Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Факультет інформаційних технологій
(факультет)

Кафедра системного аналізу та управління
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня бакалавра

Студента Кодоли Яни Олегівни
академічної групи 124-20-1
спеціальності 124 Системний аналіз
на тему: «Аналіз та оптимізація вибору комплексу профілактичних заходів щодо зниження ризику нещасних випадків на виробництві»

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*к. ф. -м. н., доц. Желдак Т.А.*			
розділів:	3			
Інформаційно-аналітичний	*к. ф. -м. н., доц. Коряшкіна Л.С.*			
Спеціальний розділ	*к. ф. -м. н., доц. Коряшкіна Л.С.*			
Рецензент				
Нормоконтролер	*к.ф.-м.н., доц. Хом'як Т.В.*			

Дніпро
2024

ЗАТВЕРДЖЕНО:
завідувач кафедри
Системного аналізу та управління
(повна назва)

_____ *к.т.н., доц. Желдак Т.А.*
(підпис) (прізвище,ініціали)

«____» _____20___року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня *бакалавра*

студенту *Кодолі Я.О.* **академічної групи** *124-20-1*
спеціальності: *124 Системний аналіз*
на тему «*Аналіз та оптимізація вибору комплексу профілактичних заходів щодо зниження ризику нещасних випадків на виробництві»*
затверджену наказом ректора НТУ «Дніпровська політехніка»
від *23.05.2024 р. №469-с*

Розділ	Зміст	Терміни виконання
1. Інформаційно-аналітичний розділ	*Проаналізувати структуру об'єкта дослідження. Визначити предметну область дослідження та проблему, що вирішується.*	10.03.2024 – 01.04.2024
2.Спеціальний розділ	*Розробити і програмно реалізувати алгоритм для оптимізації вибору комплексу профілактичних заходів щодо зниження ризику нещасних випадків на виробництві.*	01.04.2024 – 20.05.2024
3. Тестувально-аналітичний розділ	*Провести експерименти з різними вхідними даними та прикладами для перевірки коректності роботи розробки.*	21.05.2024 – 15.06.2024.

Завдання видано _____ *доц. Желдак Т.А.*
(підпис) (прізвище, ініціали)

Дата видачі: *08.01.2024 р.*

Дата подання до екзаменаційної комісії: _____

Прийнято до виконання _____ Кодола Я. О.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 88 с., 27 рис., 2 табл., 3 додатків, 36 джерел.

Об'єктом дослідження є процес управління ризиками нещасних випадків на виробництві.

Предметом дослідження є ризик небезпеки на виробництві та профілактичні заходи щодо запобігання нещасних випадків.

Метою кваліфікаційної роботи є зниження ризику нещасних випадків на виробництві за рахунок системного аналізу та оптимізації вибору комплексу профілактичних заходів.

Метод дослідження: критичний аналіз літератури, математичне моделювання, програмування з використанням алгоритмів штучного інтелекту та експериментальний аналіз отриманих результатів.

В *інформаційно-аналітичному розділі* було розглянуто актуальність теми, наведено аналіз об'єкту дослідження, проаналізовано існуючі підходи до управління ризиками на виробництві, поставлені задачі.

У *спеціальному розділі* сформовано математичну модель управління ризиками, проаналізовано та обрано алгоритм штучного інтелекту для реалізації логічної моделі, реалізовано логічну модель.

У *тестувально-аналітичному розділі* проведення експериментів та тестів роботи розробленої програми, з конкретним прикладом та різними варіаціями даних, проаналізовано результати.

Практична цінність полягає в тому, що запропонований підхід до оптимізації вибору профілактичних заходів може бути використаний на будь-яких підприємствах, де існує ризик небезпеки, для підвищення рівня безпеки праці та оптимізації витрат на запобіжні заходи.

Ключові слова: ГЕНЕТИЧНИЙ АЛГОРИТМ, МІНІМІЗАЦІЯ, НЕБЕЗПЕЧНИЙ ЧИННИК, ОПТИМІЗАЦІЯ, ПРОФІЛАКТИЧНІ ЗАХОДИ, РИЗИКИ, ШТУЧНИЙ ІНТЕЛЕКТ

ABSTRACT

Explanatory note: 88 p., 27 pictures, 2 tabels, 3 appendixes, 36 sources

Object of research in the work is the process of managing the risks of industrial accidents.

Subject of research: methods of optimizing the selection of a set of preventive measures to reduce the risk of accidents.

The purpose of the research is to analyze and optimize the selection of a set of preventive measures to reduce the risk of accidents at work.

Research methods and equipment: critical analysis of literature, mathematical modeling, programming using artificial intelligence algorithms and experimental analysis of the obtained results.

In *the information-analytical section* the relevance of the topic investigated the analysis of the research object was given, the existing approaches to risk management in production were analyzed, and the tasks were set.

In *the special section* a mathematical model of risk management was formed, an artificial intelligence algorithm was analyzed and selected for the implementation of a logical model, and a logical model was implemented.

In *the experimental and analytical section* of experiments and tests of the developed program, with a specific example and various data variations, the results are analyzed.

The practical value is that the proposed approach to optimizing the choice of preventive measures can be used at production enterprises to increase the level of labor safety and optimize costs.

Keywords: ARTIFICIAL INTELLIGENCE, DANGER FACTOR, GENETIC ALGORITHM, MINIMIZATION, OPTIMIZATION, PREVENTIVE MEASURES, RISKS

ЗМІСТ

ВСТУП

У сучасних умовах виробництва зростають вимоги до безпеки праці та зниження виробничих ризиків. Ризик виробничих нещасних випадків завжди присутній у професійній діяльності людини в будь-якій економічній галузі. Щорічно сотні працівників отримують травми на робочих місцях, що негативно впливає не лише на їхнє здоров'я, але й на економіку країни. Саме тому проведення профілактичних заходів щодо зниження рівня небезпек є надзвичайно важливою для забезпечення здоров’я працівників та економічної ефективності підприємств. Дане питання особливо актуальне для України в умовах інтеграції до європейського ринку.

Мета і завдання дослідження є зниження ризику нещасних випадків на виробництві і раціональне використання фінансових ресурсів за рахунок аналізу та оптимізації вибору профілактичних заходів. Для досягнення цієї мети необхідно провести огляд літератури, визначити актуальність теми, сформулювати задачі дослідження, розробити математичну модель, реалізувати її з використанням алгоритмів штучного інтелекту, проаналізувати результати, розробити відповідні рекомендації.

Об’єкт дослідження процес управління ризиками нещасних випадків на виробництві.

Предмет дослідження ризик небезпеки на виробництві та профілактичні заходи щодо запобігання нещасних випадків.

Методи дослідження критичний аналіз літератури, математичне моделювання, програмування з використанням алгоритмів штучного інтелекту та експериментальний аналіз отриманих результатів.

В даній роботі поставлена задача провести огляд літератури, ознайомитися з існуючими методами та підходами до управління ризиками на виробництві, визначити актуальність теми. На основі цього сформулювати оптимізаційну задачу вибору сукупності запобіжних заходів задля зниження

ризику небезпек до прийнятного рівня, побудувати і реалізувати математичну модель задачі. При цьому обрання технології розробки профілактичних заходів проводиться аргументовано з точки зору їх актуальності та ефективності.

Важливою частиною роботи є аналіз результатів проведених обчислювальних експериментів та формулювання висновків, що дозволять оптимізувати систему управління безпекою на виробництві.

Наукова новизна отриманих результатів. В роботі запропоновано новий підхід до оптимізації вибору профілактичних заходів, що поєднує зниження ризиків нещасних випадків та мінімізацію фінансових витрат. Використання математичного моделювання та алгоритмів штучного інтелекту дозволило досягти оптимального балансу між безпекою та витратами, що відрізняється від відомих раніше підходів.

1 ІНФОРМАЦІЙНО АНАЛІТИЧНИЙ

1.1 Поняття та визначення нещасних випадків

Раніше джерелами небезпеки для людини були природні явища, представники біологічного світу та різні природні процеси. З розвитком цивілізації рівень загрози зріс, і на сучасному етапі антропогенні небезпеки (створені людиною) вийшли на перший план. Причини виникнення небезпек або нещасних випадків полягають у збігу обставин, які призводять до прояву небезпеки та виникнення негативних наслідків: нервових потрясінь, травм, хвороб, а іноді й втрати життя.

Нещасний випадок – це раптова, обмежена в часі подія або непередбачуваний вплив, що відбуються під час виконання працівником своїх професійних обов'язків або в зв'язку з ними, результатом якої є травмування, пошкодження здоров'я або втрата життя працівника. Нещасні випадки мають серйозні наслідки як для самих працівників, так і для підприємства в цілому, оскільки вони спричиняють значні втрати та відповідальність за безпеку своїх співробітників [1].

Класифікувати нещасні випадки можна за різноманітними критеріями, які дозволяють більш точно визначити їхні характеристики та наслідки. До таких критеріїв належать:

– *За тяжкістю наслідків*: легкі, середньої тяжкості та тяжкі нещасні випадки. Легкі випадки призводять до незначних травм, середньої тяжкості випадки потребують медичної допомоги та періоду відновлення, тоді як важкі випадки можуть призвести до серйозних травм або втрати життя працівника;

– *За типом події:* механічні, термічні, електричні, хімічні та психологічні травми. Механічні травми можуть включати порізи, переломи та удари, термічні травми – опіки або обмороження, електричні травми –

ураження електричним струмом, хімічні травми – отруєння або опіки від хімічних речовин, а психологічні травми – стресові розлади та інші психологічні наслідки;

– *За місцем події* на робочому місці, під час транспортування, на виробничих територіях. Це дозволяє виявити, де найчастіше трапляються нещасні випадки і які місця потребують більшої уваги в плані безпеки [2-6].

Оцінка нещасних випадків включає декілька етапів спрямованих на глибоке розуміння причин і обставин події. Такі етапи:

– *Визначення небезпечних чинників:* ідентифікація конкретних елементів або умов, які могли спричинити нещасний випадок;

– *Аналіз ймовірності настання події:* ретельне вивчення умов, за яких стався нещасний випадок;

– *Оцінка наслідків:* визначення ступеня шкоди, завданої здоров'ю працівника, а також впливу події на виробничий процес і загальну безпеку підприємства;

– *Розробка та впровадження профілактичних заходів:* розробка конкретних дій, спрямованих на усунення або мінімізацію небезпечних чинників, щоб запобігти таких випадків у майбутньому [4].

Небезпечні чинники є основними причинами виникнення нещасних випадків і можуть мати різний характер та наслідки, зокрема фізичні, хімічні, біологічні, психологічні тощо. Важливо розуміти, що ефективне управління небезпечними чинниками є критичним для запобігання нещасних випадків і забезпечення безпечних умов праці. Дуже часто підприємства й самі працівники зневажають своєю безпекою під тиском роботи й забувають про ці фактори. Саме тому ця тема є дуже актуальною й досліджуваною на сьогоднішній день.

Аналізу та управлінню, так званими безпечними ризиками, або ж просто ризиками на нещасними випадками, приділяється багато уваги, наприклад, в статті *«Safety Management and Risk Assessment in Industrial*

Operations» International Journal of Industrial Engineering and Management, 2021, автори досліджують методи оцінки ризиків і розробку профілактичних заходів для зниження виробничих небезпек. У роботі акцентується увага на необхідності оцінки ймовірності виникнення ризиків та тяжкості їх наслідків за допомогою як якісних, так і кількісних методів, включаючи матриці ризиків та інші інструменти для надання числових значень ймовірності та серйозності ризиків. Особлива увага приділяється розробці та впровадженню профілактичних заходів, а також формуванню культури безпеки на підприємстві, що включає залучення керівництва та працівників до процесу управління ризиками, а також постійний моніторинг і контроль ефективності впроваджених заходів [6].

Ефективний контроль за нещасними випадками на підприємстві передбачає постійний моніторинг та контроль усіх можливих небезпечних чинників, ретельний аналіз ризиків та потенційних наслідків, а також впровадження профілактичних заходів для зниження ймовірності їх настання. Це включає регулярні інспекції, навчання персоналу, використання засобів індивідуального захисту та модернізацію обладнання [7-8]

Про впровадження безпеки детально розкрито в статті *«Fostering a Safety Culture in Manufacturing Industry through Safety Behavior: A Structural Equation Modelling Approach»*. Автори розказують про проведене дослідження, в рамках якого опитано понад 340 робітників, і зроблено висновки, що великий вплив на розвиток культури безпеки має створення більш безпечних умов праці. Автори дослідження підкреслюють, що безпечна поведінка може суттєво мінімізувати ризики та наслідки нещасних випадків, сприяючи створенню безпечного робочого середовища. Це дослідження підтверджує важливість інтеграції культури безпеки на підприємстві для забезпечення здоров’я та безпеки працівників. [9]

1.2 Небезпечні чинники

Небезпечний чинник визначається як подія, об’єкт або умова виробничого середовища, які можуть спричинити нещасний випадок, завдати шкоди здоров’ю чи життю працівників, призвести до матеріальних збитків підприємства або мати інші негативні наслідки.

Дослідження, аналіз небезпечних чинників та методів їх управління є важливою складовою забезпечення безпеки на виробництві. Дуже детально про роботу та організацію небезпечних чинників розказує *James T. Tweedy (Jim)* в своїй книзі *«Introduction to Hazard Control Management: A Vital Organizational Function»*. Книга пояснює, як перевірені принципи управління та лідерства можуть покращити контроль за небезпекою та ефективність управління безпекою в організаціях усіх типів і розмірів. Автор підкреслює значення систематичного підходу до ідентифікації, оцінки та контролю небезпечних чинників, що є ключовим для запобігання нещасним випадкам та забезпечення безпечних умов праці. [10].

У кожному з випадків, джерело небезпеки або так званий чинник виявити можна по різному, іноді це щось очевидне, наприклад, робота з високими температурами в незахищеному знаряджені, а іноді майже неможливо, наприклад, якщо справа йде з стресовою ситуацією співробітників. Саме тому для більш кращого розуміння й знаходження всіх причин та проявів небезпечних факторів їх розлили за різними класифікаціями.

Основні класифікації небезпечних чинників за деякими критеріями:

– *Фізичні небезпечні чинники*:, термічні (високі, низькі температури), електричні (електричний струм, електромагнітні), радіаційні тощо;

– *Механічні небезпечні чинники*: рухомі конструкції, гострі предмети, падаючі елементи;

– *Хімічні небезпечні чинники:* токсичні речовини (отрути, гази, рідини), корозійні речовини (кислоти, луги);

– *Психосоціальні небезпечні чинники:* стресові ситуації (висока відповідальність, тиск), психологічне напруження (конфлікти, психологічне навантаження);

– *Людський фактор*: зневажання людиною своєї безпеки, помилки при проектування та експлуатації, порушення правил техніки безпеки. [2-4]

Більш детально про найчастіше зустрічні та існуючі небезпечні чинники на виробництві можна дізнатися з численних досліджень і публікацій. Зокрема, цікаві та інформативні статті на цю тему представлені в *«Top 5 Safety Hazards in the Manufacturing Industry», July 2022 – Evotix* [11] та *«Top 10 Most Common Safety Hazards In Manufacturing» від Adam Lynch April, 2024* [12].

Нещасний випадок є результатом дії одного або кількох небезпечних чинників. Це підкреслює важливість своєчасної і правильної оцінки небезпечних чинників для запобігання нещасним випадкам. Наприклад, несправне обладнання може спричинити травмування працівника, а недостатня підготовка або обізнаність працівника може призвести до небезпечної поведінки, що загрожує здоров'ю і безпеці не тільки його самого, але й колег.

Для оцінки небезпечних чинників використовуються методи аналізу ризиків, які включають оцінку ймовірності настання події та серйозності її наслідків. Більш детально ці аспекти розглядаються в розділі 1.3. Ризики оцінюються для того, щоб визначити найбільш критичні небезпечні чинники та розробити відповідні заходи для їх усунення або мінімізації. Це дозволяє підприємствам і керівництву приймати обґрунтовані рішення щодо безпеки праці та профілактичних заходів.

Для запобігання і зменшення ймовірності виникнення небезпечних ситуацій необхідно впроваджувати профілактичні заходи, які детально

описані в розділі 1.4. Профілактичні заходи включають як технічні рішення (модернізація обладнання, встановлення захисних бар'єрів), так і організаційні (проведення навчань, впровадження правил безпеки), що спрямовані на створення безпечних умов праці і зниження ризиків для здоров’я і життя працівників [1-2, 5-8].

1.3 Поняття та види ризиків

Ризик є одним з найголовніших аспектів будь-якої діяльності, як у минулому так і сьогодні. У контексті виробничої небезпеки, ризик є як можливість визначення виникнення нещасного випадку, ситуації, що можуть призвести до шкоди здоров’ю, життю людини чи інших наслідків. Виробничий ризик, зокрема, є ймовірністю настання події з негативними наслідками, що можуть призвести до шкоди здоров’ю, життю працівників або пошкодження обладнання чи інфраструктури.

Поняття ризику охоплює ймовірність настання події, яка може мати негативний вплив або наслідки для процесу, майна або події. Ризик складається з двох основних компонентів: ймовірності настання події з негативними наслідками та величини можливих наслідків цієї події. Іншими словами, для оцінки ризику використовуються два параметри: ймовірність настання події та серйозність її наслідків [13-16].

Значення ефективного управління ризиками стає особливо очевидним у контексті виробничих підприємств, де правильне оцінювання та управління ризиками можуть запобігти значним фінансовим втратам та забезпечити безпеку працівників. У своїй статті *«Strategic Risk Management» Holger Sommerfeld* розглядає методи стратегічного управління ризиками, підкреслюючи важливість інтеграції управління ризиками в загальну стратегію підприємства для досягнення його довгострокових цілей [17].

Як зазначено вище, управління ризиками грає одну з основних ролей у роботі з ними. Однак, для правильного управління необхідно спочатку визначити, зрозуміти і оцінити ризик.

Кожен ризик можна класифікувати за різними ознаками, що дозволяє більш ефективно управляти ними та розробляти відповідні заходи для їх мінімізації. Зокрема, ризики можна класифікувати:

– *За причиною виникнення:* техногенні, природні, соціальні;
– *За наслідками:* матеріальні, екологічні, соціальні;
– *За ймовірністю настання:* низький, середній, високий.

Ключовим етапом роботи з ризиками є їх оцінка, вона включає в себе ідентифікацію потенційних небезпечних чинників, аналіз ймовірностей їх виникнення та можливих наслідків. Оцінка поділяється на якісну та кількісну.

Якісний метод базується на досвіді та знаннях експерта, може бути сформульовано просто у термінах, таких як: «високий ризик», «середній ризик», «низький ризик» тощо. Перевагою цього методу є можливість швидко оцінити ризики, а недоліком може бути суб’єктивність оцінки.

Кількісний метод оцінювання базується на математичних розрахунках. Даний метод вимагає більшої кількості інформації, забирає більше часу, але його найголовнішою перевагою є більш точні результати.

При формуванні оцінки ризиків береться до уваги ймовірність настання небезпечного чинника та тяжкість його наслідків. Ймовірність за нормами це число від 0 до 1 або ж число вказане у відсотках. Шкала оцінки серйозності наслідків визначається відповідно до підприємства, це може бути число як від 1 до 10, так і від 1 до 100. В залежності від шкали для оцінки тяжкості наслідків залежить й шкала оцінки ризиків, так як ризик дорівнює ймовірність помножена на тяжкість наслідків. Максимально допустиме значення ризику також кожне підприємство визначає для себе окремо [15, 18-22].

