурсы.
        /// </summary>
        ~MyForm()
        {
            if (components)
            {
                delete components;
            }
        }
private: System::Windows::Forms::GroupBox^ groupBox1;
protected:
private: System::Windows::Forms::TextBox^ textBox1;
private: System::Windows::Forms::Button^ button1;
private: System::Windows::Forms::GroupBox^ groupBox2;
private: System::Windows::Forms::Button^ button2;
private: System::Windows::Forms::GroupBox^ groupBox3;
private: System::Windows::Forms::TextBox^ textBox2;
private: System::Windows::Forms::Label^ label2;
private: System::Windows::Forms::ComboBox^ comboBox1;
private: System::Windows::Forms::Label^ label1;
private: System::Windows::Forms::TextBox^ textBox14;
private: System::Windows::Forms::Label^ label14;
private: System::Windows::Forms::TextBox^ textBox13;
private: System::Windows::Forms::Label^ label13;
private: System::Windows::Forms::Label^ label12;
private: System::Windows::Forms::TextBox^ textBox12;
private: System::Windows::Forms::TextBox^ textBox11;
private: System::Windows::Forms::Label^ label11;
private: System::Windows::Forms::TextBox^ textBox10;
```

```
private: System::Windows::Forms::Label^ label10;
private: System::Windows::Forms::TextBox^ textBox9;
private: System::Windows::Forms::Label^ label9;
private: System::Windows::Forms::TextBox^ textBox8;
private: System::Windows::Forms::Label^ label8;
private: System::Windows::Forms::TextBox^ textBox7;
private: System::Windows::Forms::Label^ label7;
private: System::Windows::Forms::TextBox^ textBox6;
private: System::Windows::Forms::Label^ label6;
private: System::Windows::Forms::TextBox^ textBox5;
private: System::Windows::Forms::Label^ label5;
private: System::Windows::Forms::Label^ label4;
private: System::Windows::Forms::TextBox^ textBox4;
private: System::Windows::Forms::TextBox^ textBox3;
private: System::Windows::Forms::Label^ label3;
private: System::Windows::Forms::GroupBox^ groupBox4;
private: System::Windows::Forms::Button^ button3;
private: System::Windows::Forms::TextBox^ textBox15;
private: System::Windows::Forms::Label^ label15;
private: System::Windows::Forms::GroupBox^ groupBox5;
private: System::Windows::Forms::Button^ button4;
private: System::Windows::Forms::ComboBox^ comboBox2;
private: System::Windows::Forms::Label^ label16;
private: System::Windows::Forms::GroupBox^ groupBox6;
private: System::Windows::Forms::Button^ button5;
private: System::Windows::Forms::TextBox^ textBox16;
private: System::Windows::Forms::ListView^ listView1;
private: System::Windows::Forms::ColumnHeader^ columnHeader1;
private: System::Windows::Forms::ColumnHeader^ columnHeader2;
private: System::Windows::Forms::ColumnHeader^ columnHeader3;
private: System::Windows::Forms::ColumnHeader^ columnHeader4;
private: System::Windows::Forms::ColumnHeader^ columnHeader5;
private: System::Windows::Forms::ColumnHeader^ columnHeader6;
private: System::Windows::Forms::ColumnHeader^ columnHeader7;
private: System::Windows::Forms::ColumnHeader^ columnHeader8;
private: System::Windows::Forms::ColumnHeader^ columnHeader9;
private: System::Windows::Forms::ColumnHeader^ columnHeader10;
private: System::Windows::Forms::ColumnHeader^ columnHeader11;
private: System::Windows::Forms::ColumnHeader^ columnHeader12;
private: System::Windows::Forms::ColumnHeader^ columnHeader13;
private: System::Windows::Forms::ColumnHeader^ columnHeader14;
private: System::Windows::Forms::ColumnHeader^ columnHeader15;
private: System::Windows::Forms::ColumnHeader^ columnHeader16;
private: System::Windows::Forms::ColumnHeader^ columnHeader17;
private: System::Windows::Forms::GroupBox^ groupBox7;
private: System::Windows::Forms::Button^ button8;
private: System::Windows::Forms::Button^ button7;
private: System::Windows::Forms::Button^ button6;
private: System::Windows::Forms::Button^ button9;
private: System::Windows::Forms::PictureBox^ pictureBox1;
private: System::Windows::Forms::Button^ button10;

private:
    /// <summary>
```

```
        /// Обязательная переменная конструктора.
        /// </summary>
        System::ComponentModel::Container ^components;

    #pragma region Windows Form Designer generated code
        /// <summary>
        /// Требуемый метод для поддержки конструктора — не
изменяйте
        /// содержимое этого метода с помощью редактора кода.
        /// </summary>
        void InitializeComponent(void)
        {
            this->groupBox1 = (gcnew
System::Windows::Forms::GroupBox());
            this->textBox1 = (gcnew
System::Windows::Forms::TextBox());
            this->button1 = (gcnew
System::Windows::Forms::Button());
            this->groupBox2 = (gcnew
System::Windows::Forms::GroupBox());
            this->button10 = (gcnew
System::Windows::Forms::Button());
            this->button2 = (gcnew
System::Windows::Forms::Button());
            this->groupBox3 = (gcnew
System::Windows::Forms::GroupBox());
            this->textBox14 = (gcnew
System::Windows::Forms::TextBox());
            this->label14 = (gcnew
System::Windows::Forms::Label());
            this->textBox13 = (gcnew
System::Windows::Forms::TextBox());
            this->label13 = (gcnew
System::Windows::Forms::Label());
            this->label12 = (gcnew
System::Windows::Forms::Label());
            this->textBox12 = (gcnew
System::Windows::Forms::TextBox());
            this->textBox11 = (gcnew
System::Windows::Forms::TextBox());
            this->label11 = (gcnew
System::Windows::Forms::Label());
            this->textBox10 = (gcnew
System::Windows::Forms::TextBox());
            this->label10 = (gcnew
System::Windows::Forms::Label());
            this->textBox9 = (gcnew
System::Windows::Forms::TextBox());
            this->label9 = (gcnew
System::Windows::Forms::Label());
            this->textBox8 = (gcnew
System::Windows::Forms::TextBox());
            this->label8 = (gcnew
System::Windows::Forms::Label());
```