Отже, оцінка та визначення ризиків є дуже важливим етапом для їх розуміння. Даний факт може підтвердити й навчальний посібник *«Ризик-менеджмент»* авторства *Калініченко З.Д.*. Авторка. у своєму посібнику надає детальний огляд основних концепцій та методів управління ризиками, що застосовуються в різних галузях економіки, включаючи виробничий сектор. Важливим аспектом, розглянутим у посібнику, є підходи до оцінки ризиків та розробка ефективних стратегій їх мінімізації. Дані підходи, що є критично важливими для забезпечення безпеки на виробництві, оскільки дозволяють своєчасно виявляти потенційні загрози та запобігати негативним наслідкам. [16].

Для зменшення ризику настання небезпечного чиннику проводять профілактичні заходи., більш детально про це див. розділ 1.4. Профілактичні заходи включають різні дії, спрямовані на зменшення ймовірності та серйозності наслідків потенційних нещасних випадків.

1.4 Поняття та визначення профілактичних заходів

Вирішальну роль у створені безпечного робочого середовища відіграють профілактичні заходи. Вони спрямовані на попередження виникнення небезпечних ситуацій та мінімізацію їхніх наслідків.

Означенням профілактичних заходів є комплекс дій метою яких є попередження виникнення небезпечних чинників, зниження ймовірностей їх настання та мінімізація їхніх негативних наслідків. Вони є основним елементом управлінням безпечної праці та ризиками на виробництві.

Профілактичні заходи бувають різних видів та класифікуються за різними критеріями, серед яких:

– *Інженерні заходи*: оновлення технічного обладнання, встановлення захисних огорож, автоматизацію виробничих процесів та інші технічні рішення;

– *Навчальні та підготовчі заходи*: проведення тренінгів з техніки безпеки, навчання працівників правильному користуванню обладнання організацію навчальних семінарів та практичних занять;

– *Організаційні заходи:* регулярні перевірки й аудити безпеки, встановлення чітких правил й інструкцій розробку планів евакуації та дій у надзвичайних ситуаціях.

Проведення та здійснення різних профілактичних заходів передбачено не лише внутрішніми правилами підприємства, а й регулюється на державному рівні. Зокрема, існує перелік заходів та засобів охорони праці, витрати на здійснення та придбання яких включаються до валових витрат підприємств. Цей перелік затверджений постановою Кабінету Міністрів України від 27 червня 2003 р. №994 [23-24], що підкреслює важливість забезпечення безпеки праці та дотримання стандартів охорони праці на національному рівні.

Профілактичні заходи впливають безпосередньо на ймовірність настання та тяжкість наслідків небезпечного чинника. Для кожного чинника можуть бути задіяно як один так і декілька заходів. Наприклад, для зменшення ризику механічних травм можна одночасно оновити обладнання, встановити захисні огорожі та провести навчання з техніки безпеки.

Окрім того важливу роль для визначення, який саме комплекс профілактичних заходів буде проведено грає його вартість. Кожен захід має свою ціну та свій вплив на ймовірність та тяжкість настання небезпечного чинника. Іноді може бути набагато вигіднішим й раціональнішим буде проведення декількох заходів з меншим впливом ніж один дорогий з більшим. Таким чином, підприємства мають балансувати між витратами та ефективністю заходів безпеки [23-25].

Для глибшого розуміння різних видів та цілей профілактичних заходів варто звернутися до статті *Leon Altomonte «A Guide to Risk Reduction»*. У цій статті автор детально аналізує методи та процеси зниження ризиків,

наголошуючи на важливості комплексного підходу до управління ризиками на підприємствах. *Altomonte* розглядає різні інженерні, організаційні та навчальні заходи, які можна застосувати для мінімізації ризиків. Автор також підкреслює необхідність регулярного моніторингу та оцінки ефективності впроваджених заходів, що дозволяє своєчасно вносити корективи та підвищувати загальний рівень безпеки на виробництві. Крім того, стаття містить практичні рекомендації щодо впровадження профілактичних заходів, які допоможуть підприємствам зменшити ймовірність нещасних випадків та знизити їх наслідки [26].

Важливо зазначити, що кожне підприємство визначає для себе пріоритетні показники ефективності профілактичних заходів. Для когось більш важливим є зниження показників ризиків, незалежно від витрат, для іншого підприємства є важливим мінімізувати як витрати так і ризики. Вибір комплексу заходів залежить від стратегічних цілей підприємства та його фінансових можливостей.

Після проведення профілактичних заходів також визначається ефективність їх проведення. Це включає аналіз змін у показниках безпеки, оцінку зниження ризиків та порівняння фактичних результатів із запланованими. Така оцінка дозволяє підприємству коригувати свою стратегію управління ризиками та підвищувати загальний рівень безпеки на виробництві. В результаті, постійний моніторинг та аналіз ефективності заходів сприяють створенню більш безпечного робочого середовища і оптимізації витрат [23-25].

1.5 Постановка задачі

Задача: Для підприємства певної галузі:

1. Проаналізувати джерела виникнення небезпеки,

2. Запропонувати можливі профілактичні заходи щодо зниження рівня небезпеки й оцінити їх результативність у сенсі або зниження ризику виникнення небезпеки, або зменшення настання небезпечної події;

3. Оцінити матеріальні витрати на проведення кожного із запропонованих заходів;

4. На основі отриманої інформації про рівень ризику, ймовірність настання небезпечної події, оцінки результативності запобіжних заходів та витрат на них розробити комплекс профілактичних заходів для зниження ризику нещасного випадку на підприємстві до прийнятного рівня за умови мінімізації фінансових витрат.

Поставлена задача реалізує відомий у відповідній науковій літературі комплексний підхід до управління ризиками та безпекою на виробництві, що включає аналіз ризиків, розробку та впровадження профілактичних заходів для зменшення ймовірності нещасних випадків та мінімізацію їх наслідків.

2 СПЕЦІАЛЬНИЙ РОЗДІЛ

2.1 Математичне моделювання оптимального вибору профілактичних заходів

Для побудови математичної моделі задачі оптимізації вибору профілактичних заходів щодо зниження ризику настання певної небезпечної події введемо параметри:

Параметри:

N − кількість небезпечних чинників;

A − кількість профілактичних заходів;

$R_{\text{макс}}$ – максимальне припустиме значення ризиків;

$C_{\text{макс}}$ – максимальне припустиме значення витрат;

Для небезпечного чинника, що може бути викликаний небезпекою, визначаються такі змінні:

$P_{\text{поч}_i}$ – початкове значення ймовірності, в діапазоні $0 < P_{\text{поч}} < 1$;

$K_{\text{поч}_i}$– початкове значення ступеня тяжкості, в діапазоні $1 < P_{\text{поч}} < 10$ визначеному за експертною шкалою оцінки тяжкості небезпечного чинника;

$R_{\text{поч}_i}$ – початкове значення можливих ризиків, яке розраховується за формулою $R_{\text{поч}} = P_{\text{поч}} \cdot K_{\text{поч}}$;

$R_{\text{доп}_i}$– рівень допустимого значення ризику;

C_i − загальні витрати на проведення профілактичних заходів;

$P_{\text{кін}_i}$– кінцеве значення ймовірності ризику;

$K_{\text{кін}_i}$– кінцеве значення ступеня тяжкості;

$R_{\text{кін}_i}$– кінцеве значення можливого ризику;

L_i – ефективність витрат на одну ум. грошову од. визначається за формулою: $L_i = \frac{R_{\text{поч}_i} - R_{\text{кін}_i}}{C_i}$.

Для профілактичного заходу, що знижує ризик від небезпечного чинника, що може бути викликаний небезпекою визначаються такі змінні:

P_{ai} – значення, що показаю на скільки відсотків знижується ймовірність небезпечного чинника після проведення профілактичного заходу;

T_{ai} – значення, що показує на скільки відсотків знижується тяжкість наслідків після проведення профілактичного заходу;

R_{ai} – кінцевий рівень ризику після проведення профілактичного заходу;

C_{ai} – витрати на проведення профілактичного заходу;

x_{ai} – шукана змінна, яка буде мати два стани 0 або 1.

$$x_{ai} = \begin{cases} 1, \text{якщо } a\text{ий профілактичний захід проводиться} \\ 0, \text{якщо } a\text{ий профілактичний захід НЕ проводиться} \end{cases}$$

Математична модель

Цільова функція:

$$Z = \sum_{k=1}^{i} C_i = \sum_{k=1}^{i} \sum_{l=1}^{j} C_{ai}\, x_{ai} \to min$$

за умов, що $R_{\text{кін}_i} \geq R_{\text{доп}_i}$ виконуються умови (2)–(6), при $R_{\text{кін}_i} \geq R_{\text{доп}_i}$, всі значення x_{ai} дорівнюють 0 для усіх $A = \{a_1, \dots a_j\}$

$$P_{\text{кін}_i} = max\left\{0, P_{\text{поч}_i}(1 - \sum_{l=1}^{j} P_{ai} x_{ai})\right\}$$

$$K_{\text{кін}_i} = max\left\{1, K_{\text{поч}_i}(1 - (K_{\text{поч}_i} - 1) \sum_{l=1}^{j} T_{ai} x_{ai}\right\}$$

$$R_{\text{кін}_i} = P_{\text{кін}_i} \cdot K_{\text{кін}_i}$$

$$R_{\text{кін}_i} \leq R_{\text{доп}_i}$$

$$\sum_{k=1}^{i} R_{\text{кін}_i} \leq R_{\text{макс}}$$

$$x_{ai} \in \{0,1\}$$

В якості цільової функції розглядається мінімізація витрат на комплекс профілактичних заходів з виконанням умови, що кінцеві ризики мають бути менші чи дорівнювати допустимим.

2.2 Середовище

Для розробки та моделювання програмного забезпечення для оптимізаційної задачі вибору комплексу профілактичних заходів щодо зниження ризику нещасних випадків на виробництві, було обрано мову програмування Python у поєднанні з інтегрованим середовищем розробки PyCharm.

Python володіє рядом переваг, таких як простота вивчення та використання, велика кількість наявних бібліотек для обробки даних, наукових обчислень та оптимізації, а також активною спільнотою користувачів, що забезпечує швидкий доступ до новітніх розробок та розв'язань.

При написані програми було використано бібліотеки:

– *numpy* − бібліотека для роботи з масивами та обчисленням математичних функцій;

– *mealpy* − бібіліотека для використання метаевристичних алгоритмів оптимізації;

– *.customtkinter* – бібліотека основана на *tkinter*, забезпечує можливість використання нових, сучасних віджетів.

2.3 Вибір технології

Для вирішення задач лінійної оптимізації існує безліч методів, алгоритмів та бібліотек. Проте, на даний момент, штучний інтелект (ШІ) стає все більш актуальним та ефективним інструментом для вирішення саме таких задач. Саме це й зумовило його вибір як основної технології для дослідження в цій роботі.

Поняття та алгоритми ШІ, що використовуються для вирішення оптимізаційних задач, надзвичайно різноманітні. Тому важливо чітко визначити, який з них буде найбільш підходящим для конкретної задачі. Було розглянуто спочатку декілька методів:

– *Машинне навчання* використовується для навчання алгоритмів на даних, щоб робити прогнози або приймати рішення.

– *Евристичні методи* використовують знання та досвід, щоб швидше знаходити прийнятні рішення без гарантії їхньої оптимальності.

– *Метаевристичний алгоритм* робить компроміс між евристичними підходами та точними алгоритмами, оскільки вони шукають рішення шляхом повторного вдосконалення початкового рішення [27-28].

На основі аналізу представлених алгоритмів, їх особливостей та можливостей було обрано саме метаевристичний алгоритм для вирішення задачі лінійної оптимізації. Даний метод швидший, ніж інші алгоритми, і може впоратися з більшими випадками проблеми.

Метаевристичні методи − це загальна назва для методів, які використовують ітеративний підхід для покращення рішення, не обмежуючись певними правилами або методами. Це клас алгоритмів оптимізації, які зазвичай моделюють природні процеси або механізми поведінки для пошуку оптимальних рішень у просторі можливих розв’язків.

Для вирішення саме задачі лінійної оптимізації було розглянуто декілька методів:

– *Метод відпалу (Simulated Annealing, SA)* − моделює процес охолодження речовини, дозволяючи іноді приймати гірші рішення з певною ймовірністю. Даний алгоритм може бути хорошим вибором для задач з складними просторами пошуку, де важливо уникнути зупинки в локальних оптимумах.

– *Генетичний алгоритм (Genetic Algorithm, GA)* − використовує принципи еволюції для пошуку оптимального рішення. Генетичні алгоритми можуть бути ефективними для задач, де важливо уникати повторного розміщення елементів, наприклад, в задачах упаковки.

– *Алгоритм рою часток (Particle Swarm Optimization, PSO)* − імітує поведінку зграї птахів або комах, які шукають їжу. Алгоритм рою часток може бути ефективним для задач, де цільова функція є складною і не диференційованою.

Кожен з цих алгоритмів має свої переваги й недоліки. Вибір конкретного алгоритму залежить від конкретних умов задачі та досвіду в їхньому використанні. За даних умов, що існує велика кількість змінних, багато складних умов та цільова функція, яку необхідно мінімізувати було зроблено вибір у користь саме генетичного алгоритму [29].

Генетичний алгоритм (ГА) – це метод, що використовує принципи еволюції для пошуку оптимального рішення задачі. Він працює з популяцією рішень, які називаються «хромосомами». Кожна хромосома кодує можливе рішення за допомогою набору генів. ГА ґрунтується на таких операціях еволюції, як відбір, схрещування та мутація.

Якщо розглянути генетичний алгоритм більш детально, то він складається з восьми кроків: ініціалізація, відбір, схрещування, мутація, оцінка, відбір наступного покоління, повторення, вибір оптимального рішення. Кожен крок відіграє свою важливу роль в роботі алгоритму [30].

1. *Ініціалізація* – створює початкову популяцію хромосом. Хромосоми створюються або випадковим чином або на основі евристичних методів.
2. *Відбір* – з популяції обираються хромосоми з кращою пристосованістю для репродукції. Методів відбору дуже багато: рулетка, турнірний відбір тощо.
3. *Схрещування* – «батьківські» хромосоми «схрещуються» для створення нових хромосом. Деякі методи схрещування: одноточкове, двоточнове, багатоточкове.
4. *Мутація* − з певною ймовірністю в нові хромосоми вносяться випадкові зміни (мутації).
5. *Оцінка* − кожній новій хромосомі присвоюється значення пристосованості, яке відображає її «якість» у вирішенні задачі.
6. *Відбір наступного покоління* − з нових та старих хромосом формується наступне покоління популяції. Зазвичай старі хромосоми з гіршою пристосованістю замінюються новими хромосомами.
7. *Повторення* − кроки 2-6 повторюються до тих пір, поки не буде знайдено оптимальне рішення або не буде досягнуто заданого критерію зупинки. Критерій зупинки може включати максимальну кількість ітерацій, мінімальне покращення пристосованості або досягнення заданого значення пристосованості.
8. *Вибір оптимального рішення* − після зупинки алгоритму хромосома з найкращою пристосованістю в популяції вважається оптимальним рішенням задачі.

Даний алгоритм − це потужний та універсальний інструмент для оптимізації, який може ефективно вирішувати широкий спектр задач. Його гнучкість, здатність уникати локальних оптимумів та можливість роботи з

складними просторами пошуку роблять його цінним інструментом для дослідників та практиків в різних сферах [31].

Важливо зазначити, що генетичний алгоритм – це імовірнісний алгоритм, і він не гарантує знаходження оптимального рішення кожного разу.

Однак він може дати хороші результати для широкого спектру задач, особливо для задач з великими та складними просторами пошуку.

Для реалізації алгоритму на сьогоднішній день існує величезна кількість готових бібліотек (Python: PyGAD, mealpy; Java: JDEAP; C/C++ CppGA) та програмних інструментів (MatLab, Scilab) для генетичних алгоритмів. Спосіб реалізації програми залежить тільки від особистих побажань розробника та можливостей [32-34].

В ході роботи даної кваліфікаційної роботи було обрано бібліотеку для реалізації метаевристичних методів *mealpy,* більш детально див. розділ 2.2.

2.4 Вибір або визначення гіперпараметрів

Гіперпараметри визначаються перед початком оптимізації та впливають на процес пошуку оптимального розташування об'єктів у контейнері.

Гіперпараметри повинні бути налаштовані в приблизному діапазоні, щоб отримати швидшу конвергенцію до глобального оптимуму. В даній програмній реалізації було визначено й встановлено:

- *pc* (float): [0.7, 0.95], ймовірність того, що схрещування буде застосоване до пари батьків, в програмі встановлено 0.9.
- *pm* (float): [0.01, 0.2], ймовірність того, що мутація буде застосована до конкретного гена (компонента рішення), в програмі встановлено значення 0.05.

- *epoch* (int): кількість ітерацій, після завершення яких програма завершить свою роботу, за замовчення встановлено 1000 ітерацій
- *pop_size* (int): розмір популяції, кількість рішень, які розглядаються на кожній ітерації, в даній програмі встановлено 100.

2.5 Практична реалізація

Модель реалізована за допомогою мови програмування Python, з використанням бібліотек *numpy* для роботи з масивами та математичними обчисленнями, *mealpy* для реалізації метаевристичного алгоритму, *customtkinter* та *tkinter* для реалізації інтерфейса.

Ключовим елементом програми є клас *Optimization*, який відповідає за обчислення ризиків, функцію мети та виконання генетичного алгоритму для пошуку оптимального рішення.

Ініціалізація класу Optimization здійснюється через метод *__init__*, який приймає два параметри: *data_store* та *parent.* Так званий параметр *data_store* містить в собі клас *DataStore* з усіма даними для обчислення, такі як: матриця витрат, початкові ймовірності та тяжкості, кількість факторів та максимальний допустимий ризик. Ці дані витягуються і зберігаються в змінних класу для подальшого використання.

Сам клас *Optimization* складається з декількох функцій:

- *calculate_risk(self, solution)*: відповідає за обчислення кінцевих значень ймовірностей *(P_kin),* тяжкостей *(K_kin)* та ризиків *(R_kin)* для кожного небезпечного чинника.
- *objective_function (self, solution):* є основною функцією мети для генетичного алгоритму. Він перетворює рішення у бінарний вектор, розраховує загальну вартість вибраних профілактичних заходів і загальний ризик.

- *solve (self)*: відповідає за налаштування і запуск генетичного алгоритму.
- *final_values (self, flat_solution):* обчислює кінцеві значення ймовірностей, тяжкостей і ризиків для кожного небезпечного чинника, використовуючи знайдене оптимальне рішення. Даний процес дозволяє оцінити ефективність вибраних заходів.

Інтерфейс програми розроблений за допомогою бібліотек *customtkinter* і *tkinter*, що забезпечує зручність використання та приємний зовнішній вигляд. Основне вікно програми містить кілька форм (вікон), які дозволяють користувачам вводити необхідні дані, завантажувати файли, переглядати введені дані та отримувати оптимізовані результати.

Основні форми програми:

- *Головна сторінка (HomePage):*дана форма відображає привітання та коротка інструкція користувача. Користувачам пропонується вибрати спосіб введення даних: вручну або шляхом завантаження файлу.
- *Форма введення даних про чинники (DataEntryPage):* форма, що містить гріди та кнопки для введення користувачем вручну даних про небезпечні чинники.
- *Форма введення даних про профілактичні заходи (NewDataEntryPage):* форма, що містить гріди та кнопки для введення користувачем вручну даних про профілактичні заходи.
- *Форма завантаження файлу (FileUploadPage):* форма для завантаження файлу з попередньо підготовленими даними. Відображається інструкція щодо формату файлу, щоб користувач знав, як правильно підготувати дані для завантаження.
- *Форма відображення введених даних (EnteredDataPage):* дана форма відображає дані, які користувач ввів або завантажив з

файлу в таблиці. Після перевірки даних на формі можна запустити процес пошуку рішення.

– *Форма відображення рішення (SolutionPage):* дана форма відображає результати оптимізації. Відображаються рекомендовані профілактичні заходи, їх загальна вартість, ризик настання нещасного випадку після застосування заходів, а також ефективність обраних заходів. Користувач має можливість зберегти результати у вигляді звіту.

Вся програма реалізована через класи, які поєднані між собою. Для кращого розуміння роботи програми й зв’язків між класами було зіставлено UML-діаграму, яку наведено на рис. 2.1.

Класи:

– *HomePage.*

  Методи: *__init__(), show_frame()*

– *DataEntryPage.*

  Методи: *__init__(), clear_fields(), validate_and_create_grid(), validate_grid()*

  Зв'язки: Переходить до *NewDataEntryPage*, зберігає дані в *DataStore*.

– *NewDataEntryPage*

  Методи: *__init__(), populate_grid(), validate_and_save(), clear_entries()*

  Зв'язки: Зберігає дані в *DataStore*.

– *FileUploadPage*

  Методи: *__init__(), upload_file()*

  Зв'язки: Зберігає дані в *DataStore*

– *EnteredDataPage*

  Методи: *__init__(), update_table()*

Зв'язки: Зчитує дані з *DataStore*, запускає оптимізацію через *Optimization*.

– *SolutionPage*

Методи: *__init__(), show_solution(), save_report()*

Зв'язки: Зчитує дані з *DataStore*, генерує звіт через *ReportGenerator*.

– *ReportGenerator*

Методи*: __init__(), save_report()*

Зв’язки*: Отримує дані від SolutionPage.*

– *DataStore*

Атрибути*: A, P_ai, T_ai, probabilities, severities, num_factors, max_risk*

Методи*: set_parameters(), set_initial_data(), set_measures_data(), get_num_factors(), get_max_risk()*

Зв'язки*:* Зберігає дані від *DataEntryPage, NewDataEntryPage* та *FileUploadPage.* Дані зчитуються *EnteredDataPage, Optimization* та *SolutionPage.*

– *Optimization*

Атрибути*: data_store, parent*

Методи*: __init__(), calculate_risk(), objective_function(), solve(), final_values()*

Зв'язки: Зчитує дані з *DataStore,* відображає прогрес у *ProgressWindow,* завершує оптимізацію в *SolutionPage.*

– *ProgressWindow*

Методи*: __init__(), update_progress(), close()*

Зв'язки: Використовується *Optimization* для відображення прогресу оптимізації.

Вза’ємодії між зв’язками:

1. *HomePage*: Користувач переходить або до *DataEntryPage*, або до *FileUploadPage*.
2. *DataEntryPage*: Користувач вводить основні параметри, які зберігаються в DataStore, і переходить до *NewDataEntryPage*.
3. *NewDataEntryPage*: Користувач вводить додаткові дані про профілактичні заходи, які зберігаються в *DataStore*.
4. *FileUploadPage*: Користувач завантажує файл з даними, які зберігаються в *DataStore*.
5. *EnteredDataPage*: Дані з *DataStore* зчитуються і відображаються. Користувач може запустити оптимізацію через *Optimization*.
6. *Optimization*: Виконує оптимізацію, зчитуючи дані з *DataStore* і відображаючи прогрес у *ProgressWindow*. Після завершення передає результати в *SolutionPage*.
7. *SolutionPage*: Зчитує дані з *DataStore*, відображає результати оптимізації, і дозволяє згенерувати звіт через *ReportGenerator*.
8. *ReportGenerator*: Генерує звіт на основі даних з *SolutionPage*.

Код програми наведено в Додатку В.

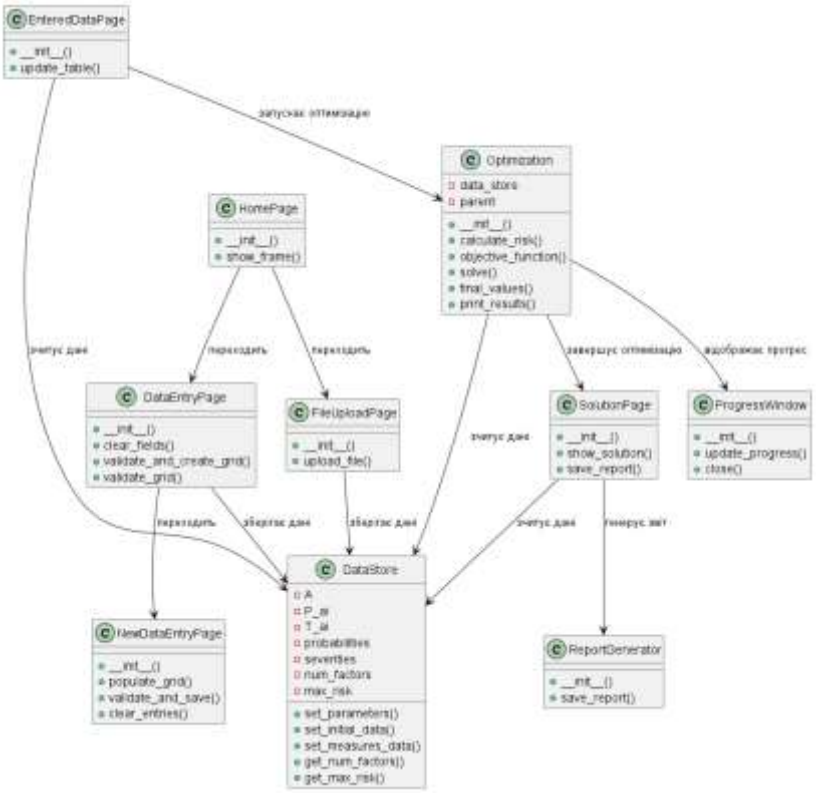


Рис. 2.1 – UML-діаграма зв’язків класів

2.6 Інструкція користувача

Запуск програми виконується за допомогою виконавчого файлу з назвою «Оптимізація ризиків» (рис. 2.2). Якщо виникають помилки, потрібно переконатись що пристрій відповідає мінімальним системним вимогам та встановлений програмний пакет PyCharm, а також бібліотеки *mealpy*, *numpy*, *customtkinter ma tkinter*.

Після запуску програми відкривається вікно програми з головною сторінкою (рис. 2.3), що містить інформацію про ціль роботи програми та подальші дії.

На даному етапі користувачу необхідно обрати спосіб надання даних для пошуку рішення. Він може ввести їх самостійно або завантажити файл з підготовленими даними.

Рис. 2.2 – Виконавчий файл для запуску програми

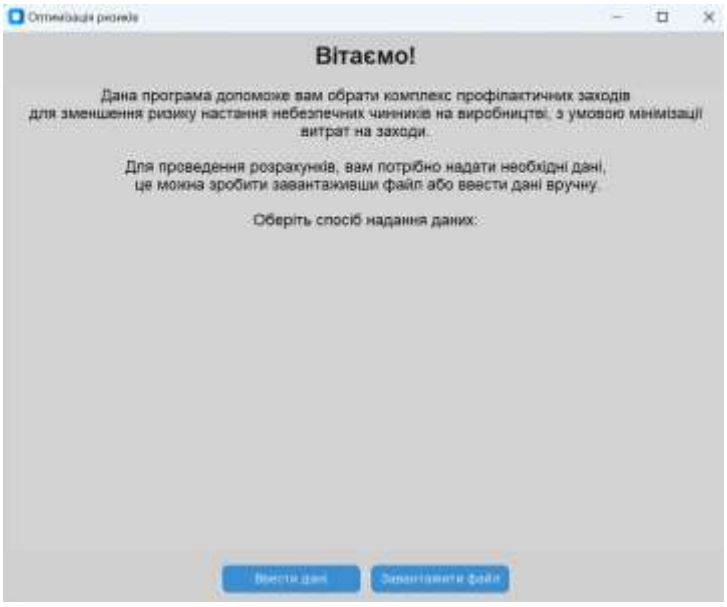


Рис. 2.3 – Головна сторінка програми

1. *Введення даних вручну.*

При введенні даних вручну відкривається наступне вікно (рис. 2.4), де користувачу необхідно ввести «Максимальне значення тяжкості наслідків», для визначення шкали її оцінки, «Максимальне допустиме значення ризику», для розуміння побажань користувача та «Кількість небезпечний чинників», що можуть спричинити нещасний випадок.

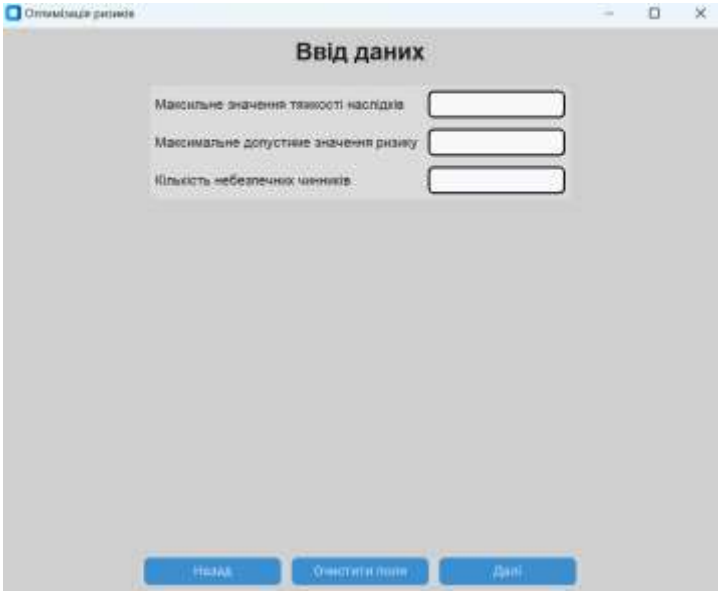


Рис. 2.4 – Форма для вводу даних

Після введення користувачем всіх необхідних даних він має натиснути кнопку «Далі». Якщо всі дані були введені коректно, тобто тільки числові значення відповідного типу, то на формі стають видимими комірки для вводу інформації про небезпечні чинники (див. рис. 2.4).

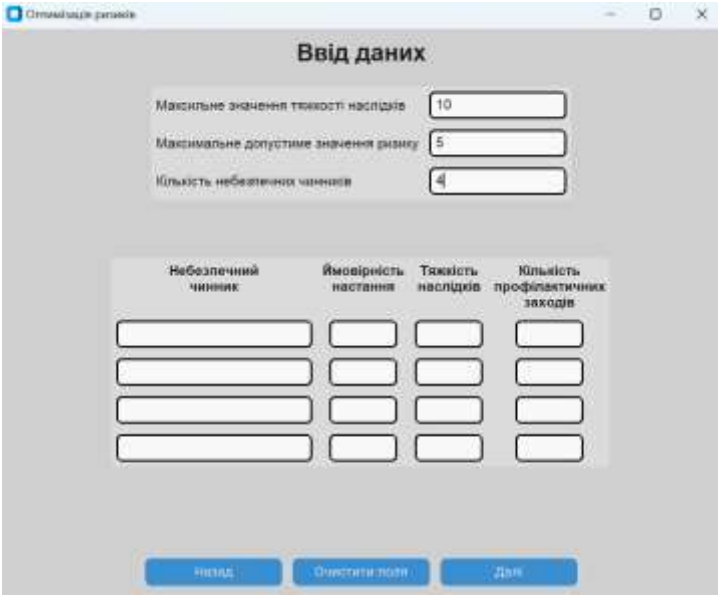


Рис. 2.5 – Форма для вводу даних про небезпечні чинники

На даному етапі користувач вводить назви небезпечних чинників, ймовірність їх настання, тяжкість їх наслідків та кількість профілактичних заходів, які можна застосувати для даного небезпечного чинника для зменшення ризиків його настання.

Необхідно вважати, що всі поля для введення є обов'язковими й мають відповідати дійсності. Тобто в поле для вводу ймовірності необхідно ввести число від 0 до 1, в поле для введення тяжкості наслідків необхідно ввести число, яке не перевищує максимальне можливе значення аби відповідати шкалі оцінювання та кількість профілактичних заходів має бути тільки цілим числом. Якщо якесь з введених значень не відповідає правилам, то комірка буде підсвічена червоним кольором, як показано на рис. 2.6.

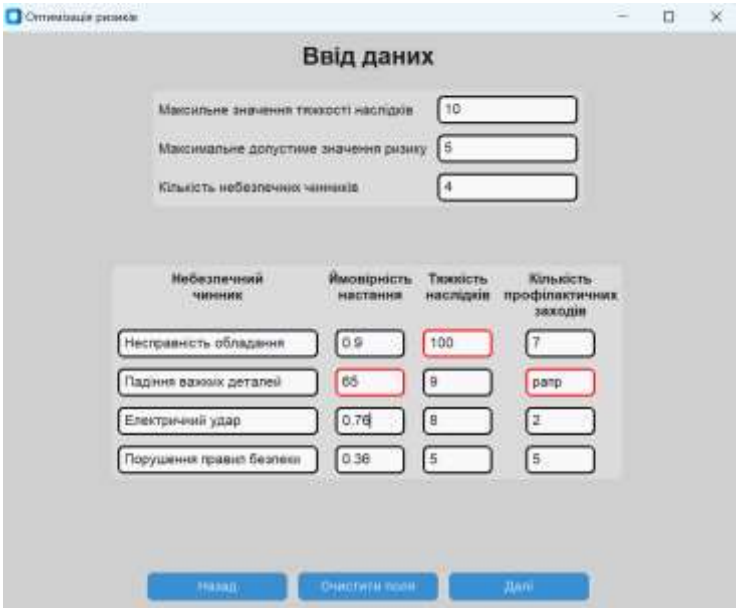


Рис. 2.6 – Форма для вводу даних про небезпечні чинники з неправильними даними

Якщо ж всі дані було введено правильно, відкривається наступна форма для введення інформації про профілактичні заходи (рис. 2.7). Користувач має ввести назви профілактичних заходів, вартість їх проведення та на скільки вони зменшують ймовірність настання чинника й тяжкості наслідків.

Рис. 2.7 – Форма для вводу даних про профілактичні заходи

Після завершення вводу інформації користувач натискає кнопку «Зберегти». В разі, якщо введені дані також не відповідають дійсності, то комірки підсвічуються червоним, як показано на рис. 2.8, тоді користувач має виправити неправильні дані й знову натиснути кнопку «Зберегти».

Коли всі дані введені правильно, то відкривається вікно з таблицею з введеними даними.

Рис. 2.8 – Форма для вводу даних про профілактичні заходи з неправильними даними

2. Завантаження даних

Якщо користувач на головній сторінці обрав «Завантажити файл», тоді відкривається форма для завантаження файлу, рис. 2.9. На даній сторінці наведено інструкцію, як має виглядати файл з даними.

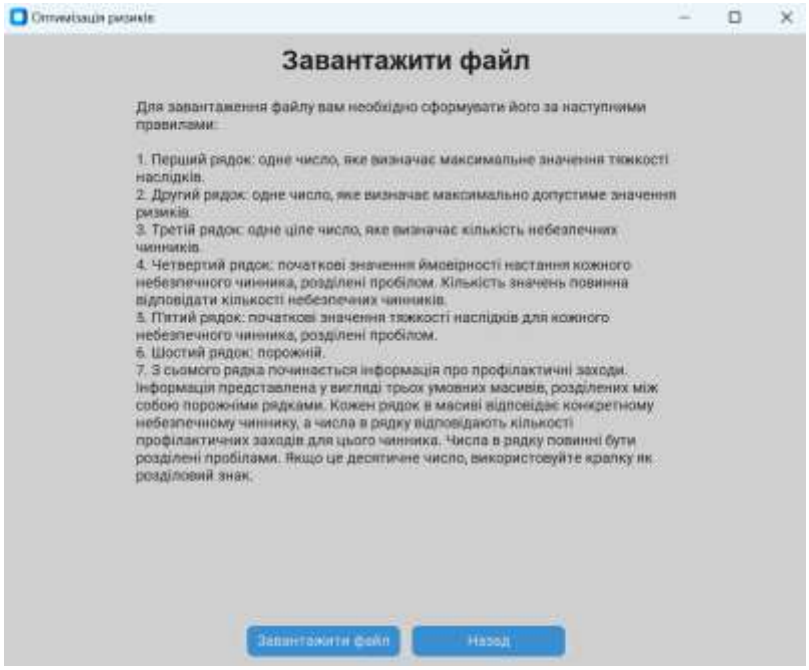


Рис. 2.9 – Форма для завантаження файлу

Інструкція підготовки файлу:

«Для завантаження файлу вам необхідно сформувати його за наступними правилами та зберегти в форматі *.txt*:

1. Перший рядок: одне число, яке визначає максимальне значення тяжкості наслідків.

2. Другий рядок: одне число, яке визначає максимально допустиме значення ризиків.

3. Третій рядок: одне ціле число, яке визначає кількість небезпечних чинників.

4. Четвертий рядок: початкові значення ймовірності настання кожного небезпечного чинника, розділені пробілом. Кількість значень повинна відповідати кількості небезпечних чинників.

5. П’ятий рядок: початкові значення тяжкості наслідків для кожного небезпечного чинника, розділені пробілом.

6. Шостий рядок: порожній.

7. З сьомого рядка починається інформація про профілактичні заходи. Інформація представлена у вигляді умовних масивів, які складаються з трьох рядків, розділених між собою порожніми рядками. Кожен умовний масив відповідає одному небезпечному чиннику, тобто кількість масивів має відповідати кількості небезпечних чинників.

7.1. Перший рядок в умовному масиві відповідає за вартість профілактичних заходів.

7.2. Другий рядок в умовному масиві відповідає за ймовірність, точніше на скільки захід її зменшує.

7.3. Третій рядок в умовному масиві відповідає за тяжкість, точніше на скільки захід її зменшує.

Числа в рядку повинні бути розділені пробілами. Якщо це десяткове число, використовуйте крапку як розділовий знак.»

Користувач має уважно ознайомитися з інструкцією й правильно підготувати дані у файлі, на рис. 2.10 наведено приклад вигляду файлу з підготовленою інформацією.