```
            this->textBox7                  =                   (gcnew
System::Windows::Forms::TextBox());
            this->label7                    =                   (gcnew
System::Windows::Forms::Label());
            this->textBox6                  =                   (gcnew
System::Windows::Forms::TextBox());
            this->label6                    =                   (gcnew
System::Windows::Forms::Label());
            this->textBox5                  =                   (gcnew
System::Windows::Forms::TextBox());
            this->label5                    =                   (gcnew
System::Windows::Forms::Label());
            this->label4                    =                   (gcnew
System::Windows::Forms::Label());
            this->textBox4                  =                   (gcnew
System::Windows::Forms::TextBox());
            this->textBox3                  =                   (gcnew
System::Windows::Forms::TextBox());
            this->label3                    =                   (gcnew
System::Windows::Forms::Label());
            this->textBox2                  =                   (gcnew
System::Windows::Forms::TextBox());
            this->label2                    =                   (gcnew
System::Windows::Forms::Label());
            this->comboBox1                 =                   (gcnew
System::Windows::Forms::ComboBox());
            this->label1                    =                   (gcnew
System::Windows::Forms::Label());
            this->groupBox4                 =                   (gcnew
System::Windows::Forms::GroupBox());
            this->button3                   =                   (gcnew
System::Windows::Forms::Button());
            this->textBox15                 =                   (gcnew
System::Windows::Forms::TextBox());
            this->label15                   =                   (gcnew
System::Windows::Forms::Label());
            this->groupBox5                 =                   (gcnew
System::Windows::Forms::GroupBox());
            this->button4                   =                   (gcnew
System::Windows::Forms::Button());
            this->comboBox2                 =                   (gcnew
System::Windows::Forms::ComboBox());
            this->label16                   =                   (gcnew
System::Windows::Forms::Label());
            this->groupBox6                 =                   (gcnew
System::Windows::Forms::GroupBox());
            this->button5                   =                   (gcnew
System::Windows::Forms::Button());
            this->textBox16                 =                   (gcnew
System::Windows::Forms::TextBox());
            this->listView1                 =                   (gcnew
System::Windows::Forms::ListView());
            this->columnHeader1               =                 (gcnew
System::Windows::Forms::ColumnHeader());
```

```
            this->columnHeader2              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader3              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader4              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader5              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader6              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader7              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader8              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader9              =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader10             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader11             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader12             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader13             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader14             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader15             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader16             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->columnHeader17             =              (gcnew
System::Windows::Forms::ColumnHeader());
            this->groupBox7                   =             (gcnew
System::Windows::Forms::GroupBox());
            this->button9                    =              (gcnew
System::Windows::Forms::Button());
            this->button8                    =              (gcnew
System::Windows::Forms::Button());
            this->button7                    =              (gcnew
System::Windows::Forms::Button());
            this->button6                    =              (gcnew
System::Windows::Forms::Button());
            this->pictureBox1                 =             (gcnew
System::Windows::Forms::PictureBox());
            this->groupBox1->SuspendLayout();
            this->groupBox2->SuspendLayout();
            this->groupBox3->SuspendLayout();
            this->groupBox4->SuspendLayout();
            this->groupBox5->SuspendLayout();
            this->groupBox6->SuspendLayout();
            this->groupBox7->SuspendLayout();

    (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(t
his->pictureBox1))->BeginInit();
```

```
            this->SuspendLayout();
            //
            // groupBox1
            //
            this->groupBox1->Controls->Add(this->textBox1);
            this->groupBox1->Controls->Add(this->button1);
            this->groupBox1->Location                                =
System::Drawing::Point(12, 12);
            this->groupBox1->Name = L"groupBox1";
            this->groupBox1->Size   =   System::Drawing::Size(392,
55);
            this->groupBox1->TabIndex = 0;
            this->groupBox1->TabStop = false;
            this->groupBox1->Text = L"Завантажити данні з файлу";
            //
            // textBox1
            //
            this->textBox1->Location = System::Drawing::Point(15,
19);
            this->textBox1->Name = L"textBox1";
            this->textBox1->Size   =   System::Drawing::Size(237,
20);
            this->textBox1->TabIndex = 1;
            this->textBox1->Text = L"data.csv";
            //
            // button1
            //
            this->button1->Font              =              (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                  static_cast<System::Byte>(204)));
            this->button1->Location = System::Drawing::Point(275,
19);
            this->button1->Name = L"button1";
            this->button1->Size = System::Drawing::Size(96, 30);
            this->button1->TabIndex = 0;
            this->button1->Text = L"Завантажити";
            this->button1->UseVisualStyleBackColor = true;
            this->button1->Click             +=             gcnew
System::EventHandler(this, &MyForm::button1_Click);
            //
            // groupBox2
            //
            this->groupBox2->Controls->Add(this->button10);
            this->groupBox2->Controls->Add(this->button2);
            this->groupBox2->Controls->Add(this->groupBox3);
            this->groupBox2->Controls->Add(this->textBox2);
            this->groupBox2->Controls->Add(this->label2);
            this->groupBox2->Controls->Add(this->comboBox1);
            this->groupBox2->Controls->Add(this->label1);
            this->groupBox2->Location                                =
System::Drawing::Point(12, 73);
            this->groupBox2->Name = L"groupBox2";
```