Коли файл підготовлений й збережений, користувач натискає кнопку «Завантажити файл», обирає його в його провіднику й автоматично відкривається вікно з введеними даними, як продемонстровано на рис. 2.11.

```
10
5
4
0.87 0.75 0.8 0.32
8 9 6 4

9.997 18.638 52.279 13.819 50.830 5.304 11.690
0.05 0.1 0 0.15 0.08 0.2 0
0 0.05 0.2 0 0.1 0 0.1

71.634 26.397 30.173 11.626
0.27 0.38 0.43 0.47
0.01 0.05 0.03 0

60.476 57.168 56.283 72.138 5.156
0.22 0.14 0 0.19 0.05
0 0.01 0.15 0 0.02

16.300 58.864 20.878 75.145 25.616
0.26 0.07 0.15 0.04 0.17
0 0.08 0.02 0.12 0
```

Рис. 2.10 – Приклад оформлення файлу з даними

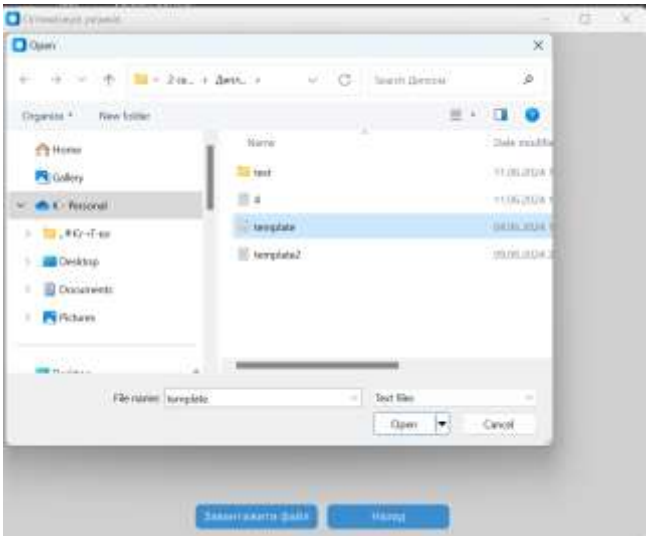


Рис. 2.11 – Завантаження файлу

Наступні кроки є однаковими для обох випадків (введення даних вручну або завантаження файлу).

Після етапу введення інформації виводиться вікно з таблицею й усіма даними, користувач може переглянути всі дані на правильність, див. рис. 2.12. В разі, користувач обрав спосіб завантаження даних з файлу, назви небезпечних чинників та профілактичних чинників нумеруються автоматично. Для зручності подальшої роботи з даними, користувачу рекомендується так само попередньо пронумерувати чинники та заходи для себе.

Якщо все достовірно, тоді користувач натискає кнопку «Пошук рішення» й запускається процес оптимізації, як показано на рис. 2.13.

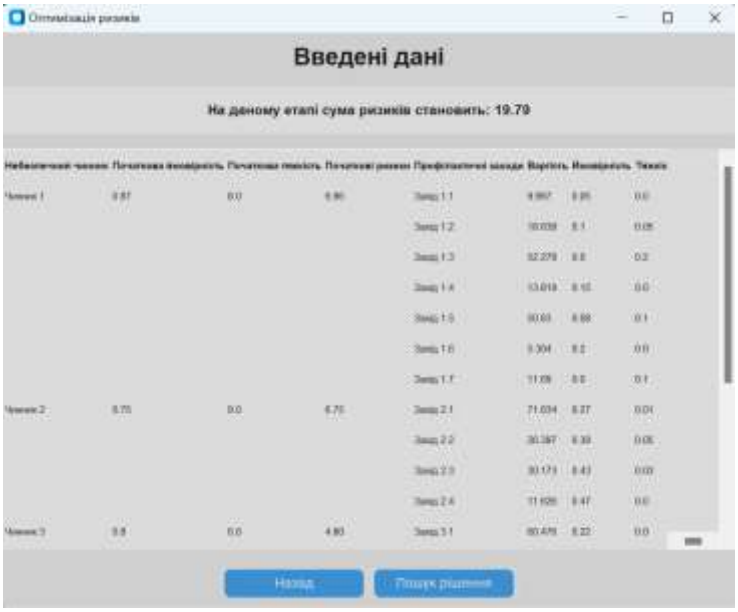


Рис. 2.12 – Форма з введеними даними для перевірки

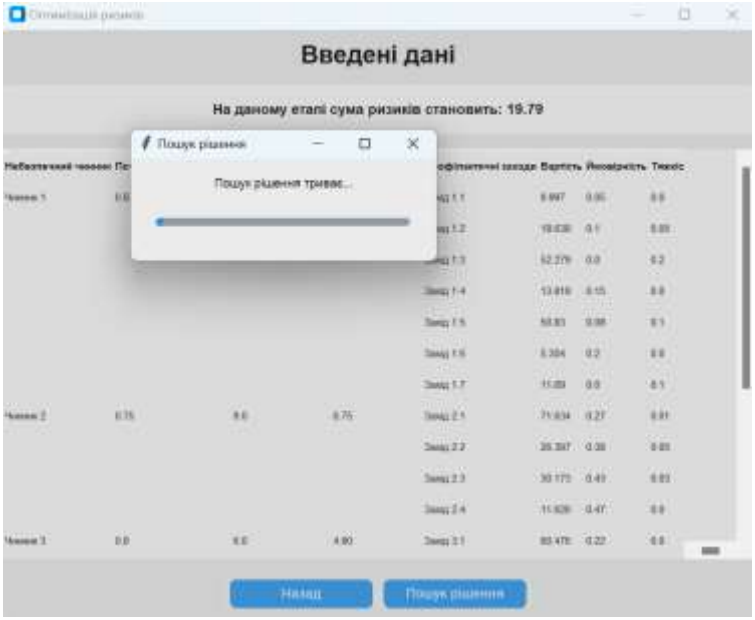


Рис. 2.13 – Процес пошуку рішення

По завершенню роботи оптимізаційного процесу, відкривається нове вікно з відображенням всіх результатів, див. рис. 2.14. Користувач може побачити загальну вартість рекомендованих профілактичні заходів, знижене значення ризиків настання нещасного випадку, ефективність проведення

рекомендованих профілактичних заходів, та безпосередньо список самих профілактичних заходів.

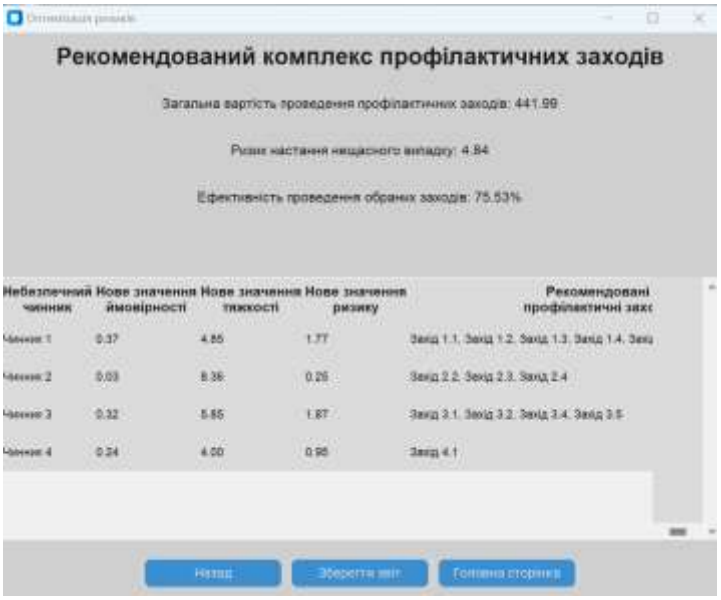


Рис. 2.14 – Форма з результатами розрахунків

У випадку, якщо програмі не вдалося знайти оптимального рішення, про це буде виведене повідомлення на екран, як на рис. 2.15.

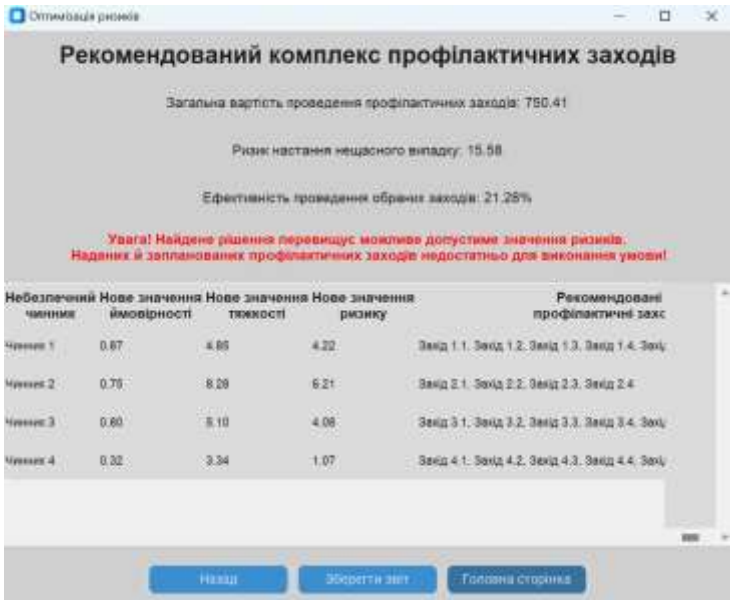


Рис. 2.15 – Форма з результатами, в випадку їх не оптимальності

Ознайомившись з результатами, користувач може завантажити звіт на комп’ютер. Для цього йому необхідно натиснути кнопку «Зберегти звіт», назвати файл та місце його зберігання в себе на комп’ютері, див рис. 2.16.

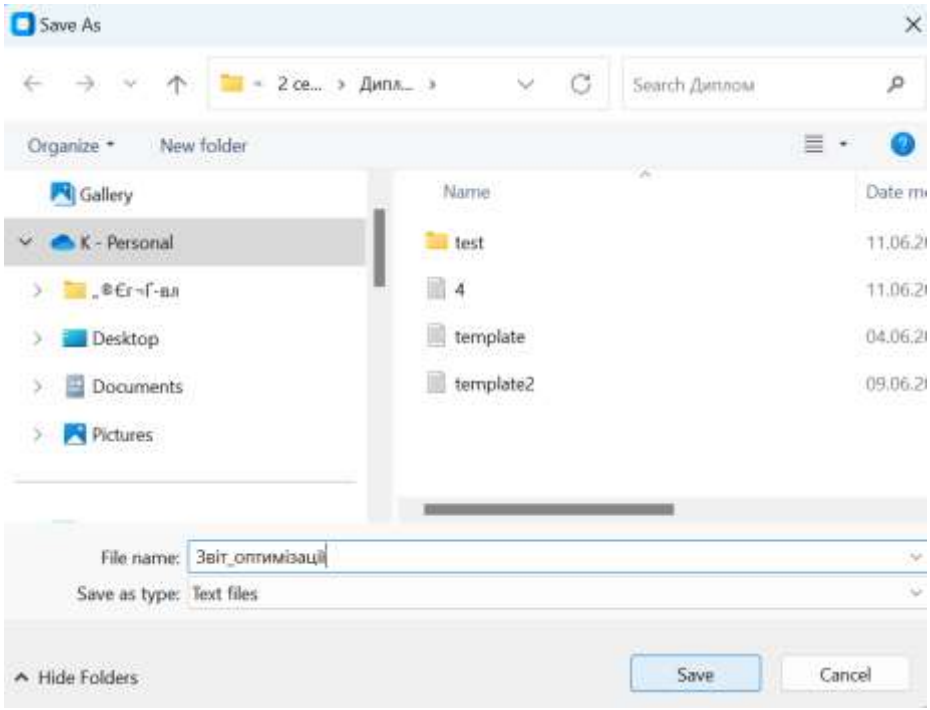


Рис. 2.16 – Збереження звіту

Існує два шаблони звіту, один якщо рішення було знайдено (рис. 2.17) та звіт для випадку, якщо оптимального рішення знайдено не було (рис. 2.18).

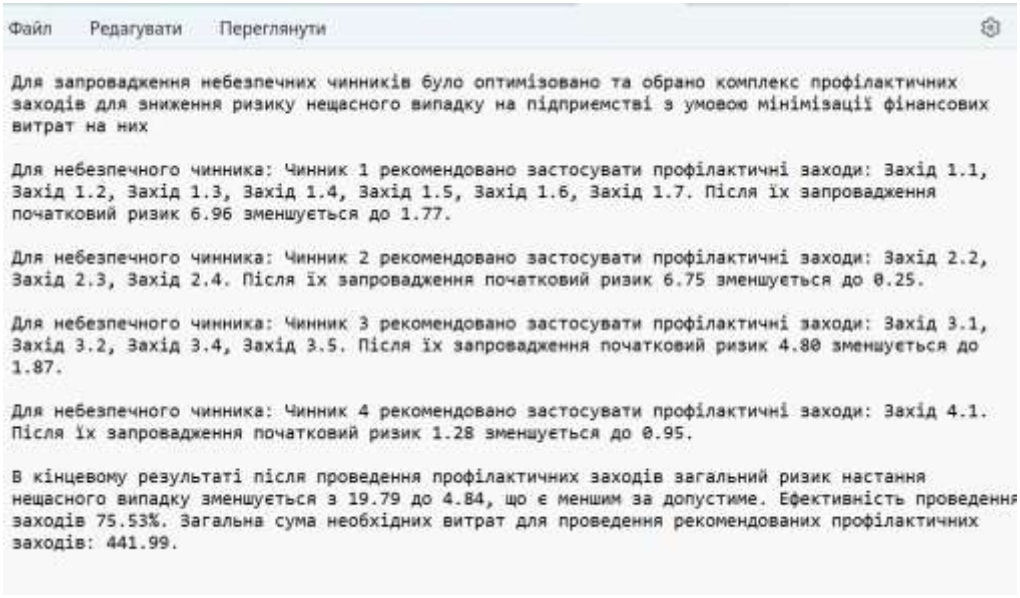


Рис. 2.17 – Приклад шаблону звіту 1

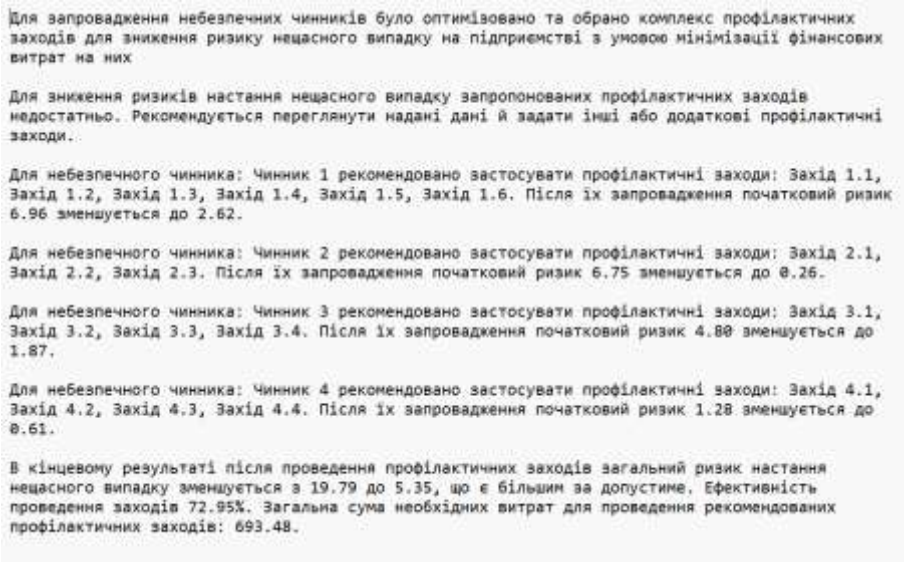


Рис. 2.18 – Приклад шаблону звіту 2

Після завершення роботи з програмою користувач може або закрити її, або ж повернутися до головної сторінки й провести нові розрахунки.

3 ТЕСТУВАЛЬНО-АНАЛІТИЧНИЙ РОЗДІЛ

3.1 Планування тестування

Для перевірки роботи програми буде проведено декілька тестувань. В даному розділі за основу взято конкретну задачу, з якої обрано всі або лише деякі дані для різних тестувань.

Приклад задачі:

Автомобільний завод, на якому здійснюється складання автомобілів, звернувся до експерта зі запитом проаналізувати та оптимізувати комплекс профілактичних заходів для зменшення ризику нещасного випадку.

Завод повідомив про конкретний нещасний випадок, який би він уникнути – травмування працівника під час роботи на конвеєрі. Підприємством також було надано ряд небезпечних чинників, які б могли призвести до даного нещасного випадку: несправність обладнання, падіння важких деталей, електричний удар та порушення правил безпеки.

В ході спілкування, консультацій та збору інформації було отримано дані про ймовірність настання та тяжкість наслідків кожного небезпечного чинника:

- *несправність обладнання:* ймовірність – 0.9, тяжкість – 7;
- *падіння важких деталей:* ймовірність – 0.65, тяжкість – 9;
- *електричний удар:* ймовірність – 0.76, тяжкість – 8;
- *порушення правил безпеки:* ймовірність – 0.36, тяжкість – 5.

Наступним етапом стало зіставлення списків можливих профілактичних заходів та визначення їх ефективності для конкретних небезпечних чинників та ціна їх проведення чи застосування. Всі отримані дані були зведені в таблицю 3.1 для зручності.

Таблиця 3.1

Небезпечний чинник	**Профілактичний захід**	**Вартість проведення**	**Зниження ймовірності**	**Зниження тяжкості**
Несправність обладнання	Оновлення обладнання	10.768	0.04	0
	Регулярне технічне обслуговування	16.768	0.1	0.04
	Використання захисних бар'єрів	49.745	0	0.1
	Встановлення датчиків несправності	16.862	0.18	0
	Автоматизація процесів	51.674	0.08	0.2
	Забезпечення належного освітлення	7.365	0.2	0
	Використання спеціальних інструментів	11.690	0	0.19
Падіння важких деталей	Використання кранів та підйомників	68.359	0.19	0.04
	Регулярний огляд та заміна підйомного обладнання	30.852	0.42	0.01
	Тренінги для працівників	12.357	0.35	0.25
	Використання страхувальних систем	9.675	0.54	0
Еклектичний удар	Регулярна перевірка електрообладнання	56.678	0.17	0
	Використання захисних пристроїв	48.697	0.13	0.07
	Навчання працівників правилами безпеки	53.687	0	0.21
	Встановлення захисних бар'єрів	67.657	0.16	0

Продовження табл. 2.1

	Використання спеціальних інструментів	7.356	0.03	0.05
Порушення правил безпеки	Регулярні тренінги з техніки безпеки	19.756	0.31	0
	Впровадження системи управління безпекою праці	69.456	0.05	0.06
	Моніторинг дотримання правил безпеки	14.435	0.19	0.03
	Використання засобів індивідуального захисту	69.576	0.08	0.17
	Організація контрольних інспекцій	19.576	0.19	0

Для зручності подальшого розуміння результатів буде пронумеровано всі чинники та профілактичні заходи та зібрано в таблицю 3.2.