```
            this->groupBox2->Size   =   System::Drawing::Size(392,
304);
            this->groupBox2->TabIndex = 1;
            this->groupBox2->TabStop = false;
            this->groupBox2->Text = L"Додати";
            //
            // button10
            //
            this->button10->Font             =             (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button10->Location                                =
System::Drawing::Point(275, 270);
            this->button10->Name = L"button10";
            this->button10->Size = System::Drawing::Size(96, 28);
            this->button10->TabIndex = 8;
            this->button10->Text = L"Графік";
            this->button10->UseVisualStyleBackColor = true;
            this->button10->Click            +=            gcnew
System::EventHandler(this, &MyForm::button10_Click);
            //
            // button2
            //
            this->button2->Font             =             (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button2->Location  =  System::Drawing::Point(6,
270);
            this->button2->Name = L"button2";
            this->button2->Size = System::Drawing::Size(235, 28);
            this->button2->TabIndex = 5;
            this->button2->Text = L"Додати";
            this->button2->UseVisualStyleBackColor = true;
            this->button2->Click            +=            gcnew
System::EventHandler(this, &MyForm::button2_Click);
            //
            // groupBox3
            //
            this->groupBox3->Controls->Add(this->textBox14);
            this->groupBox3->Controls->Add(this->label14);
            this->groupBox3->Controls->Add(this->textBox13);
            this->groupBox3->Controls->Add(this->label13);
            this->groupBox3->Controls->Add(this->label12);
            this->groupBox3->Controls->Add(this->textBox12);
            this->groupBox3->Controls->Add(this->textBox11);
            this->groupBox3->Controls->Add(this->label11);
            this->groupBox3->Controls->Add(this->textBox10);
            this->groupBox3->Controls->Add(this->label10);
            this->groupBox3->Controls->Add(this->textBox9);
            this->groupBox3->Controls->Add(this->label9);
```

```
            this->groupBox3->Controls->Add(this->textBox8);
            this->groupBox3->Controls->Add(this->label8);
            this->groupBox3->Controls->Add(this->textBox7);
            this->groupBox3->Controls->Add(this->label7);
            this->groupBox3->Controls->Add(this->textBox6);
            this->groupBox3->Controls->Add(this->label6);
            this->groupBox3->Controls->Add(this->textBox5);
            this->groupBox3->Controls->Add(this->label5);
            this->groupBox3->Controls->Add(this->label4);
            this->groupBox3->Controls->Add(this->textBox4);
            this->groupBox3->Controls->Add(this->textBox3);
            this->groupBox3->Controls->Add(this->label3);
            this->groupBox3->Location = System::Drawing::Point(6,
69);
            this->groupBox3->Name = L"groupBox3";
            this->groupBox3->Size   =   System::Drawing::Size(380,
195);
            this->groupBox3->TabIndex = 4;
            this->groupBox3->TabStop = false;
            this->groupBox3->Text = L"Продано товарів у місяць";
            //
            // textBox14
            //
            this->textBox14->Location                              =
System::Drawing::Point(305, 162);
            this->textBox14->Name = L"textBox14";
            this->textBox14->Size   =   System::Drawing::Size(69,
20);
            this->textBox14->TabIndex = 23;
            this->textBox14->Text = L"630";
            //
            // label14
            //
            this->label14->AutoSize = true;
            this->label14->Location = System::Drawing::Point(187,
162);
            this->label14->Name = L"label14";
            this->label14->Size = System::Drawing::Size(112, 13);
            this->label14->TabIndex = 22;
            this->label14->Text = L"Дванадцятий місяць";
            //
            // textBox13
            //
            this->textBox13->Location                              =
System::Drawing::Point(305, 136);
            this->textBox13->Name = L"textBox13";
            this->textBox13->Size   =   System::Drawing::Size(69,
20);
            this->textBox13->TabIndex = 21;
            this->textBox13->Text = L"550";
            //
            // label13
            //
            this->label13->AutoSize = true;
```

```
            this->label13->Location = System::Drawing::Point(184,
136);
            this->label13->Name = L"label13";
            this->label13->Size = System::Drawing::Size(117, 13);
            this->label13->TabIndex = 20;
            this->label13->Text = L"Одиннадцятий місяць";
            //
            // label12
            //
            this->label12->AutoSize = true;
            this->label12->Location = System::Drawing::Point(211,
113);
            this->label12->Name = L"label12";
            this->label12->Size = System::Drawing::Size(88, 13);
            this->label12->TabIndex = 19;
            this->label12->Text = L"Десятий місяць";
            //
            // textBox12
            //
            this->textBox12->Location                          =
System::Drawing::Point(305, 110);
            this->textBox12->Name = L"textBox12";
            this->textBox12->Size   =   System::Drawing::Size(69,
20);
            this->textBox12->TabIndex = 18;
            this->textBox12->Text = L"320";
            //
            // textBox11
            //
            this->textBox11->Location                          =
System::Drawing::Point(305, 84);
            this->textBox11->Name = L"textBox11";
            this->textBox11->Size   =   System::Drawing::Size(69,
20);
            this->textBox11->TabIndex = 17;
            this->textBox11->Text = L"470";
            //
            // label11
            //
            this->label11->AutoSize = true;
            this->label11->Location = System::Drawing::Point(209,
84);
            this->label11->Name = L"label11";
            this->label11->Size = System::Drawing::Size(90, 13);
            this->label11->TabIndex = 16;
            this->label11->Text = L"Дев\'ятий місяць";
            //
            // textBox10
            //
            this->textBox10->Location                          =
System::Drawing::Point(305, 56);
            this->textBox10->Name = L"textBox10";
            this->textBox10->Size   =   System::Drawing::Size(69,
20);
```