Таблиця 3.2

№	**Небезпечний чинник**	**№**	**Профілактичний захід**
1	Несправність обладнання	1.1	Оновлення обладнання
		1.2	Регулярне технічне обслуговування
		1.3	Використання захисних бар’єрів
		1.4	Встановлення датчиків несправності
		1.5	Автоматизація процесів
		1.6	Забезпечення належного освітлення
		1.7	Використання спеціальних інструментів
2	Падіння важких деталей	2.1	Використання кранів та підйомників
		2.2	Регулярний огляд та заміна підйомного обладнання
		2.3	Тренінги для працівників

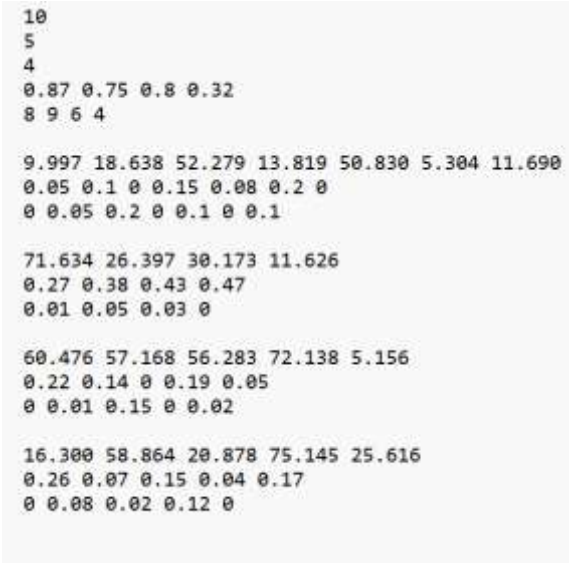
		2.4	Використання страхувальних систем

Продовження табл. 3.2

3	Еклектичний удар	3.1	Регулярна перевірка електрообладнання
		3.2	Використання захисних пристроїв
		3.3	Навчання працівників правилами безпеки
		3.4	Встановлення захисних бар'єрів
		3.5	Використання спеціальних інструментів
4	Порушення правил безпеки	4.1	Регулярні тренінги з техніки безпеки
		4.2	Впровадження системи управління безпекою праці
		4.3	Моніторинг дотримання правил безпеки
		4.4	Використання засобів індивідуального захисту
		4.5	Організація контрольних інспекцій

Тест 1

В даному тесті розглянуто всі небезпечні чинники та всі профілактичні заходи. Для можливості завантаження файлу в програму його було підготовлено, див. рис. 3.1.



```
10
5
4
0.87 0.75 0.8 0.32
8 9 6 4

9.997 18.638 52.279 13.819 50.830 5.304 11.690
0.05 0.1 0 0.15 0.08 0.2 0
0 0.05 0.2 0 0.1 0 0.1

71.634 26.397 30.173 11.626
0.27 0.38 0.43 0.47
0.01 0.05 0.03 0

60.476 57.168 56.283 72.138 5.156
0.22 0.14 0 0.19 0.05
0 0.01 0.15 0 0.02

16.300 58.864 20.878 75.145 25.616
0.26 0.07 0.15 0.04 0.17
0 0.08 0.02 0.12 0
```

Рис. 3.1 – Підготовлений файл з даними для тесту 1

Тест 2

В даному тесті розглянуто тільки небезпечні чинники 1, 2, 3 та профілактичні заходи: 1.1-1.6, 2.1-2.4, 3.1-3.5. Для можливості завантаження файлу в програму його було підготовлено, див. рис. 3.2.

```
10
5
3
0.87 0.75 0.8
8 9 6

9.997 18.638 52.279 13.819 50.830 5.304
0.04 0.1 0 0.18 0.08 0.2
0 0.04 0.1 0 0.2 0

71.634 26.397 30.173
0.19 0.42 0.35
0.04 0.01 0.25

11.676 48.697 53.687 67.657 7.356
0.17 0.13 0 0.16 0.03
0 0.07 0.21 0 0.05
```

Рис. 3.2 − Підготовлений файл з даними для тесту 2

Тест 3

В даному тесті розглянуто всі небезпечні чинники та деякі профілактичні заходи: Для можливості завантаження файлу в програму його було підготовлено, див. рис. 3.3.

```
10
5
4
0.87 0.75 0.8 0.32
8 9 6 4

9.997 18.638 52.279 13.819 50.830 11.690
0.05 0.1 0 0.15 0.08 0
0 0.05 0.2 0 0.1 0.1

71.634 26.397 11.626
0.27 0.38 0.47
0.01 0.05 0

60.476 57.168 56.283 72.138
0.22 0.14 0 0.19
0 0.01 0.15 0

58.864 20.878 75.145 25.616
0.07 0.15 0.04 0.17
0.08 0.02 0.12 0
```

Рис. 3.3 − Підготовлений файл з даними для тесту 3

3.2 Результати тестування

Результати тесту 1

Для запровадження небезпечних чинників було оптимізовано та обрано комплекс профілактичних заходів для зниження ризику нещасного випадку на підприємстві з умовою мінімізації фінансових витрат на них

Для небезпечного чинника: Чинник 1 рекомендовано застосувати профілактичні заходи: Захід 1.1, Захід 1.2, Захід 1.3, Захід 1.4, Захід 1.5, Захід 1.6, Захід 1.7. Після їх запровадження початковий ризик 6.96 зменшується до 1.77.

Для небезпечного чинника: Чинник 2 рекомендовано застосувати профілактичні заходи: Захід 2.2, Захід 2.3, Захід 2.4. Після їх запровадження початковий ризик 6.75 зменшується до 0.25.

Для небезпечного чинника: Чинник 3 рекомендовано застосувати профілактичні заходи: Захід 3.1, Захід 3.2, Захід 3.4, Захід 3.5. Після їх запровадження початковий ризик 4.80 зменшується до 1.87.

Для небезпечного чинника: Чинник 4 рекомендовано застосувати профілактичні заходи: Захід 4.1. Після їх запровадження початковий ризик 1.28 зменшується до 0.95.

В кінцевому результаті після проведення профілактичних заходів загальний ризик настання нещасного випадку зменшується з 19.79 до 4.84, що є меншим за допустиме. Ефективність проведення заходів 75.53%. Загальна сума необхідних витрат для проведення рекомендованих профілактичних заходів: 441.99.

Проаналізувавши отримані результати можна побачити, що отриманий розрахунок є оптимальним, так як він дозволяє досягти значного зниження ризику нещасних випадків до рівня нижче допустимого, забезпечуючи при цьому високу ефективність проведених заходів. Крім того, загальна вартість

реалізації заходів є економічно обґрунтованою, що підтверджує доцільність їх впровадження.

Результати роботи програми та сформований звіт продемонстровано на рис. 3.4–3.5.

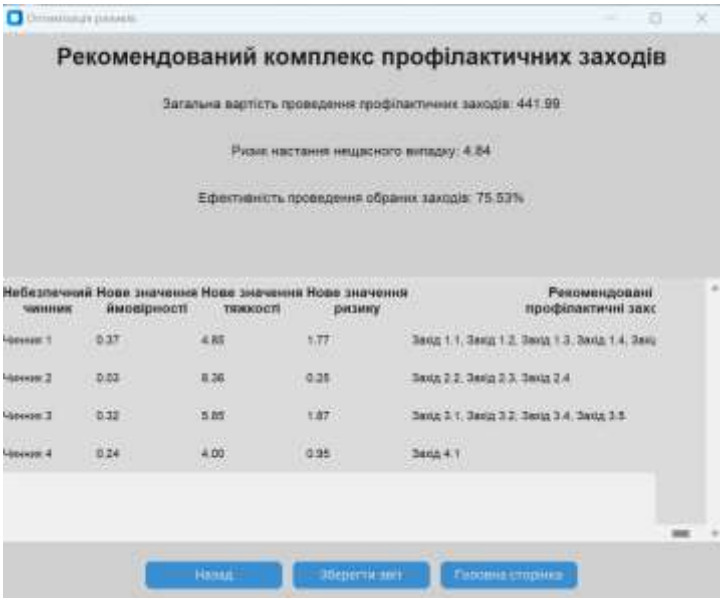


Рис. 3.4 – Результати тесту 1

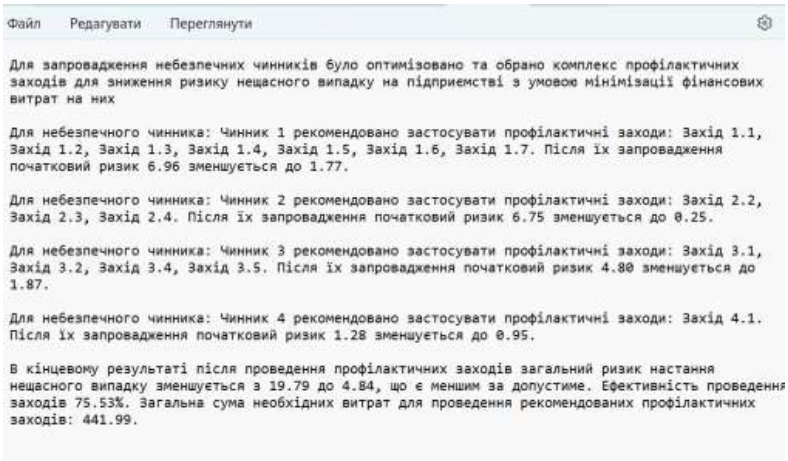


Рис. 3.5 – Звіт тесту 1

Результати тесту 2

Для запровадження небезпечних чинників було оптимізовано та обрано комплекс профілактичних заходів для зниження ризику нещасного випадку на підприємстві з умовою мінімізації фінансових витрат на них

Для небезпечного чинника: Чинник 1 рекомендовано застосувати профілактичні заходи: Захід 1.2, Захід 1.4, Захід 1.5, Захід 1.6. Після їх запровадження початковий ризик 6.96 зменшується до 2.42.

Для небезпечного чинника: Чинник 2 рекомендовано застосувати профілактичні заходи: Захід 2.1, Захід 2.2, Захід 2.3. Після їх запровадження початковий ризик 6.75 зменшується до 0.20.

Для небезпечного чинника: Чинник 3 рекомендовано застосувати профілактичні заходи: Захід 3.1, Захід 3.2, Захід 3.3, Захід 3.5. Після їх запровадження початковий ризик 4.80 зменшується до 2.33.

В кінцевому результаті після проведення профілактичних заходів загальний ризик настання нещасного випадку зменшується з 18.51 до 4.95, що є меншим за допустиме. Ефективність проведення заходів 73.26%. Загальна сума необхідних витрат для проведення рекомендованих профілактичних заходів: 338.21.

Проаналізувавши отримані результати можна побачити, що отриманий розрахунок є оптимальним, так як він дозволяє досягти значного зниження ризику нещасних випадків до рівня нижче допустимого, забезпечуючи при цьому високу ефективність проведених заходів. Крім того, загальна вартість реалізації заходів є економічно обґрунтованою, що підтверджує доцільність їх впровадження.

Результати роботи програми та сформований звіт продемонстровано на рис. 3.6–3.7.

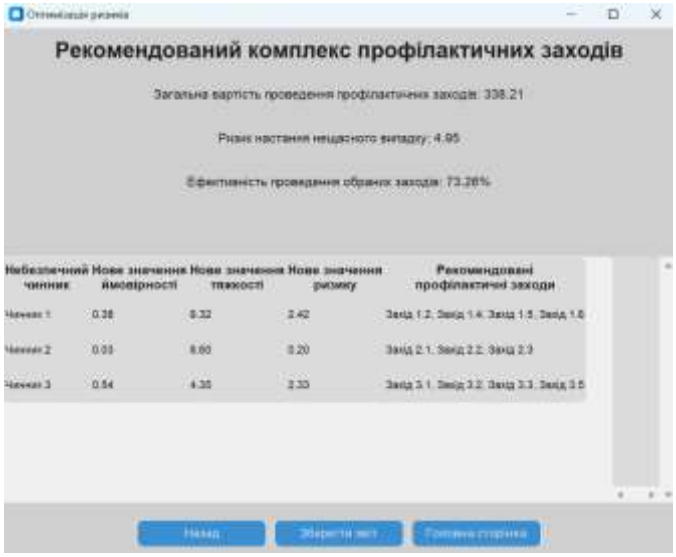


Рис. 3.6 – Результати тесту 2

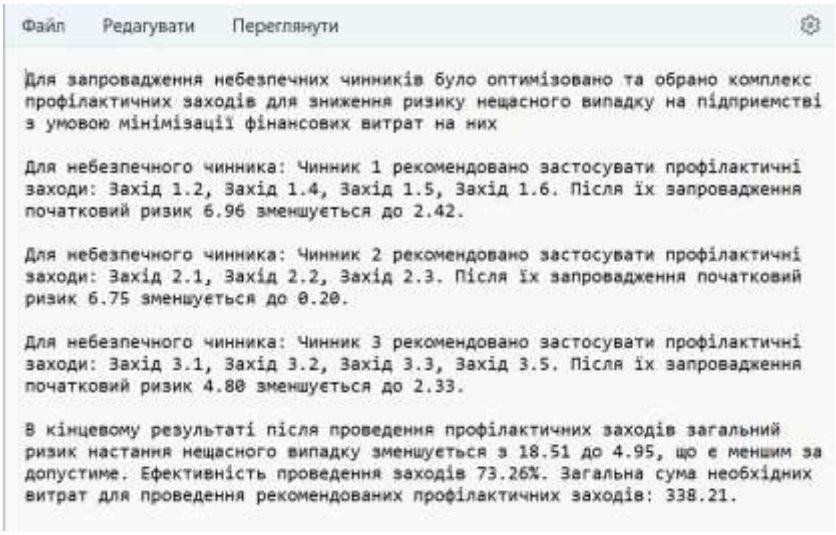


Рис. 3.7 – Звіт тесту 2

Результати тесту 3

Для запровадження небезпечних чинників було оптимізовано та обрано комплекс профілактичних заходів для зниження ризику нещасного випадку на підприємстві з умовою мінімізації фінансових витрат на них

Для зниження ризиків настання нещасного випадку запропонованих профілактичних заходів недостатньо. Рекомендується переглянути надані дані й задати інші або додаткові профілактичні заходи.

Для небезпечного чинника: Чинник 1 рекомендовано застосувати профілактичні заходи: Захід 1.1, Захід 1.2, Захід 1.3, Захід 1.4, Захід 1.5, Захід 1.6. Після їх запровадження початковий ризик 6.96 зменшується до 2.62.

Для небезпечного чинника: Чинник 2 рекомендовано застосувати профілактичні заходи: Захід 2.1, Захід 2.2, Захід 2.3. Після їх запровадження початковий ризик 6.75 зменшується до 0.26.

Для небезпечного чинника: Чинник 3 рекомендовано застосувати профілактичні заходи: Захід 3.1, Захід 3.2, Захід 3.3, Захід 3.4. Після їх запровадження початковий ризик 4.80 зменшується до 1.87.

Для небезпечного чинника: Чинник 4 рекомендовано застосувати профілактичні заходи: Захід 4.1, Захід 4.2, Захід 4.3, Захід 4.4. Після їх запровадження початковий ризик 1.28 зменшується до 0.61.

В кінцевому результаті після проведення профілактичних заходів загальний ризик настання нещасного випадку зменшується з 19.79 до 5.35, що є більшим за допустиме. Ефективність проведення заходів 72.95%. Загальна сума необхідних витрат для проведення рекомендованих профілактичних заходів: 693.48.

Проаналізувавши отримані результати можна побачити, що запропонований комплекс профілактичних заходів, незважаючи на значне зниження загального ризику нещасних випадків та високу ефективність, не дозволяє досягти рівня ризику нижче допустимого. Це свідчить про необхідність перегляду та оптимізації обраних заходів, можливо, шляхом включення додаткових заходів або заміни деяких з них на більш ефективні. Крім того, слід враховувати економічну складову, прагнучі до мінімізації витрат на реалізацію профілактичних заходів.

Результати роботи програми та сформований звіт продемонстровано на рис. 3.8–3.9.

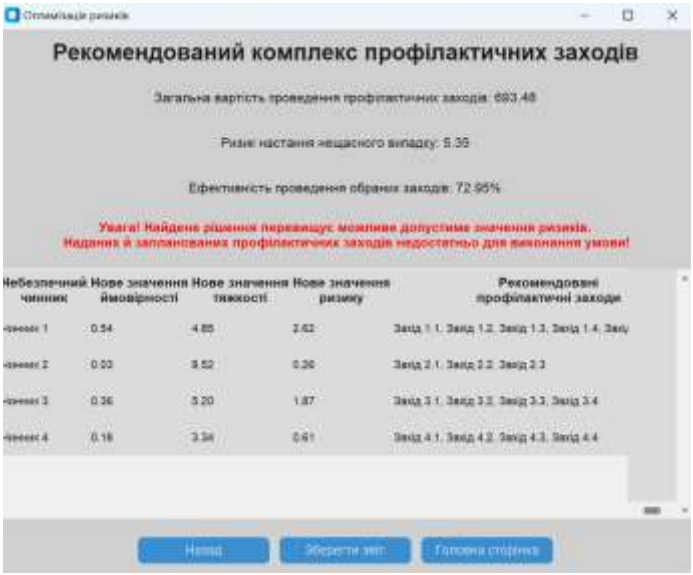


Рис. 3.8 – Результати тесту 3

Для запровадження небезпечних чинників було оптимізовано та обрано комплекс профілактичних заходів для зниження ризику нещасного випадку на підприємстві з умовою мінімізації фінансових витрат на них

Для зниження ризиків настання нещасного випадку запропонованих профілактичних заходів недостатньо. Рекомендується переглянути надані дані й задати інші або додаткові профілактичні заходи.

Для небезпечного чинника: Чинник 1 рекомендовано застосувати профілактичні заходи: Захід 1.1, Захід 1.2, Захід 1.3, Захід 1.4, Захід 1.5, Захід 1.6. Після їх запровадження початковий ризик 6.96 зменшується до 2.62.

Для небезпечного чинника: Чинник 2 рекомендовано застосувати профілактичні заходи: Захід 2.1, Захід 2.2, Захід 2.3. Після їх запровадження початковий ризик 6.75 зменшується до 0.26.

Для небезпечного чинника: Чинник 3 рекомендовано застосувати профілактичні заходи: Захід 3.1, Захід 3.2, Захід 3.3, Захід 3.4. Після їх запровадження початковий ризик 4.80 зменшується до 1.87.

Для небезпечного чинника: Чинник 4 рекомендовано застосувати профілактичні заходи: Захід 4.1, Захід 4.2, Захід 4.3, Захід 4.4. Після їх запровадження початковий ризик 1.28 зменшується до 0.61.

В кінцевому результаті після проведення профілактичних заходів загальний ризик настання нещасного випадку зменшується з 19.79 до 5.35, що є більшим за допустиме. Ефективність проведення заходів 72.95%. Загальна сума необхідних витрат для проведення рекомендованих профілактичних заходів: 693.48.

Рис. 3.9 – Звіт тесту 3

ВИСНОВКИ

У даній кваліфікаційній роботі було проведено комплексний аналіз та оптимізацію вибору профілактичних заходів для зниження ризику нещасних випадків на виробництві. Запропоновано підхід, що поєднує зниження ризиків небезпечних подій до прийнятного рівня та мінімізацію фінансових витрат, що є особливо актуальним в умовах сучасного виробництва та інтеграції України до європейського ринку.

В інформаційно-аналітичному розділі було проведено огляд літератури з питань управління ризиками на виробництві, проаналізовано існуючі підходи та методики оцінки ризиків. Визначено актуальність проблеми та обґрунтовано необхідність розробки нових підходів до оптимізації профілактичних заходів. Сформульовано основні задачі дослідження та визначено предмет та об'єкт дослідження.

У спеціальному розділі розроблено математичну модель управління ризиками нещасних випадків. Проведено аналіз та обґрунтування вибору генетичного алгоритму як методу оптимізації. Реалізовано програмне забезпечення з використанням алгоритмів штучного інтелекту, що дозволяє автоматизувати процес вибору оптимального комплексу профілактичних заходів.

У тестувально-аналітичному розділі проведено експериментальні дослідження розробленої програми на конкретному прикладі з різними варіаціями вхідних даних. Проаналізовано отримані результати та оцінено ефективність запропонованого підходу.

Практична цінність роботи полягає у можливості використання запропонованого підходу на виробничих підприємствах для підвищення рівня безпеки праці та оптимізації витрат на профілактичні заходи. Розроблена програма може бути використана як інструмент підтримки прийняття рішень у сфері управління ризиками.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Березюк О.В., Лемешев М.С. Безпека життєдіяльності − Вінницький національний технічний університет – URL: https://web.posibnyky.vntu.edu.ua/fmbt/berezyuk_bezpeka_zhittyediyalnosti/index.htm
2. Безпека життєдіяльності та цивільний захист − [Електронний ресурс]: підручник для студ. спеціальностей з природничих, соціально-гуманітарних наук та інженерно-комунікаційних технологій / О. Г. Левченко, О. В. Землянська, Н. А. Праховнік, В. В. Зацарний; КПІ ім. Ігоря Сікорського. – Електронні текстові данні. – Київ: КПІ ім. Ігоря Сікорського, 2019. – 267 с.
3. Carsten Magiera. Von der Unfallanalyse zur Prävention - Sicherheitsingenieur, 2024 − https://www.sifa-sibe.de/arbeitssicherheit/unfallgeschehen/von-der-unfallanalyse-zur-praevention/.
4. Безпека життєдіяльності. Основи охорони праці : текст Б40 лекцій / О. Г. Янчик, В. В. Горбенко, С. В. Котлярова та ін. – Харків : НТУ «ХПІ», 2016. – 164 с.
5. Hollnagel, E. Barriers and Accident Prevention – 1st Edition, London, 2004 – 242 p.
6. Safety Management and Risk Assessment in Industrial Operations International Journal of Industrial Engineering and Management, 2021. URL: https://www.ieomsociety.org/journals/
7. Sustaining Organizational Outcomes in Manufacturing Firms: The Role of HRM and Occupational Health and Safety MDPI, 2024 – URL: https://www.mdpi.com/2071-1050/16/3/1035
8. Bundesanstalt für Arbeitsschutz und Arbeitsmedizin (BAuA) − Ratgeber zur Gefährdungsbeurteilung Handbuch für Arbeitsschutzfachleute Ratgeber zur

Gefährdungsbeurteilung Handbuch für Arbeitsschutzfachleut, Berlin, 2016, 256 S.

9. Fostering a Safety Culture in Manufacturing Industry through Safety Behavior: A Structural Equation Modelling Approach – Journal of Safety and Sustainability, 2024.
10. James T. Tweedy Introduction to Hazard Control Management: A Vital Organizational Function, publisher: Taylor & Francis Inc; 1st edition, 2013, 288 p.
11. Top 5 Safety Hazards in the Manufacturing Industry, Evotix, July 2022, URL: https://www.evotix.com/resources/blog/top-safety-hazards-in-the-manufacturing-industry
12. Top 10 Most Common Safety Hazards In Manufacturing, Adam Lynch April, 2024, URL: https://www.imectechnologies.com/2024/04/17/hazards-in-manufacturing/.
13. Коць О.О. – Класифікація ризиків промислового підприємства − Національний університет «Львівська політехніка» Підручник. – 2-ге вид.; доп. і перероб. − К.: КНЕУ, 2004. – 468 с
14. Safety and Risk Management in Manufacturing Facilities – Process Equipment & Controls, URL: https://processequipmentandcontrols.com/safety-and-risk-management-in-manufacturing-facilities/
15. Вишневська В.А., Ніколаєв І. В. – Класифікація ризиків промислових підприємств як передумова управління ними − Братислава – Харків, ВШЕМ – ХНЕУ ім. С. Кузнеця, 2020.
16. Калініченко З.Д. – «Ризик-менеджмент: навчальний посібник для здобувачів спец. 051 «Економіка» та 073 «Менеджмент»» − Дніпро: ДДУВС, 2021. 224 с
17. Holger Sommerfeld Strategic Risk Management, Risikomanagagement und Controling, 2018

18. Атаманчук П.С., Мендерецький В. В., Панчук О. П. Чорна О. Г – Основи охорони праці − Безпека життєдіяльності. Навч. посіб. – К.: Центр учбової літератури, 2011. – 276 с.
19. Imoh Antai, Roland Hellberg – Characterizing the defense industry for risk management: a systems approach − Journal of Defense Analytics and Logistics, 2024
20. Швець Ю.О. Ризики в діяльності промислових підприємств: види, методи оцінки та заходи подолання ризику / Ю.О. Швець // Науковий вісник Ужгородського національного університету. Серія: Міжнародні економічні відносини та світове господарство. – 2018. – № 17. Ч. 2. – С. 131–135.
21. Hossein Abedsoltan − Future of process safety: Insights, approaches, and potential developments − University of Kansas, March 2024
22. Hopkin, P. Fundamentals of Risk Management: Understanding, Evaluating and Implementing Effective Risk Management - London: Kogan Page, 2017.
23. Chizubem Benson − The impact of interventions on health, safety and environment in the process industry − European University Cyprus, December 2023
24. Перелік заходів та засобів з охорони праці, витрати на здійснення та придбання яких включаються до валових витрат − Урядовий портал – Єдиний веб-портал органів виконавчої влади України – URL: https://www.kmu.gov.ua/npas/1316161
25. Manuele, F.A. On the Practice of Safety - Hoboken: John Wiley & Sons, 2008
26. Leon Altomonte − A Guide to Risk Reduction URL: https://safetyculture.com/topics/risk-management/risk-reduction/
27. Шелудько Г.А., Науменко В.В., Стрельнікова О.О. Методи розв’язання задач оптимізації: Конспект лекцій. − Харків: УкрДАЗТ, 2014. – 50 с.

28. Глибовець М.М., Олецький О.В., Системи штучного інтелекту, навчальний посібник, Національний транспортний університет, Київ – 94 с.
29. О.В. Бондаренко, О.В. Устиненко, В.І. Сєриков, Метаевристичні алгоритми. Метафори-стратегії, оглядова стаття, НТУ «Харківський політехнічний університет», 2023
30. Я. Пиріг, М. Климаш, Ю. Пиріг, О. Лаврів Генетичний алгоритм як засіб розв'язання оптимізаційних задач – Національний університет «Львівська політехніка», 2023− Електронний ресурс: https://science.lpnu.ua/sites/default/files/journal-paper/2023/oct/31568/paper10.pdf
31. Мартинова О.В, Стєпанова К. В., Генетичний алгоритм для розв'язання оптимізаційних задач, оглядова стаття, Харківський національний екномічний університет ім. С. Кузнеця, Харків, 2019
32. Johansson R. Numerical Python: A Practical Techniques Approach for Industry. Apress,2015. 512 p.
33. Л.Є. Ковалев, Медведєва М. О., Побережець І.І. Аналіз можливостей бібліотек Python при вивченні курсу «Математичне програмування», публікація в iТехніка Наука, 2024 Електронний ресурс: https://www.researchgate.net/publication/380493256_Analiz_mozlivostej_bibliotek_Python_pri_vivcenni_kursu_Matematicne_programu-vanna
34. Інформаційні технології: наука, техніка, технологія, освіта, здоров'я: тези доповідей XXXI міжнародної науково-практичної конференції MicroCAD-2023, 17–20 травня 2023 р. / за ред. проф. Сокола Є.І. Харків: НТУ «ХПІ». — 1406 с
35. Електронний ресурс: MEALPY (MEta-heuristic ALgorithms in PYthon) − https://mealpy.readthedocs.io/en/latest/pages/models/mealpy.evolutionary_based.html#module-mealpy.evolutionary_based.GA

ДОДАТКИ

ДОДАТОК А

Відомість матеріалів кваліфікаційної роботи

№ з/п	Позначення	Найменування	Кількість аркушів	Примітки
1				
2		Документація		
3				
4	САУ.КР.24.09.ПЗ	Пояснювальна записка	88	Формат А4
5				
6	САУ.КР.24.09.ДМ	Демонстраційний матеріал	12	Презентація на CD-R
7				
8	САУ.КР.24.09.КР	Копія роботи	1	Диск CD-R
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				

					САУ.КР.24.09.ДА.ПЗ.			
Змін.	Аркуш	№ докум.	Підпис	Дата				
Розроб.		Кодола Я.О.			**Матеріали кваліфікаційної роботи**	Літ.	Аркуш	Аркушів
К. розд.		Коряшкіна Л.С.						
Керівн.		Коряшкіна Л.С.				НТУ «ДП», 12; 124-20-1		
Н.контр.		Хом’як Т. В.						
Зав. каф.		Желдак Т. А.						

ДОДАТОК Б

Відгук
на кваліфікаційну роботу бакалавра
студентки групи *124 – 20 – 1*
спеціальності *124 Системний аналіз*

Тема кваліфікаційної роботи: _____
_____
Обсяг кваліфікаційної роботи _____ стор.
Мета кваліфікаційної роботи: _____
_____
Актуальність теми _____
_____
_____
Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності бакалавра спеціальності 124 Системний аналіз, оскільки _____
_____
_____
Виконані в кваліфікаційній роботі завдання відповідають вимогам ступеня бакалавра. Оригінальність наукових рішень полягає в _____
_____
_____
_____
Практичне значення результатів кваліфікаційної роботи полягає в _____
_____
_____
Висновки підтверджують можливість використання результатів роботи в _____
_____
_____
Оформлення пояснювальної записки та демонстраційного матеріалу до неї виконано згідно з вимогами. Роботу виконано самостійно, відповідно до завдання та у повному обсязі (*в разі невідповідності – вказати*)
У роботі відзначено такі недоліки: _____
_____
_____
Кваліфікаційна робота в цілому заслуговує оцінки:_____

З урахуванням висловлених зауважень автор (не) заслуговує присвоєння освітньої кваліфікації «бакалавр з системного аналізу».

Керівник кваліфікаційної роботи бакалавра,
науковий ступінь, вчене звання, посада _____ / ПІБ