```
            this->textBox10->TabIndex = 15;
            this->textBox10->Text = L"590";
            //
            // label10
            //
            this->label10->AutoSize = true;
            this->label10->Location = System::Drawing::Point(208,
57);
            this->label10->Name = L"label10";
            this->label10->Size = System::Drawing::Size(89, 13);
            this->label10->TabIndex = 14;
            this->label10->Text = L"Восьмий місяць";
            //
            // textBox9
            //
            this->textBox9->Location                              =
System::Drawing::Point(305, 30);
            this->textBox9->Name = L"textBox9";
            this->textBox9->Size = System::Drawing::Size(69, 20);
            this->textBox9->TabIndex = 13;
            this->textBox9->Text = L"680";
            //
            // label9
            //
            this->label9->AutoSize = true;
            this->label9->Location  =  System::Drawing::Point(209,
30);
            this->label9->Name = L"label9";
            this->label9->Size = System::Drawing::Size(83, 13);
            this->label9->TabIndex = 12;
            this->label9->Text = L"Сьомий місяць";
            //
            // textBox8
            //
            this->textBox8->Location                              =
System::Drawing::Point(109, 162);
            this->textBox8->Name = L"textBox8";
            this->textBox8->Size = System::Drawing::Size(69, 20);
            this->textBox8->TabIndex = 11;
            this->textBox8->Text = L"710";
            //
            // label8
            //
            this->label8->AutoSize = true;
            this->label8->Location  =  System::Drawing::Point(19,
162);
            this->label8->Name = L"label8";
            this->label8->Size = System::Drawing::Size(82, 13);
            this->label8->TabIndex = 10;
            this->label8->Text = L"Шостий місяць";
            //
            // textBox7
            //
```

```
            this->textBox7->Location                                    =
System::Drawing::Point(109, 133);
            this->textBox7->Name = L"textBox7";
            this->textBox7->Size = System::Drawing::Size(69, 20);
            this->textBox7->TabIndex = 9;
            this->textBox7->Text = L"390";
            //
            // label7
            //
            this->label7->AutoSize = true;
            this->label7->Location  =  System::Drawing::Point(22,
133);
            this->label7->Name = L"label7";
            this->label7->Size = System::Drawing::Size(77, 13);
            this->label7->TabIndex = 8;
            this->label7->Text = L"П\'ятий місяць";
            //
            // textBox6
            //
            this->textBox6->Location                                    =
System::Drawing::Point(109, 107);
            this->textBox6->Name = L"textBox6";
            this->textBox6->Size = System::Drawing::Size(69, 20);
            this->textBox6->TabIndex = 7;
            this->textBox6->Text = L"280";
            //
            // label6
            //
            this->label6->AutoSize = true;
            this->label6->Location  =  System::Drawing::Point(6,
104);
            this->label6->Name = L"label6";
            this->label6->Size = System::Drawing::Size(98, 13);
            this->label6->TabIndex = 6;
            this->label6->Text = L"Четвертий місяць";
            //
            // textBox5
            //
            this->textBox5->Location                                    =
System::Drawing::Point(109, 81);
            this->textBox5->Name = L"textBox5";
            this->textBox5->Size = System::Drawing::Size(68, 20);
            this->textBox5->TabIndex = 5;
            this->textBox5->Text = L"540";
            //
            // label5
            //
            this->label5->AutoSize = true;
            this->label5->Location  =  System::Drawing::Point(27,
81);
            this->label5->Name = L"label5";
            this->label5->Size = System::Drawing::Size(76, 13);
            this->label5->TabIndex = 4;
            this->label5->Text = L"Третій місяць";
```

```
            //
            // label4
            //
            this->label4->AutoSize = true;
            this->label4->Location  =  System::Drawing::Point(22,
54);
            this->label4->Name = L"label4";
            this->label4->Size = System::Drawing::Size(81, 13);
            this->label4->TabIndex = 3;
            this->label4->Text = L"Другий місяць";
            //
            // textBox4
            //
            this->textBox4->Location                              =
System::Drawing::Point(109, 54);
            this->textBox4->Name = L"textBox4";
            this->textBox4->Size = System::Drawing::Size(68, 20);
            this->textBox4->TabIndex = 2;
            this->textBox4->Text = L"340";
            //
            // textBox3
            //
            this->textBox3->Location                              =
System::Drawing::Point(109, 27);
            this->textBox3->Name = L"textBox3";
            this->textBox3->Size = System::Drawing::Size(68, 20);
            this->textBox3->TabIndex = 1;
            this->textBox3->Text = L"250";
            //
            // label3
            //
            this->label3->AutoSize = true;
            this->label3->Location  =  System::Drawing::Point(19,
30);
            this->label3->Name = L"label3";
            this->label3->Size = System::Drawing::Size(84, 13);
            this->label3->TabIndex = 0;
            this->label3->Text = L"Перший місяць";
            //
            // textBox2
            //
            this->textBox2->Location                              =
System::Drawing::Point(166, 43);
            this->textBox2->Name = L"textBox2";
            this->textBox2->Size   =   System::Drawing::Size(137,
20);
            this->textBox2->TabIndex = 3;
            this->textBox2->Text = L"Яблуко";
            //
            // label2
            //
            this->label2->AutoSize = true;
            this->label2->Location  =  System::Drawing::Point(51,
46);
```

```
            this->label2->Name = L"label2";
            this->label2->Size = System::Drawing::Size(87, 13);
            this->label2->TabIndex = 2;
            this->label2->Text = L"Назва продукту";
            //
            // comboBox1
            //
            this->comboBox1->FormattingEnabled = true;
            this->comboBox1->Items->AddRange(gcnew    cli::array<
System::Object^   >(4)  {  L"Овочі", L"Фрукти", L"Хімія", L"Техніка"
});
            this->comboBox1->Location                              =
System::Drawing::Point(166, 16);
            this->comboBox1->Name = L"comboBox1";
            this->comboBox1->Size   =   System::Drawing::Size(137,
21);
            this->comboBox1->TabIndex = 1;
            this->comboBox1->Text = L"Фрукти";
            //
            // label1
            //
            this->label1->AutoSize = true;
            this->label1->Location  =  System::Drawing::Point(64,
16);
            this->label1->Name = L"label1";
            this->label1->Size = System::Drawing::Size(74, 13);
            this->label1->TabIndex = 0;
            this->label1->Text = L"Тип продукту";
            //
            // groupBox4
            //
            this->groupBox4->Controls->Add(this->button3);
            this->groupBox4->Controls->Add(this->textBox15);
            this->groupBox4->Controls->Add(this->label15);
            this->groupBox4->Location                              =
System::Drawing::Point(12, 383);
            this->groupBox4->Name = L"groupBox4";
            this->groupBox4->Size   =   System::Drawing::Size(392,
73);
            this->groupBox4->TabIndex = 2;
            this->groupBox4->TabStop = false;
            this->groupBox4->Text = L"Видалити";
            //
            // button3
            //
            this->button3->Font              =              (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button3->Location = System::Drawing::Point(275,
22);
            this->button3->Name = L"button3";
            this->button3->Size = System::Drawing::Size(96, 36);
```