ДОДАТОК В

Код програми

```
import customtkinter as ctk
from data_entry_page import DataEntryPage
from data_store import DataStore
from entered_data_page import EnteredDataPage
from file_upload_page import FileUploadPage
from home_page import HomePage
from new_data_entry_page import NewDataEntryPage
from solution_page import SolutionPage
class App(ctk.CTk):
    def __init__(self):
        super().__init__()
        # Налаштування основного вікна
        self.title("Оптимізація ризиків")
        self.geometry("740x580")  # Збільшення розмірів вікна
        self.data_store = DataStore()  # Ініціалізація сховища даних
        # Створення контейнера для розміщення фреймів
        container = ctk.CTkFrame(self)
        container.pack(fill="both", expand=True)
        self.frames = {}  # Словник для зберігання фреймів
        for F in (HomePage, DataEntryPage, NewDataEntryPage,
FileUploadPage, EnteredDataPage, SolutionPage):
            # Створення кожного фрейму та додавання його в словник
            frame = F(parent=container, controller=self,
data_store=self.data_store)
            self.frames[F.__name__] = frame
            frame.grid(row=0, column=0, sticky="nsew")
        self.show_frame("HomePage")  # Показ початкового фрейму
    def show_frame(self, page_name):
        # Підняти фрейм на передній план
        frame = self.frames[page_name]
        frame.tkraise()
if __name__ == "__main__":
    # Запуск програми
```

```
    app = App()
    app.mainloop()

class DataStore:
    def __init__(self):
        # Ініціалізація всіх необхідних параметрів для зберігання даних
        self.max_severity = None  # Максимальна тяжкість наслідків
        self.max_risk = None  # Максимально допустимий ризик
        self.num_factors = 0  # Кількість небезпечних чинників
        self.probabilities = []  # Ймовірності настання небезпечних чинників
        self.severities = []  # Тяжкість наслідків небезпечних чинників
        self.measures = []  # Кількість профілактичних заходів для кожного
чинника
        self.A = []  # Вартість профілактичних заходів
        self.P_ai = []  # Зниження ймовірності для кожного профілактичного
заходу
        self.T_ai = []  # Зниження тяжкості для кожного профілактичного
заходу
        self.factors_data = []  # Дані про небезпечні чинники
        self.factors_names = []  # Назви небезпечних чинників
        self.measures_names = []  # Назви профілактичних заходів
        self.input_method = 1  # Метод введення даних: 1 - вручну, 0 - з
файлу

    def set_parameters(self, max_severity, max_risk, num_factors):
        # Встановлення параметрів задачі
        self.max_severity = max_severity
        self.max_risk = max_risk
        self.num_factors = num_factors
    def set_initial_data(self, probabilities, severities, measures,
factors_names):
        # Встановлення початкових даних про небезпечні чинники
        self.probabilities = probabilities
        self.severities = severities
        self.measures = measures
        self.factors_names = factors_names
    def get_initial_data(self):
        # Отримання початкових даних про небезпечні чинники
        return self.probabilities, self.severities, self.measures
    def set_measures_data(self, A, P_ai, T_ai, measures_names):
        # Встановлення даних про профілактичні заходи
```

```
        self.A = A
        self.P_ai = P_ai
        self.T_ai = T_ai
        self.measures_names = measures_names
    def set_factors_data(self, factors_data):
        # Встановлення даних про небезпечні чинники
        self.factors_data = factors_data
    def get_measures_data(self):
        # Отримання даних про профілактичні заходи
        return self.A, self.P_ai, self.T_ai
    def get_num_factors(self):
        # Отримання кількості небезпечних чинників
        return self.num_factors
    def get_factors_names(self):
        # Отримання назв небезпечних чинників
        return self.factors_names
    def set_input_method(self, method):
        # Встановлення методу введення даних
        self.input_method = method
    def get_input_method(self):
        # Отримання методу введення даних
        return self.input_method

import customtkinter as ctk
import tkinter as tk

class HomePage(ctk.CTkFrame):
    def __init__(self, parent, controller, data_store):
        super().__init__(parent)
        # Ініціалізація головної сторінки
        self.controller = controller
        self.data_store = data_store
        # Створення та налаштування елементів інтерфейсу
        label = ctk.CTkLabel(self, text="Вітаємо!", font=("Arial", 24,
"bold"))
        label.pack(pady=10)
        info = ("Дана програма допоможе вам обрати комплекс профілактичних
заходів\n для зменшення ризику настання "
                "небезпечних чинників на виробництві, з умовою мінімізації
витрат на заходи.\n\n"
```

```
                "Для проведення розрахунків, вам потрібно надати необхідні
дані,\n це можна зробити завантаживши файл "
                "або ввести дані вручну.\n\n"
                "Оберіть спосіб надання даних:")
        # Відображення інформаційного тексту
        info_text = tk.Text(self, wrap="word", font=("Arial", 18), height=10,
width=50, bd=0, bg="#D3D3D3", fg="black")
        info_text.insert("1.0", info)
        info_text.config(state="disabled")
        info_text.tag_add("center", "1.0", "end")
        info_text.tag_configure("center", justify='center')
        info_text.pack(pady=10, padx=10, fill="both", expand=True)
        # Створення кнопок для переходу до інших сторінок
        button_frame = ctk.CTkFrame(self, fg_color="#D3D3D3")
        button_frame.pack(pady=10)
        data_entry_button = ctk.CTkButton(button_frame, text="Ввести дані",
command=lambda: controller.show_frame("DataEntryPage"))
        data_entry_button.pack(side="left", padx=5)
        file_upload_button = ctk.CTkButton(button_frame, text="Завантажити
файл", command=lambda: controller.show_frame("FileUploadPage"))
        file_upload_button.pack(side="left", padx=5)
        # Центрування фрейму
        self.place(relx=0.5, rely=0.5, anchor="center")

import customtkinter as ctk
from tkinter import filedialog

class FileUploadPage(ctk.CTkFrame):
    def __init__(self, parent, controller, data_store):
        super().__init__(parent)
        self.controller = controller
        self.data_store = data_store
        # Створення заголовка сторінки
        label = ctk.CTkLabel(self, text="Завантажити файл", font=("Arial",
24, "bold"))
        label.pack(pady=10)
        # Інструкції для користувача щодо формату файлу
        instruction_label = ctk.CTkLabel(self, text="Для завантаження файлу
вам необхідно сформувати його за наступними правилами:\n\n"
                                                 "1. Перший рядок: одне
число, яке визначає максимальне значення тяжкості наслідків.\n"
```

```
                                                    "2. Другий рядок: одне
число, яке визначає максимально допустиме значення ризиків.\n"
                                                    "3. Третій рядок: одне
ціле число, яке визначає кількість небезпечних чинників.\n"
                                                    "4. Четвертий рядок:
початкові значення ймовірності настання кожного небезпечного чинника,
розділені пробілом. Кількість значень повинна відповідати кількості
небезпечних чинників.\n"
                                                    "5. П'ятий рядок:
початкові значення тяжкості наслідків для кожного небезпечного чинника,
розділені пробілом.\n"
                                                    "6. Шостий рядок:
порожній.\n"
                                                    "7. З сьомого рядка
починається інформація про профілактичні заходи. Інформація представлена у
вигляді трьох умовних масивів, розділених між собою порожніми рядками. "
                                                    "Кожен рядок в масиві
відповідає конкретному небезпечному чиннику, а числа в рядку відповідають
кількості профілактичних заходів для цього чинника. Числа в рядку повинні
бути розділені пробілами. "
                                                    "Якщо це десятичне число,
використовуйте крапку як розділовий знак.",
                                    justify="left", wraplength=500)
        instruction_label.pack(pady=10)
        # Створення фрейму для кнопок
        button_frame = ctk.CTkFrame(self, fg_color="#D3D3D3")
        button_frame.pack(side="bottom", pady=10)
        # Кнопка для повернення на головну сторінку
        back_button      =      ctk.CTkButton(button_frame,      text="Назад",
command=lambda: self.controller.show_frame("HomePage"))
        back_button.grid(row=0, column=1, padx=5)
        # Кнопка для завантаження файлу
        self.upload_button  =  ctk.CTkButton(button_frame,  text="Завантажити
файл", command=self.upload_file)
        self.upload_button.grid(row=0, column=0, padx=5)
        # Мітка для виводу повідомлень про помилки
        self.error_label = ctk.CTkLabel(self, text="", text_color="red")
        self.error_label.pack(pady=5)
    def upload_file(self):
        file_path   =   filedialog.askopenfilename(filetypes=[("Text   files",
"*.txt")])
```

```
        if file_path:
            try:
                # Читання вмісту файлу
                with open(file_path, 'r') as file:
                    lines = file.readlines()
                    # Перевірка кількості рядків у файлі
                    if len(lines) < 6:
                        raise ValueError("Файл містить недостатню кількість
рядків.")
                    # Читання основних параметрів з файлу
                    max_severity = float(lines[0].strip())
                    max_risk = float(lines[1].strip())
                    num_factors = int(lines[2].strip())
                    probabilities = list(map(float,
lines[3].strip().split()))
                    severities = list(map(float, lines[4].strip().split()))
                    # Перевірка відповідності кількості значень ймовірностей
та тяжкості кількості небезпечних чинників
                    if len(probabilities) != num_factors or len(severities)
!= num_factors:
                        raise ValueError("Кількість значень ймовірностей та
тяжкості не відповідає кількості небезпечних чинників.")
                    # Ініціалізація масивів для профілактичних заходів
                    A = []
                    P_ai = []
                    T_ai = []
                    factors_names = [f"Чинник {i+1}" for i in
range(num_factors)]
                    measures_names = []
                    current_line = 6
                    # Читання даних про профілактичні заходи для кожного
небезпечного чинника
                    for i in range(num_factors):
                        factor_A = list(map(float,
lines[current_line].strip().split()))
                        factor_P_ai = list(map(float, lines[current_line +
1].strip().split()))
                        factor_T_ai = list(map(float, lines[current_line +
2].strip().split()))
                        # Перевірка відповідності кількості значень між
масивами
```

```
                        if len(factor_A) != len(factor_P_ai) or len(factor_A)
!= len(factor_T_ai):
                            raise        ValueError("Кількість        значень
профілактичних заходів не відповідає між масивами A, P_ai, T_ai.")

                        A.append(factor_A)
                        P_ai.append(factor_P_ai)
                        T_ai.append(factor_T_ai)
                        measures_names.append([f"Захід {i+1}.{j+1}" for j in
range(len(factor_A))])
                        current_line += 4
                    # Збереження параметрів у DataStore
                    self.data_store.set_parameters(max_severity,     max_risk,
num_factors)
                    self.data_store.set_initial_data(probabilities,
severities, [len(a) for a in A], factors_names)
                    self.data_store.set_measures_data(A,      P_ai,      T_ai,
measures_names)
                    # Виведення повідомлення про успішне завантаження файлу
                    self.error_label.configure(text="Файл              успішно
завантажено", text_color="green")
                    self.controller.show_frame("EnteredDataPage")
                    self.controller.frames['EnteredDataPage'].update_table()
# Оновлення таблиці
            except Exception as e:
                # Виведення повідомлення про помилку завантаження файлу
                self.error_label.configure(text=f"Помилка завантаження файлу:
{str(e)}", text_color="red")

import customtkinter as ctk
import tkinter as tk

class DataEntryPage(ctk.CTkFrame):
    def __init__(self, parent, controller, data_store):
        super().__init__(parent)
        self.controller = controller
        self.data_store = data_store
        self.entries = []
        # Створення заголовка сторінки
        label  =  ctk.CTkLabel(self,  text="Ввід  даних",  font=("Arial",  24,
"bold"))
```

```
        label.pack(pady=10)
        # Створення фрейму для введення основних параметрів
        input_frame = ctk.CTkFrame(self)
        input_frame.pack(pady=10)
        # Поле введення максимального значення тяжкості наслідків
        max_severity_label = ctk.CTkLabel(input_frame, text="Максимальне
значення тяжкості наслідків", font=("Arial", 14))
        max_severity_label.grid(row=0, column=0, padx=5, pady=5, sticky="w")
        self.max_severity_entry = ctk.CTkEntry(input_frame, font=("Arial",
14), border_color="black")
        self.max_severity_entry.grid(row=0, column=1, padx=5, pady=5)
        # Поле введення максимального допустимого значення ризику
        max_risk_label = ctk.CTkLabel(input_frame, text="Максимальне
допустиме значення ризику", font=("Arial", 14))
        max_risk_label.grid(row=1, column=0, padx=5, pady=5, sticky="w")
        self.max_risk_entry = ctk.CTkEntry(input_frame, font=("Arial", 14),
border_color="black")
        self.max_risk_entry.grid(row=1, column=1, padx=5, pady=5)
        # Поле введення кількості небезпечних чинників
        num_factors_label = ctk.CTkLabel(input_frame, text="Кількість
небезпечних чинників", font=("Arial", 14))
        num_factors_label.grid(row=2, column=0, padx=5, pady=5, sticky="w")
        self.num_factors_entry = ctk.CTkEntry(input_frame, font=("Arial",
14), border_color="black")
        self.num_factors_entry.grid(row=2, column=1, padx=5, pady=5)
        # Мітка для відображення помилок
        self.error_label = ctk.CTkLabel(self, text="", font=("Arial", 12),
text_color="red")
        self.error_label.pack(pady=5)
        # Створення фрейму для кнопок
        self.button_frame = ctk.CTkFrame(self, fg_color="#D3D3D3")
        self.button_frame.pack(side="bottom", pady=10)
        # Кнопка для повернення на головну сторінку
        self.back_button = ctk.CTkButton(self.button_frame, text="Назад",
command=lambda: controller.show_frame("HomePage"))
        self.back_button.pack(side="left", padx=5)
        # Кнопка для очищення полів введення
        self.clear_button = ctk.CTkButton(self.button_frame, text="Очистити
поля", command=self.clear_fields)
        self.clear_button.pack(side="left", padx=5)
        # Кнопка для переходу до введення детальних даних
```

```
        self.next_button = ctk.CTkButton(self.button_frame, text="Далі",
command=self.validate_and_create_grid)
        self.next_button.pack(side="left", padx=5)
        self.grid_frame = None
        self.new_next_button = None
    def validate_float(self, entry, min_value=None, max_value=None):
        value = entry.get()
        try:
            float_value = float(value)
            if min_value is not None and float_value < min_value:
                raise ValueError
            if max_value is not None and float_value > max_value:
                raise ValueError
            entry.configure(border_color="black")
            return True
        except ValueError:
            entry.configure(border_color="red")
            return False
    def validate_integer(self, entry):
        value = entry.get()
        if value.isdigit():
            entry.configure(border_color="black")
            return True
        else:
            entry.configure(border_color="red")
            return False
    def validate_and_create_grid(self):
        is_valid = True
        # Перевірка правильності введення максимального значення тяжкості
наслідків
        if not self.validate_float(self.max_severity_entry):
            is_valid = False
        # Перевірка правильності введення максимального допустимого значення
ризику
        if not self.validate_float(self.max_risk_entry):
            is_valid = False
        # Перевірка правильності введення кількості небезпечних чинників
        if not self.validate_integer(self.num_factors_entry):
            is_valid = False
        if is_valid:
            # Збереження параметрів у DataStore
```

```
            self.data_store.set_parameters(
                float(self.max_severity_entry.get()),
                float(self.max_risk_entry.get()),
                int(self.num_factors_entry.get())
            )
            self.create_grid()
            self.error_label.configure(text="")
            self.next_button.pack_forget()
            if not self.new_next_button:
                self.new_next_button    =    ctk.CTkButton(self.button_frame,
text="Далі", command=self.validate_grid)
                self.new_next_button.pack(side="left", padx=5)
        else:
            self.error_label.configure(text="Неправильно введені значення")
    def create_grid(self):
        if self.grid_frame:
            self.grid_frame.destroy()
        num_factors = self.data_store.get_num_factors()
        self.max_severity = self.data_store.max_severity
        self.grid_frame = ctk.CTkFrame(self)
        self.grid_frame.pack(pady=10)
        headers     =     ["Небезпечний\nчинник",     "Ймовірність\nнастання",
"Тяжкість\nнаслідків", "Кількість\nпрофілактичних\nзаходів"]
        for col, header in enumerate(headers):
            label = ctk.CTkLabel(self.grid_frame, text=header, font=("Arial",
14, "bold"))
            label.grid(row=0, column=col, padx=5, pady=5, sticky="n")
        self.entries = []
        for row in range(1, num_factors + 1):
            row_entries = []
            for col in range(4):
                if col == 0:
                    entry = ctk.CTkEntry(self.grid_frame, font=("Arial", 14),
width=200, border_color="black")
                else:
                    entry = ctk.CTkEntry(self.grid_frame, font=("Arial", 14),
width=70, border_color="black")
                entry.grid(row=row, column=col, padx=5, pady=5)
                row_entries.append(entry)
            self.entries.append(row_entries)
    def validate_grid(self):
```

```
        is_valid = True
        probabilities = []
        severities = []
        measures = []
        factors_names = []
        for row_entries in self.entries:
            factor_name = row_entries[0].get()
            probability = row_entries[1].get()
            severity = row_entries[2].get()
            num_measures = row_entries[3].get()
            if not self.validate_float(row_entries[1], min_value=0,
max_value=1):
                is_valid = False
            if not self.validate_float(row_entries[2], min_value=0,
max_value=self.max_severity):
                is_valid = False
            if not self.validate_integer(row_entries[3]):
                is_valid = False
            if factor_name and probability and severity and num_measures:
                probabilities.append(float(probability))
                severities.append(float(severity))
                measures.append(int(num_measures))
                factors_names.append(factor_name)
            else:
                is_valid = False
        if is_valid:
            # Збереження початкових даних у DataStore
            self.data_store.set_initial_data(probabilities, severities,
measures, factors_names)

self.controller.frames['NewDataEntryPage'].populate_grid(self.entries)
            self.data_store.set_input_method(1)  # Введення даних вручну
            self.controller.show_frame("NewDataEntryPage")
            self.error_label.configure(text="")
        else:
            self.error_label.configure(text="Неправильно введені значення")
    def clear_fields(self):
        # Очищення полів введення
        self.max_severity_entry.delete(0, tk.END)
        self.max_risk_entry.delete(0, tk.END)
        self.num_factors_entry.delete(0, tk.END)
```

```
        self.max_severity_entry.configure(border_color="black")
        self.max_risk_entry.configure(border_color="black")
        self.num_factors_entry.configure(border_color="black")
        self.error_label.configure(text="")
        if self.grid_frame:
            self.grid_frame.destroy()
            self.grid_frame = None
        if self.new_next_button:
            self.new_next_button.pack_forget()
            self.new_next_button = None
        self.next_button.pack(side="left", padx=5)
        self.back_button.pack(side="left", padx=5)

import customtkinter as ctk
from work_with_grid import WorkWithGrid

class NewDataEntryPage(ctk.CTkFrame):
    def __init__(self, parent, controller, data_store):
        super().__init__(parent)
        self.controller = controller
        self.data_store = data_store
        # Створення заголовка сторінки
        label = ctk.CTkLabel(self, text="Введіть інформацію", font=("Arial",
24, "bold"))
        label.pack(pady=10)
        # Створення фрейму для сітки введення даних
        self.grid_frame = ctk.CTkFrame(self)
        self.grid_frame.pack(fill="both", expand=True)
        # Створення фрейму для кнопок
        button_frame = ctk.CTkFrame(self, fg_color="#D3D3D3")
        button_frame.pack(pady=10)
        # Кнопка для повернення на попередню сторінку
        self.back_button = ctk.CTkButton(button_frame, text="Назад",
command=lambda: self.controller.show_frame("DataEntryPage"))
        self.back_button.grid(row=0, column=0, padx=5)
        # Кнопка для очищення полів введення
        self.clear_button = ctk.CTkButton(button_frame, text="Очистити поля",
command=self.clear_entries)
        self.clear_button.grid(row=0, column=1, padx=5)
        # Кнопка для збереження введених даних
```

```
        self.save_button = ctk.CTkButton(button_frame, text="Зберегти",
command=self.validate_and_save)
        self.save_button.grid(row=0, column=2, padx=5)
        # Мітка для відображення помилок
        self.error_label = ctk.CTkLabel(self, text="", text_color="red")
        self.error_label.pack(pady=5)
    def populate_grid(self, data_entries):
        # Очистка попередніх введених даних з сітки
        for widget in self.grid_frame.winfo_children():
            widget.destroy()
        # Створення заголовків стовпців
        headers = ["Небезпечний чинник", "Профілактичні заходи", "Вартість",
"Зменшення \n ймовірності", "Зменшення \nтяжкості"]
        for col, header in enumerate(headers):
            label = ctk.CTkLabel(self.grid_frame, text=header, font=("Arial",
14, "bold"))
            label.grid(row=0, column=col, padx=5, pady=5, sticky="n")
        self.entries = []
        current_row = 1
        # Заповнення сітки даними
        for row_entries in data_entries:
            factor_name = row_entries[0].get()
            num_measures = int(row_entries[3].get())
            for j in range(num_measures):
                grid_row_entries = []
                if j == 0:
                    label = ctk.CTkLabel(self.grid_frame, text=factor_name,
font=("Arial", 14))
                    label.grid(row=current_row, column=0, padx=5, pady=5,
sticky="w")
                else:
                    label = ctk.CTkLabel(self.grid_frame, text="",
font=("Arial", 14))
                    label.grid(row=current_row, column=0, padx=5, pady=5,
sticky="w")
                measure_entry = ctk.CTkEntry(self.grid_frame, font=("Arial",
14), border_color="black")
                measure_entry.grid(row=current_row, column=1, padx=5, pady=5)
                grid_row_entries.append(measure_entry)
                cost_entry = ctk.CTkEntry(self.grid_frame, font=("Arial",
14), border_color="black")
```

```
                cost_entry.grid(row=current_row, column=2, padx=5, pady=5)
                grid_row_entries.append(cost_entry)
                probability_entry       =       ctk.CTkEntry(self.grid_frame,
font=("Arial", 14), border_color="black")
                probability_entry.grid(row=current_row,    column=3,    padx=5,
pady=5)
                grid_row_entries.append(probability_entry)
                severity_entry = ctk.CTkEntry(self.grid_frame, font=("Arial",
14), border_color="black")
                severity_entry.grid(row=current_row,     column=4,     padx=5,
pady=5)
                grid_row_entries.append(severity_entry)
                self.entries.append(grid_row_entries)
                current_row += 1
    def validate_and_save(self):
        all_valid = True
        A, P_ai, T_ai = [], [], []
        factors_names = []
        measures_names = []
        num_factors = self.data_store.num_factors
        for i in range(num_factors):
            start_idx = sum(self.data_store.measures[:i])
            end_idx = start_idx + self.data_store.measures[i]
            factor_A = []
            factor_P_ai = []
            factor_T_ai = []
            factor_measures_names = []
            for j in range(start_idx, end_idx):
                measure_entry = self.entries[j][0]
                cost_entry = self.entries[j][1]
                probability_entry = self.entries[j][2]
                severity_entry = self.entries[j][3]
                # Перевірка правильності введених даних
                if not WorkWithGrid.validate_float(cost_entry):
                    all_valid = False