```
            this->button3->TabIndex = 2;
            this->button3->Text = L"Видалити";
            this->button3->UseVisualStyleBackColor = true;
            this->button3->Click            +=            gcnew
System::EventHandler(this, &MyForm::button3_Click);
            //
            // textBox15
            //
            this->textBox15->Location                              =
System::Drawing::Point(105, 31);
            this->textBox15->Name = L"textBox15";
            this->textBox15->Size   =   System::Drawing::Size(147,
20);
            this->textBox15->TabIndex = 1;
            this->textBox15->Text = L"Яблуко";
            //
            // label15
            //
            this->label15->AutoSize = true;
            this->label15->Location  =  System::Drawing::Point(12,
31);
            this->label15->Name = L"label15";
            this->label15->Size = System::Drawing::Size(87, 13);
            this->label15->TabIndex = 0;
            this->label15->Text = L"Назва продукту";
            //
            // groupBox5
            //
            this->groupBox5->Controls->Add(this->button4);
            this->groupBox5->Controls->Add(this->comboBox2);
            this->groupBox5->Controls->Add(this->label16);
            this->groupBox5->Location                              =
System::Drawing::Point(12, 462);
            this->groupBox5->Name = L"groupBox5";
            this->groupBox5->Size   =   System::Drawing::Size(392,
67);
            this->groupBox5->TabIndex = 3;
            this->groupBox5->TabStop = false;
            this->groupBox5->Text = L"Сортування";
            //
            // button4
            //
            this->button4->Font             =             (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button4->Location = System::Drawing::Point(275,
24);
            this->button4->Name = L"button4";
            this->button4->Size = System::Drawing::Size(96, 37);
            this->button4->TabIndex = 2;
            this->button4->Text = L"Сортувати";
            this->button4->UseVisualStyleBackColor = true;
```

```
            this->button4->Click += gcnew
System::EventHandler(this, &MyForm::button4_Click);
            //
            // comboBox2
            //
            this->comboBox2->FormattingEnabled = true;
            this->comboBox2->Items->AddRange(gcnew cli::array<
System::Object^ >(15) {
                L"Тип продукту", L"Назва продукту", L"Перший
місяць",
                L"Другий місяць", L"Третій місяць",
L"Четвертий місяць", L"П\'ятий місяць", L"Шостий місяць", L"Сьомий
місяць", L"Восьмий місяць",
                L"Дев\'ятий місяць", L"Десятий місяць",
L"Одиннадцятий місяць", L"Дванадцятий місяць", L"Прогноз"
            });
            this->comboBox2->Location =
System::Drawing::Point(104, 32);
            this->comboBox2->Name = L"comboBox2";
            this->comboBox2->Size = System::Drawing::Size(148,
21);
            this->comboBox2->TabIndex = 1;
            //
            // label16
            //
            this->label16->AutoSize = true;
            this->label16->Location = System::Drawing::Point(21,
35);
            this->label16->Name = L"label16";
            this->label16->Size = System::Drawing::Size(77, 13);
            this->label16->TabIndex = 0;
            this->label16->Text = L"Сортувати за ";
            //
            // groupBox6
            //
            this->groupBox6->Controls->Add(this->button5);
            this->groupBox6->Controls->Add(this->textBox16);
            this->groupBox6->Location =
System::Drawing::Point(12, 545);
            this->groupBox6->Name = L"groupBox6";
            this->groupBox6->Size = System::Drawing::Size(392,
73);
            this->groupBox6->TabIndex = 4;
            this->groupBox6->TabStop = false;
            this->groupBox6->Text = L"Зберегти данні до файлу";
            //
            // button5
            //
            this->button5->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
```

```
            this->button5->Location = System::Drawing::Point(275,
30);
            this->button5->Name = L"button5";
            this->button5->Size = System::Drawing::Size(96, 33);
            this->button5->TabIndex = 1;
            this->button5->Text = L"Зберегти";
            this->button5->UseVisualStyleBackColor = true;
            this->button5->Click             +=             gcnew
System::EventHandler(this, &MyForm::button5_Click);
            //
            // textBox16
            //
            this->textBox16->Location                                =
System::Drawing::Point(15, 37);
            this->textBox16->Name = L"textBox16";
            this->textBox16->Size   =   System::Drawing::Size(237,
20);
            this->textBox16->TabIndex = 0;
            this->textBox16->Text = L"data.csv";
            //
            // listView1
            //
            this->listView1->Anchor                                  =
static_cast<System::Windows::Forms::AnchorStyles>(((System::Windows
::Forms::AnchorStyles::Top                                           |
System::Windows::Forms::AnchorStyles::Left)
                  | System::Windows::Forms::AnchorStyles::Right));
            this->listView1->Columns->AddRange(gcnew  cli::array<
System::Windows::Forms::ColumnHeader^  >(17) {
                  this->columnHeader1, this->columnHeader2,
                       this->columnHeader3,   this->columnHeader4,
this->columnHeader5,    this->columnHeader6,    this->columnHeader7,
this->columnHeader8,
                       this->columnHeader9,  this->columnHeader10,
this->columnHeader11,  this->columnHeader12,  this->columnHeader13,
this->columnHeader14,
                       this->columnHeader15, this->columnHeader16,
this->columnHeader17
            });
            this->listView1->HideSelection = false;
            this->listView1->Location                                =
System::Drawing::Point(410, 80);
            this->listView1->Name = L"listView1";
            this->listView1->Size   =   System::Drawing::Size(958,
265);
            this->listView1->TabIndex = 5;
            this->listView1->UseCompatibleStateImageBehavior    =
false;
            this->listView1->View                                    =
System::Windows::Forms::View::Details;
            //
            // columnHeader1
            //
            this->columnHeader1->Text = L"#";
```