                if not WorkWithGrid.validate_float(probability_entry, 0, 1):
                    all_valid = False
                if    not    WorkWithGrid.validate_float(severity_entry,    0,
self.data_store.max_severity):
                    all_valid = False
                factor_A.append(float(cost_entry.get()))
```

```
                factor_P_ai.append(float(probability_entry.get()))
                factor_T_ai.append(float(severity_entry.get()))
                factor_measures_names.append(measure_entry.get())
            A.append(factor_A)
            P_ai.append(factor_P_ai)
            T_ai.append(factor_T_ai)
            measures_names.append(factor_measures_names)
        # Збереження назв небезпечних чинників
        for row_entries in self.entries:
            factor_name = row_entries[0].get()
            if factor_name:
                factors_names.append(factor_name)
        if all_valid:
            # Збереження даних у DataStore
            self.data_store.set_measures_data(A, P_ai, T_ai, measures_names)
            self.controller.frames['EnteredDataPage'].update_table()         #
Оновлення таблиці
            self.controller.show_frame("EnteredDataPage")
            self.error_label.configure(text="")
        else:
            self.error_label.configure(text="Неправильно введені дані")
            error_label   =   ctk.CTkLabel(self.grid_frame,   text="Неправильно
введені дані", text_color="red")
            error_label.grid(row=len(self.entries) + 1, columnspan=5, pady=5)
    def clear_entries(self):
        # Очищення всіх полів введення
        WorkWithGrid.clear_entries(self.entries)

import customtkinter as ctk
from optimization import Optimization
import tkinter as tk
from progress_window import ProgressWindow

class EnteredDataPage(ctk.CTkFrame):
    def __init__(self, parent, controller, data_store):
        super().__init__(parent)
        self.controller = controller
        self.data_store = data_store
        # Створення заголовка сторінки
        label  =  ctk.CTkLabel(self,  text="Введені  дані",  font=("Arial",  24,
"bold"))
```

```
        label.pack(pady=10)
        # Створення фрейму для відображення суми ризиків
        self.risk_label_frame = ctk.CTkFrame(self)
        self.risk_label_frame.pack(fill="x", pady=5)
        self.risk_label    =    ctk.CTkLabel(self.risk_label_frame,    text="",
font=("Arial", 14, "bold"))
        self.risk_label.pack(pady=10)
        # Створення контейнера для таблиці з даними
        self.table_container = ctk.CTkFrame(self, width=1000, height=600)
        self.table_container.pack(pady=10, fill="both", expand=True)
        self.canvas = tk.Canvas(self.table_container, width=1000, height=600)
        self.canvas.pack(side=tk.LEFT, fill=tk.BOTH, expand=True)
        self.scrollbar_y         =         tk.Scrollbar(self.table_container,
orient="vertical", command=self.canvas.yview)
        self.scrollbar_y.pack(side=tk.RIGHT, fill="y")
        self.scrollbar_x         =         tk.Scrollbar(self.table_container,
orient="horizontal", command=self.canvas.xview)
        self.scrollbar_x.pack(side=tk.BOTTOM, fill="x")
        self.canvas.configure(yscrollcommand=self.scrollbar_y.set,
xscrollcommand=self.scrollbar_x.set)
        self.table_frame = ctk.CTkFrame(self.canvas)
        self.canvas.create_window((0,       0),       window=self.table_frame,
anchor="nw")
        self.table_frame.bind("<Configure>", self.on_frame_configure)
        # Створення фрейму для кнопок
        button_frame = ctk.CTkFrame(self, fg_color="#D3D3D3")
        button_frame.pack(pady=10)
        # Кнопка для повернення на попередню сторінку
        self.back_button    =    ctk.CTkButton(button_frame,    text="Назад",
command=self.go_back)
        self.back_button.pack(side="left", padx=5)
        # Кнопка для запуску оптимізації
        self.update_button    =    ctk.CTkButton(button_frame,    text="Пошук
рішення", command=self.solve_optimization)
        self.update_button.pack(side="left", padx=5)
        # Центрування фрейму
        button_frame.pack(anchor="center")
    def on_frame_configure(self, event):
        # Налаштування області прокрутки
        self.canvas.configure(scrollregion=self.canvas.bbox("all"))
    def update_table(self):
```

```
        # Очищення попереднього вмісту таблиці
        for widget in self.table_frame.winfo_children():
            widget.destroy()
        # Розрахунок загального ризику
        total_risk = sum(prob * sev for prob, sev in
zip(self.data_store.probabilities, self.data_store.severities))
        self.risk_label.configure(text=f"На даному етапі сума ризиків
становить: {total_risk:.2f}")
        # Відображення заголовків таблиці
        headers = ["Небезпечний чинник", "Початкова ймовірність", "Початкова
тяжкість", "Початкові ризики", "Профілактичні заходи", "Вартість",
"Ймовірність", "Тяжкість"]
        for col, header in enumerate(headers):
            label = ctk.CTkLabel(self.table_frame, text=header,
font=("Arial", 10, "bold"))
            label.grid(row=0, column=col, padx=2, pady=1, sticky="n")
        current_row = 1
        for i, factor_name in enumerate(self.data_store.factors_names):
            initial_risk = self.data_store.probabilities[i] *
self.data_store.severities[i]
            for j in range(len(self.data_store.A[i])):
                if j == 0:
                    factor_label = ctk.CTkLabel(self.table_frame,
text=factor_name, font=("Arial", 10))
                    factor_label.grid(row=current_row, column=0, padx=2,
pady=1, sticky="w")
                    prob_label = ctk.CTkLabel(self.table_frame,
text=self.data_store.probabilities[i], font=("Arial", 10))
                    prob_label.grid(row=current_row, column=1, padx=2,
pady=1, sticky="w")
                    sev_label = ctk.CTkLabel(self.table_frame,
text=self.data_store.severities[i], font=("Arial", 10))
                    sev_label.grid(row=current_row, column=2, padx=2, pady=1,
sticky="w")
                    risk_label = ctk.CTkLabel(self.table_frame,
text=f"{initial_risk:.2f}", font=("Arial", 10))
                    risk_label.grid(row=current_row, column=3, padx=2,
pady=1, sticky="w")
                else:
                    ctk.CTkLabel(self.table_frame, text="", font=("Arial",
10)).grid(row=current_row, column=0, padx=2, pady=1, sticky="w")
```

```
                    ctk.CTkLabel(self.table_frame,   text="",   font=("Arial",
10)).grid(row=current_row, column=1, padx=2, pady=1, sticky="w")
                    ctk.CTkLabel(self.table_frame,   text="",   font=("Arial",
10)).grid(row=current_row, column=2, padx=2, pady=1, sticky="w")
                    ctk.CTkLabel(self.table_frame,   text="",   font=("Arial",
10)).grid(row=current_row, column=3, padx=2, pady=1, sticky="w")
                measure_label         =         ctk.CTkLabel(self.table_frame,
text=self.data_store.measures_names[i][j], font=("Arial", 10))
                measure_label.grid(row=current_row, column=4, padx=2, pady=1,
sticky="w")
                cost_label          =          ctk.CTkLabel(self.table_frame,
text=self.data_store.A[i][j], font=("Arial", 10))
                cost_label.grid(row=current_row,  column=5,  padx=2,  pady=1,
sticky="w")
                prob_ai_label         =         ctk.CTkLabel(self.table_frame,
text=self.data_store.P_ai[i][j], font=("Arial", 10))
                prob_ai_label.grid(row=current_row, column=6, padx=2, pady=1,
sticky="w")
                sev_ai_label         =         ctk.CTkLabel(self.table_frame,
text=self.data_store.T_ai[i][j], font=("Arial", 10))
                sev_ai_label.grid(row=current_row, column=7, padx=2, pady=1,
sticky="w")
                current_row += 1
    def go_back(self):
        # Перехід на попередню сторінку залежно від методу введення даних
        if self.data_store.get_input_method() == 1:
            self.controller.show_frame("DataEntryPage")
        else:
            self.controller.show_frame("FileUploadPage")
    def solve_optimization(self):
        # Створення вікна прогресу
        self.progress_window = ProgressWindow(self, total_iterations=300)
        self.progress_window.grab_set()
        # Запуск оптимізації
        optimizer               =               Optimization(self.data_store,
parent=self.progress_window)
        solution = optimizer.solve()
        self.progress_window.close()
        # Відкриття нової сторінки SolutionPage з рішенням
        self.controller.frames['SolutionPage'].show_solution()
        self.controller.show_frame('SolutionPage')
```

```
import numpy as np
from mealpy.evolutionary_based.GA import BaseGA
from mealpy.utils.problem import FloatVar
from progress_window import ProgressWindow

class Optimization:
    def __init__(self, data_store, parent=None):
        #Ініціалізує клас Optimization з даними зі сховища та визначає необхідні змінні.
        self.data_store = data_store
        self.parent = parent
        # Ініціалізація даних зі сховища
        self.A = self.data_store.A
        self.P_ai = self.data_store.P_ai
        self.T_ai = self.data_store.T_ai
        self.P_start = self.data_store.probabilities
        self.K_start = self.data_store.severities
        self.N = self.data_store.num_factors
        self.R_max = self.data_store.max_risk
        # Підготовка даних у плоский формат для зручності обробки
        self.A_flat = [item for sublist in self.A for item in sublist]
        self.P_ai_flat = [item for sublist in self.P_ai for item in sublist]
        self.T_ai_flat = [item for sublist in self.T_ai for item in sublist]
        # Визначення індексів для доступу до елементів плоских масивів
        self.start_indices = np.cumsum([0] + [len(sublist) for sublist in self.A[:-1]])
        self.end_indices = np.cumsum([len(sublist) for sublist in self.A])
    def calculate_risk(self, solution):
        """Розраховує ризики для кожного фактору на основі рішення.
        :return: Сума ризиків, ймовірності та тяжкості для кожного фактору, а також окремі ризики"""

        P_kin = []
        K_kin = []
        R_kin = []
        for i in range(self.N):
            start_idx = self.start_indices[i]
            end_idx = self.end_indices[i]
            # Розрахунок кінцевої ймовірності
            P_kin_i = max(0.03, self.P_start[i] * (
```

```
                    1       -       np.sum(solution[start_idx:end_idx]       *
np.array(self.P_ai_flat[start_idx:end_idx])))
            P_kin.append(P_kin_i)
            # Розрахунок кінцевої тяжкості
            K_kin_i  =  max(1,  (self.K_start[i]  -  (self.K_start[i]  -  1)  *
np.sum(
                solution[start_idx:end_idx]                                    *
np.array(self.T_ai_flat[start_idx:end_idx])))
            K_kin.append(K_kin_i)
            # Розрахунок ризику
            R_kin.append(P_kin_i * K_kin_i)
        total_risk = np.sum(R_kin)
        return total_risk, P_kin, K_kin, R_kin
    def objective_function(self, solution):

        """Визначає об'єктивну функцію для оптимізації.
        :return: Загальна вартість рішення"""
        solution = np.array([1 if x > 0.5 else 0 for x in solution])
        total_cost = np.sum(solution * self.A_flat)
        total_risk, R_kin = self.calculate_risk(solution)[:2]
        if total_risk > self.R_max:
            total_cost += 1e1000
        for i in range(self.N):
            if R_kin[i] > self.R_max:
                total_cost += 1e1000
        return total_cost,
    def solve(self):
        """Запускає процес оптимізації з використанням генетичного алгоритму
та повертає оптимальне рішення.
        :return: Бінарна матриця з оптимальними значеннями профілактичних
заходів"""

        problem_dict = {
            "bounds":   FloatVar(lb=[0]   *   len(self.A_flat),   ub=[1]   *
len(self.A_flat), name="x"),
            "obj_func": self.objective_function,
            "minmax": "min",
        }
        model = BaseGA(epoch=300, pop_size=100, pc=0.9, pm=0.05)
        progress_window = ProgressWindow(self.parent, total_iterations=300)
        progress_window.grab_set()
```

```
        def progress_callback(epoch=300):
            progress_window.update_progress(epoch)
        model.callback = lambda epoch: progress_callback(epoch)
        g_best = model.solve(problem_dict)
        progress_window.close()
        binary_solution = [1 if x > 0.5 else 0 for x in g_best.solution]
        total_risk,          P_kin,          K_kin,          R_kin          =
self.calculate_risk(binary_solution)
        if total_risk > self.R_max:
            # Заповнення бінарної матриці одиницями, якщо ризик перевищує
допустимий рівень
            binary_solution = [1] * len(self.A_flat)
        binary_matrix = []
        for i in range(self.N):
            start_idx = self.start_indices[i]
            end_idx = self.end_indices[i]
            binary_matrix.append(binary_solution[start_idx:end_idx])
        return binary_matrix
    def final_values(self, flat_solution):
        #return: Сума ризиків, ймовірності та тяжкості для кожного фактору, а
також окремі ризики
        return self.calculate_risk(flat_solution)

import customtkinter as ctk

class ProgressWindow(ctk.CTkToplevel):
    def __init__(self, parent, total_iterations):
        super().__init__(parent)
        self.title("Пошук рішення")
        self.geometry("300x100")
        self.progress_label    =    ctk.CTkLabel(self,    text="Пошук    рішення
триває...", font=("Arial", 12))
        self.progress_label.pack(pady=10)
        self.progress_bar = ctk.CTkProgressBar(self, width=250)
        self.progress_bar.pack(pady=10)
        self.progress_bar.set(0)
        self.total_iterations = total_iterations
    def update_progress(self, epoch):
        progress = (epoch + 1) / self.total_iterations
        self.progress_bar.set(progress)
        self.update_idletasks()
```

```
    def close(self):
        self.destroy()

import tkinter.filedialog as fd
from report_generator import ReportGenerator
from optimization import Optimization
import customtkinter as ctk
import tkinter as tk

class SolutionPage(ctk.CTkFrame):
    # Ініціалізація сторінки з рішенням
    def __init__(self, parent, controller, data_store):
        super().__init__(parent)
        self.controller = controller
        self.data_store = data_store
        # Встановлення заголовку сторінки
        label = ctk.CTkLabel(self, text="Рекомендований комплекс
профілактичних заходів", font=("Arial", 24, "bold"))
        label.pack(pady=10)
        # Відображення різних показників
        self.cost_label = ctk.CTkLabel(self, text="", font=("Arial", 14))
        self.cost_label.pack(pady=10)
        self.risk_label = ctk.CTkLabel(self, text="", font=("Arial", 14))
        self.risk_label.pack(pady=10)
        self.efficiency_label = ctk.CTkLabel(self, text="", font=("Arial",
14))
        self.efficiency_label.pack(pady=10)
        self.warning_label = ctk.CTkLabel(self, text="", font=("Arial", 14,
"bold"), text_color="red")
        self.warning_label.pack(pady=10)
        # Створення фрейму для таблиці з прокруткою
        self.table_container = ctk.CTkFrame(self, width=1000, height=200)  #
Зменшення висоти таблиці
        self.table_container.pack(pady=10, fill="both", expand=True)
        self.canvas = tk.Canvas(self.table_container, width=1000, height=200)
# Зменшення висоти таблиці
        self.canvas.pack(side=tk.LEFT, fill=tk.BOTH, expand=True)
        self.scrollbar_y = tk.Scrollbar(self.table_container,
orient="vertical", command=self.canvas.yview)
        self.scrollbar_y.pack(side=tk.RIGHT, fill="y")
```

```
        self.scrollbar_x = tk.Scrollbar(self.table_container,
orient="horizontal", command=self.canvas.xview)
        self.scrollbar_x.pack(side=tk.BOTTOM, fill="x")
        self.canvas.configure(yscrollcommand=self.scrollbar_y.set,
xscrollcommand=self.scrollbar_x.set)
        self.table_frame = ctk.CTkFrame(self.canvas)
        self.canvas.create_window((0, 0), window=self.table_frame,
anchor="nw")
        self.table_frame.bind("<Configure>", self.on_frame_configure)
        # Створення кнопок для навігації
        button_frame = ctk.CTkFrame(self, fg_color="#D3D3D3")
        button_frame.pack(pady=10)
        self.back_button = ctk.CTkButton(button_frame, text="Назад",
command=lambda: controller.show_frame("EnteredDataPage"))
        self.back_button.grid(row=0, column=0, padx=5)
        self.save_button = ctk.CTkButton(button_frame, text="Зберегти звіт",
command=self.save_report)
        self.save_button.grid(row=0, column=1, padx=5)
        self.home_button = ctk.CTkButton(button_frame, text="Головна
сторінка", command=lambda: controller.show_frame("HomePage"))
        self.home_button.grid(row=0, column=2, padx=5)
        # Центрування фрейму
        button_frame.pack(anchor="center")
    #Оновлення області прокрутки при зміні розміру таблиці
    def on_frame_configure(self, event):
        self.canvas.configure(scrollregion=self.canvas.bbox("all"))
    #Відображення рішення на сторінці
    def show_solution(self, solution):
        for widget in self.table_frame.winfo_children():
            widget.destroy()
        total_cost = 0
        for i in range(len(self.data_store.A)):
            for j in range(len(self.data_store.A[i])):
                if solution[i][j] == 1:
                    total_cost += self.data_store.A[i][j]
        self.cost_label.configure(text=f"Загальна вартість проведення
профілактичних заходів: {total_cost:.2f}")
        optimizer = Optimization(self.data_store, parent=self)
        flat_solution = [item for sublist in solution for item in sublist]
        total_risk, P_kin, K_kin, R_kin =
optimizer.final_values(flat_solution)
```

```
        self.risk_label.configure(text=f"Ризик настання нещасного випадку: {total_risk:.2f}")
        if total_risk > self.data_store.max_risk:
            self.warning_label.configure(text="Увага! Найдене рішення перевищує можливе допустиме значення ризиків.\n Наданих й запланованих профілактичних заходів недостатньо для виконання умови!")
        else:
            self.warning_label.configure(text="")
        initial_total_risk = sum(self.data_store.probabilities[i] * self.data_store.severities[i] for i in range(self.data_store.num_factors))
        efficiency = ((initial_total_risk - total_risk) / initial_total_risk) * 100
        self.efficiency_label.configure(text=f"Ефективність проведення обраних заходів: {efficiency:.2f}%")
        headers = [
            "Небезпечний\nчинник",
            "Нове значення\nймовірності",
            "Нове значення\nтяжкості",
            "Нове значення\nризику",
            "Рекомендовані\nпрофілактичні заходи"
        ]
        for col, header in enumerate(headers):
            label = ctk.CTkLabel(self.table_frame, text=header, font=("Arial", 14, "bold"), justify="center")
            label.grid(row=0, column=col, padx=2, pady=5, sticky="n")  # Збільшення відступу між рядками
        current_row = 1
        for i, factor_name in enumerate(self.data_store.factors_names):
            factor_label = ctk.CTkLabel(self.table_frame, text=factor_name, font=("Arial", 12))
            factor_label.grid(row=current_row, column=0, padx=2, pady=5, sticky="w")  # Збільшення відступу між рядками
            prob_label = ctk.CTkLabel(self.table_frame, text=f"{P_kin[i]:.2f}", font=("Arial", 12))
            prob_label.grid(row=current_row, column=1, padx=2, pady=5, sticky="w")  # Збільшення відступу між рядками
            sev_label = ctk.CTkLabel(self.table_frame, text=f"{K_kin[i]:.2f}", font=("Arial", 12))
            sev_label.grid(row=current_row, column=2, padx=2, pady=5, sticky="w")  # Збільшення відступу між рядками
```

```
            risk_label            =            ctk.CTkLabel(self.table_frame,
text=f"{R_kin[i]:.2f}", font=("Arial", 12))
            risk_label.grid(row=current_row,    column=3,    padx=2,    pady=5,
sticky="w")  # Збільшення відступу між рядками
            measures   =   [self.data_store.measures_names[i][j]   for   j   in
range(len(self.data_store.measures_names[i])) if solution[i][j] == 1]
            measures_text = ", ".join(measures)
            measures_label          =          ctk.CTkLabel(self.table_frame,
text=measures_text, font=("Arial", 12))
            measures_label.grid(row=current_row,  column=4,  padx=2,  pady=5,
sticky="w")  # Збільшення відступу між рядками
            current_row += 1
        self.solution = solution
        self.P_kin = P_kin
        self.K_kin = K_kin
        self.R_kin = R_kin
        self.total_risk = total_risk
        self.efficiency = efficiency
        self.total_cost = total_cost
    #Збереження звіту у вигляді текстового файлу
    def save_report(self):
        file_path       =       fd.asksaveasfilename(defaultextension=".txt",
filetypes=[("Text files", "*.txt")])
        if file_path:
            report_generator = ReportGenerator(
                self.data_store,
                self.solution,
                self.P_kin,
                self.K_kin,
                self.R_kin,
                self.total_risk,
                self.efficiency,
                self.total_cost
            )
            report_generator.save_report(file_path)

class ReportGenerator:
    def __init__(self, data_store, solution, P_kin, K_kin, R_kin, total_risk,
efficiency, total_cost):
        self.data_store = data_store
        self.solution = solution
```

```
        self.P_kin = P_kin
        self.K_kin = K_kin
        self.R_kin = R_kin
        self.total_risk = total_risk
        self.efficiency = efficiency
        self.total_cost = total_cost
    def generate_report(self):
        initial_total_risk = sum(self.data_store.probabilities[i] *
self.data_store.severities[i] for i in range(self.data_store.num_factors))
        report_lines = [
            "Для запровадження небезпечних чинників було оптимізовано та
обрано комплекс профілактичних заходів для зниження ризику нещасного випадку
на підприємстві з умовою мінімізації фінансових витрат на них"
        ]
        if self.total_risk <= self.data_store.max_risk:
            for i, factor_name in enumerate(self.data_store.factors_names):
                measures = [self.data_store.measures_names[i][j] for j in
range(len(self.data_store.measures_names[i])) if self.solution[i][j] == 1]
                measures_text = ", ".join(measures)
                initial_risk = self.data_store.probabilities[i] *
self.data_store.severities[i]
                report_lines.append(
                    f"Для небезпечного чинника: {factor_name} рекомендовано
застосувати профілактичні заходи: {measures_text}. "
                    f"Після їх запровадження початковий ризик
{initial_risk:.2f} зменшується до {self.R_kin[i]:.2f}."
                )
            report_lines.append(
                f"В кінцевому результаті після проведення профілактичних
заходів загальний ризик настання нещасного випадку зменшується з
{initial_total_risk:.2f} до {self.total_risk:.2f}, що є меншим за допустиме.
"
                f"Ефективність проведення заходів {self.efficiency:.2f}%.
Загальна сума необхідних витрат для проведення рекомендованих профілактичних
заходів: {self.total_cost:.2f}."
            )
        else:
            report_lines.append(
                "Для зниження ризиків настання нещасного випадку
запропонованих профілактичних заходів недостатньо. Рекомендується переглянути
надані дані й задати інші або додаткові профілактичні заходи."
```

```
            )
            for i, factor_name in enumerate(self.data_store.factors_names):
                measures = [f"Захід {i+1}.{j+1}" for j in
range(len(self.data_store.measures_names[i]))]
                measures_text = ", ".join(measures)
                initial_risk = self.data_store.probabilities[i] *
self.data_store.severities[i]
                final_risk = self.P_kin[i] * self.K_kin[i]
                report_lines.append(
                    f"Для небезпечного чинника: {factor_name} рекомендовано
застосувати профілактичні заходи: {measures_text}. "
                    f"Після їх запровадження початковий ризик
{initial_risk:.2f} зменшується до {final_risk:.2f}."
                )
            report_lines.append(
                f"В кінцевому результаті після проведення профілактичних
заходів загальний ризик настання нещасного випадку зменшується з
{initial_total_risk:.2f} до {self.total_risk:.2f}, що є більшим за допустиме.
"
                f"Ефективність проведення заходів {self.efficiency:.2f}%.
Загальна сума необхідних витрат для проведення рекомендованих профілактичних
заходів: {self.total_cost:.2f}."
            )
        return "\n\n".join(report_lines)
    def save_report(self, file_path):
        report = self.generate_report()
        with open(file_path, 'w', encoding='utf-8') as file:
            file.write(report)

class WorkWithGrid:
    @staticmethod
    def validate_float(entry, min_value=None, max_value=None):
        value = entry.get()
        try:
            float_value = float(value)
            if min_value is not None and float_value < min_value:
                raise ValueError
            if max_value is not None and float_value > max_value:
                raise ValueError
            entry.configure(border_color="black")
            return True
```

```
    except ValueError:
        entry.configure(border_color="red")
        return False
@staticmethod
def validate_integer(entry):
    value = entry.get()
    if value.isdigit():
        entry.configure(border_color="black")
        return True
    else:
        entry.configure(border_color="red")
        return False

@staticmethod
def clear_entries(entries):
    for row_entries in entries:
        for entry in row_entries:
            entry.delete(0, "end")
            entry.configure(border_color="black")
```