```
this->columnHeader1->Width = 21;
//
// columnHeader2
//
this->columnHeader2->Text = L"Індекс";
this->columnHeader2->Width = 46;
//
// columnHeader3
//
this->columnHeader3->Text = L"Тип";
//
// columnHeader4
//
this->columnHeader4->Text = L"Назва";
//
// columnHeader5
//
this->columnHeader5->Text = L"1 місяць";
this->columnHeader5->Width = 57;
//
// columnHeader6
//
this->columnHeader6->Text = L"2 місяць";
this->columnHeader6->Width = 56;
//
// columnHeader7
//
this->columnHeader7->Text = L"3 місяць";
this->columnHeader7->Width = 56;
//
// columnHeader8
//
this->columnHeader8->Text = L"4 місяць";
this->columnHeader8->Width = 55;
//
// columnHeader9
//
this->columnHeader9->Text = L"5 місяць";
this->columnHeader9->Width = 57;
//
// columnHeader10
//
this->columnHeader10->Text = L"6 місяць";
this->columnHeader10->Width = 55;
//
// columnHeader11
//
this->columnHeader11->Text = L"7 місяць";
this->columnHeader11->Width = 56;
//
// columnHeader12
//
this->columnHeader12->Text = L"8 місяць";
//
```

```
            // columnHeader13
            //
            this->columnHeader13->Text = L"9 місяць";
            //
            // columnHeader14
            //
            this->columnHeader14->Text = L"10 місяць";
            this->columnHeader14->Width = 64;
            //
            // columnHeader15
            //
            this->columnHeader15->Text = L"11 місяць";
            this->columnHeader15->Width = 64;
            //
            // columnHeader16
            //
            this->columnHeader16->Text = L"12 місяць";
            this->columnHeader16->Width = 65;
            //
            // columnHeader17
            //
            this->columnHeader17->Text = L"Прогноз";
            //
            // groupBox7
            //
            this->groupBox7->Anchor                                    =
static_cast<System::Windows::Forms::AnchorStyles>((System::Windows
::Forms::AnchorStyles::Top                                             |
System::Windows::Forms::AnchorStyles::Left)
                  | System::Windows::Forms::AnchorStyles::Right));
            this->groupBox7->Controls->Add(this->button9);
            this->groupBox7->Controls->Add(this->button8);
            this->groupBox7->Controls->Add(this->button7);
            this->groupBox7->Controls->Add(this->button6);
            this->groupBox7->Location                                  =
System::Drawing::Point(410, 12);
            this->groupBox7->Name = L"groupBox7";
            this->groupBox7->Size   =   System::Drawing::Size(958,
55);
            this->groupBox7->TabIndex = 6;
            this->groupBox7->TabStop = false;
            this->groupBox7->Text = L"Прогнозування методом";
            //
            // button9
            //
            this->button9->Font              =              (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                  static_cast<System::Byte>(204)));
            this->button9->Location = System::Drawing::Point(715,
19);
            this->button9->Name = L"button9";
            this->button9->Size = System::Drawing::Size(214, 30);
```

```
            this->button9->TabIndex = 3;
            this->button9->Text = L"Трігг-Ліча";
            this->button9->UseVisualStyleBackColor = true;
            this->button9->Click            +=            gcnew
System::EventHandler(this, &MyForm::button9_Click);
            //
            // button8
            //
            this->button8->Font             =             (gcnew
System::Drawing::Font(L"Microsoft     Sans     Serif",     8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button8->Location = System::Drawing::Point(481,
19);
            this->button8->Name = L"button8";
            this->button8->Size = System::Drawing::Size(214, 30);
            this->button8->TabIndex = 2;
            this->button8->Text = L"Нормативний метод";
            this->button8->UseVisualStyleBackColor = true;
            this->button8->Click            +=            gcnew
System::EventHandler(this, &MyForm::button8_Click);
            //
            // button7
            //
            this->button7->Font             =             (gcnew
System::Drawing::Font(L"Microsoft     Sans     Serif",     8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button7->Location = System::Drawing::Point(251,
19);
            this->button7->Name = L"button7";
            this->button7->Size = System::Drawing::Size(214, 30);
            this->button7->TabIndex = 1;
            this->button7->Text = L"Екстраполяційний метод";
            this->button7->UseVisualStyleBackColor = true;
            this->button7->Click            +=            gcnew
System::EventHandler(this, &MyForm::button7_Click);
            //
            // button6
            //
            this->button6->Font             =             (gcnew
System::Drawing::Font(L"Microsoft     Sans     Serif",     8.25F,
System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->button6->Location  =  System::Drawing::Point(20,
19);
            this->button6->Name = L"button6";
            this->button6->Size = System::Drawing::Size(214, 30);
            this->button6->TabIndex = 0;
            this->button6->Text = L"Балансовий метод";
            this->button6->UseVisualStyleBackColor = true;
```

```
            this->button6->Click            +=            gcnew
System::EventHandler(this, &MyForm::button6_Click);
            //
            // pictureBox1
            //
            this->pictureBox1->Anchor                                    =
static_cast<System::Windows::Forms::AnchorStyles>(((System::Window
s::Forms::AnchorStyles::Top                                              |
System::Windows::Forms::AnchorStyles::Bottom)
                  | System::Windows::Forms::AnchorStyles::Left)
                  | System::Windows::Forms::AnchorStyles::Right));
            this->pictureBox1->BackColor                                 =
System::Drawing::SystemColors::HighlightText;
            this->pictureBox1->BorderStyle                               =
System::Windows::Forms::BorderStyle::Fixed3D;
            this->pictureBox1->Location                                  =
System::Drawing::Point(410, 351);
            this->pictureBox1->Name = L"pictureBox1";
            this->pictureBox1->Size  =  System::Drawing::Size(958,
257);
            this->pictureBox1->TabIndex = 7;
            this->pictureBox1->TabStop = false;
            this->pictureBox1->Paint          +=          gcnew
System::Windows::Forms::PaintEventHandler(this,
&MyForm::pictureBox1_Paint);
            //
            // MyForm
            //
            this->AutoScaleDimensions = System::Drawing::SizeF(6,
13);
            this->AutoScaleMode                                          =
System::Windows::Forms::AutoScaleMode::Font;
            this->ClientSize = System::Drawing::Size(1380, 627);
            this->Controls->Add(this->pictureBox1);
            this->Controls->Add(this->groupBox7);
            this->Controls->Add(this->listView1);
            this->Controls->Add(this->groupBox6);
            this->Controls->Add(this->groupBox5);
            this->Controls->Add(this->groupBox4);
            this->Controls->Add(this->groupBox2);
            this->Controls->Add(this->groupBox1);
            this->Name = L"MyForm";
            this->Text = L"Product";
            this->groupBox1->ResumeLayout(false);
            this->groupBox1->PerformLayout();
            this->groupBox2->ResumeLayout(false);
            this->groupBox2->PerformLayout();
            this->groupBox3->ResumeLayout(false);
            this->groupBox3->PerformLayout();
            this->groupBox4->ResumeLayout(false);
            this->groupBox4->PerformLayout();
            this->groupBox5->ResumeLayout(false);
            this->groupBox5->PerformLayout();
            this->groupBox6->ResumeLayout(false);
```

```
            this->groupBox6->PerformLayout();
            this->groupBox7->ResumeLayout(false);

     (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(t
his->pictureBox1))->EndInit();
            this->ResumeLayout(false);

        }
    #pragma endregion

     private:  System::Void  button1_Click(System::Object^  sender,
System::EventArgs^ e) {
        string                          str                         =
msclr::interop::marshal_as<std::string>(textBox1->Text);
        product = getDataFromCSV(str);
        index = 0;
        this->listView1->Items->Clear();

        // добавляем строки в listView1
        for (int i = 0; i < product.size(); i++)
        {
            index++;
            //   создадим   объект   ListViewItem   (строку)   для
listView1
            ListViewItem^ listViewItem = gcnew ListViewItem();
            listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
            for (int j = 0; j < product[i].size(); ++j)
                listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
            // добавляем созданный элемент listViewItem (строку)
в listView1
            listView1->Items->Add(listViewItem);
        }
    }

     private:  System::Void  button2_Click(System::Object^  sender,
System::EventArgs^ e) {
        string                          type                        =
msclr::interop::marshal_as<std::string>(comboBox1->SelectedItem-
>ToString());
        string                          name                        =
msclr::interop::marshal_as<std::string>(textBox2->Text);
        string                           m1                         =
msclr::interop::marshal_as<std::string>(textBox3->Text);
        string                           m2                         =
msclr::interop::marshal_as<std::string>(textBox4->Text);
        string                           m3                         =
msclr::interop::marshal_as<std::string>(textBox5->Text);
        string                           m4                         =
msclr::interop::marshal_as<std::string>(textBox6->Text);
        string                           m5                         =
msclr::interop::marshal_as<std::string>(textBox7->Text);
```

```
          string                          m6                          =
msclr::interop::marshal_as<std::string>(textBox8->Text);
          string                          m7                          =
msclr::interop::marshal_as<std::string>(textBox9->Text);
          string                          m8                          =
msclr::interop::marshal_as<std::string>(textBox10->Text);
          string                          m9                          =
msclr::interop::marshal_as<std::string>(textBox11->Text);
          string                          m10                         =
msclr::interop::marshal_as<std::string>(textBox12->Text);
          string                          m11                         =
msclr::interop::marshal_as<std::string>(textBox13->Text);
          string                          m12                         =
msclr::interop::marshal_as<std::string>(textBox14->Text);


          product.push_back({              type,              name,
m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12 });

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
     private:  System::Void  button3_Click(System::Object^  sender,
System::EventArgs^ e) {
          vector<vector<string>> v;
          for (int i = 0; i < product.size(); ++i)
               if                  (product[i][1]                  !=
msclr::interop::marshal_as<std::string>(textBox15->Text))
                    v.push_back(product[i]);
          product = v;

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
```

```
                index++;
                //   создадим   объект   ListViewItem   (строку)   для
listView1
                ListViewItem^ listViewItem = gcnew ListViewItem();
                listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
                for (int j = 0; j < product[i].size(); ++j)
                     listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
                // добавляем созданный элемент listViewItem (строку)
в listView1
                listView1->Items->Add(listViewItem);
          }
     }
     private:  System::Void  button5_Click(System::Object^  sender,
System::EventArgs^ e) {
          ofstream myFile;
          string                          str                          =
msclr::interop::marshal_as<std::string>(textBox16->Text);
          myFile.open(str);
          for (int i = 0; i < product.size(); ++i)
          {
                for (int j = 0; j < product[i].size(); ++j)
                     myFile << product[i][j] << ";";
                myFile << endl;
          }
     }
     private:  System::Void  button4_Click(System::Object^  sender,
System::EventArgs^ e) {
          for (int i = 0; i < product.size() - 1; ++i)
                for (int j = i + 1; j < product.size(); ++j)
                     if (comboBox2->SelectedIndex < 2) {
                          if  (product[i][comboBox2->SelectedIndex]  >
product[j][comboBox2->SelectedIndex])
                          {
                               vector<string> s = product[i];
                               product[i] = product[j];
                               product[j] = s;
                          }
                     }
                     else
                     {
                          if              (stoi(product[i][comboBox2-
>SelectedIndex]) > stoi(product[j][comboBox2->SelectedIndex]))
                          {
                               vector<string> s = product[i];
                               product[i] = product[j];
                               product[j] = s;
                          }
                     }
          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
```

```
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
     private:  System::Void  button6_Click(System::Object^  sender,
System::EventArgs^ e) {
          for (int i = 0; i < product.size(); ++i)
          {
               double sum = 0;
               for (int j = 2; j < 14; ++j)
                    sum += stoi(product[i][j]);
               sum /= 12;
               if (product[i].size() == 15)
                    product[i][14] = to_string(sum);
               else
                    product[i].push_back(to_string(sum));
          }

          index = 0;
          this->listView1->Items->Clear();

          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
     private:  System::Void  button7_Click(System::Object^  sender,
System::EventArgs^ e) {
          for (int i = 0; i < product.size(); ++i)
          {
```

```
                double sum = 0;
                sum     =     2     *     stoi(product[i][12])     -
stoi(product[i][11]);
                if (product[i].size() == 15)
                     product[i][14] = to_string(sum);
                else
                     product[i].push_back(to_string(sum));
           }

           index = 0;
           this->listView1->Items->Clear();

           // добавляем строки в listView1
           for (int i = 0; i < product.size(); i++)
           {
                index++;
                //   создадим   объект   ListViewItem   (строку)   для
listView1
                ListViewItem^ listViewItem = gcnew ListViewItem();
                listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
                for (int j = 0; j < product[i].size(); ++j)
                     listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
                // добавляем созданный элемент listViewItem (строку)
в listView1
                listView1->Items->Add(listViewItem);
           }
     }
     private:  System::Void  button8_Click(System::Object^  sender,
System::EventArgs^ e) {
           for (int i = 0; i < product.size(); ++i)
           {
                int nx = product[i].size() - 2;
                int nx2 = nx * 2;
                double* am2 = new double[nx2];
                for (int j = 0; j < nx2 - 1; j++)
                {
                     if (j % 2 == 0)
                          am2[j]  =  (double)stoi(product[i][j  /  2  +
2]);
                     if (j % 2 != 0)
                          am2[j] = (double)(stoi(product[i][i / 2 +
2]) + stoi(product[i][j / 2 + 1 + 2])) / 2;
                }
                if (product[i].size() == 15)
                     product[i][14] = to_string(am2[nx / 2 - 1 + 2]);
                else
                     product[i].push_back(to_string(am2[nx / 2 - 1 +
2]));
           }

           index = 0;
           this->listView1->Items->Clear();
```

```
          // добавляем строки в listView1
          for (int i = 0; i < product.size(); i++)
          {
               index++;
               //   создадим   объект   ListViewItem   (строку)   для
listView1
               ListViewItem^ listViewItem = gcnew ListViewItem();
               listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
               for (int j = 0; j < product[i].size(); ++j)
                    listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
               // добавляем созданный элемент listViewItem (строку)
в listView1
               listView1->Items->Add(listViewItem);
          }
     }
    private:   System::Void   button9_Click(System::Object^   sender,
System::EventArgs^ e) {
     for (int i = 0; i < product.size(); ++i)
     {
          double sum = 0;
          for (int j = 2; j < 14; ++j)
               sum += stoi(product[i][j]);
          sum /= 12;
          double     delta     =     abs(stoi(product[i][12])     -
stoi(product[i][13]));
          double  xi  =  sum  *  (stoi(product[i][12])+delta*1.005)  +
(1.1 - sum) * stoi(product[i][13]);
          if (product[i].size() == 15)
               product[i][14] = to_string(xi);
          else
               product[i].push_back(to_string(xi));
     }

     index = 0;
     this->listView1->Items->Clear();

     // добавляем строки в listView1
     for (int i = 0; i < product.size(); i++)
     {
          index++;
          // создадим объект ListViewItem (строку) для listView1
          ListViewItem^ listViewItem = gcnew ListViewItem();
          listViewItem->SubItems->Add(gcnew
System::String(to_string(index).c_str()));
          for (int j = 0; j < product[i].size(); ++j)
               listViewItem->SubItems->Add(gcnew
System::String(product[i][j].c_str()));
          // добавляем созданный элемент listViewItem (строку) в
listView1
          listView1->Items->Add(listViewItem);
     }
```

```
    }
    private:  System::Void  button10_Click(System::Object^  sender,
System::EventArgs^ e) {
     pictureBox1->Refresh();
    }
    private: System::Void pictureBox1_Paint(System::Object^ sender,
System::Windows::Forms::PaintEventArgs^ e) {
     if (product.size() != 0) {
          e->Graphics->DrawLine(System::Drawing::Pens::Red,     150,
20, 150, 120);
          for (int i = 0; i < product.size(); i++)
          {
               for (int j = 2; j < product[i].size()-1; ++j) {
                    if (i == 0)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Blue,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))/10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))/10);
                    if (i == 1)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Green,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 2)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Yellow,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 3)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Brown,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 4)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Aqua,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 5)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Black,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 6)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::DarkOrange, (j + 1) * 50, (1000 -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 7)
                         e->Graphics-
>DrawLine(System::Drawing::Pens::Pink,  (j  +  1)  *  50,  (1000  -
(int(atof(product[i][j].c_str()))))  /  10,  (j  +  2)  *  50,  (1000  -
(int(atof(product[i][j + 1].c_str()))))  /  10);
                    if (i == 8)
```

```
                    e->Graphics-
>DrawLine(System::Drawing::Pens::Purple, (j + 1) * 50, (1000 -
(int(atof(product[i][j].c_str()))))) / 10, (j + 2) * 50, (1000 -
(int(atof(product[i][j + 1].c_str()))))) / 10);
                  e->Graphics-
>DrawLine(System::Drawing::Pens::Red, (j + 1) * 50, 120, (j + 2) *
50, 120);

              }

          }
      }
     }
     };
     }
```

Додаток Б(MyForm.cpp)

```
#include "MyForm.h"
#include <Windows.h>

using namespace ProGnoz; // Название проекта
int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int) {
 Application::EnableVisualStyles();
 Application::SetCompatibleTextRenderingDefault(false);
 Application::Run(gcnew MyForm);
 return 0;
}
```