МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут
комп'ютерних наук та управління проектами
(повне найменування інституту, назва факультету)

Програмного забезпечення автоматизованих систем
(повна назва кафедри)

Пояснювальна записка
до кваліфікаційної (магістерської) роботи

за темою: Удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програмного забезпечення для його реалізації

Виконав: студент 6 курсу, групи 6151м
спеціальності
121 «Інженерія програмного забезпечення»
(шифр і назва спеціальності)

Литовченко О.В.
(підпис, прізвище та ініціали)
Керівник Фаріонова Т.А.
(підпис, прізвище та ініціали)
Рецензент Приходько С.Б.
(підпис, прізвище та ініціали)
Завідувач кафедри Приходько С. Б.
(підпис, прізвище та ініціали)

м. Миколаїв – 2020 р.

Міністерство освіти і науки України
Національний університет кораблебудування
імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Освітній ступень Магістр
Галузь 12 «Інформаційні технології»
(шифр і назва)
Спеціальність 121 «Інженерія програмного забезпечення»
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри

" 26 " 10 2020 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ (МАГІСТЕРСЬКУ) РОБОТУ СТУДЕНТУ

Литовченку Олексію Володимировичу
(прізвище, ім'я, по батькові)

1.Тема магістерської роботи: Удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програмного забезпечення для його реалізації
керівник роботи Латанська Людмила Олексіївна, к.ф.-м. н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " 26 " жовтня 2020 року №1037-уч
2. Строк подання студентом роботи 01.12.2020 року
3. Вихідні дані до роботи

4.Зміст магістерської роботи (МР):
• Титульний аркуш, завдання на кваліфікаційну (магістерську) роботу, реферат (українською, англійською), зміст, перелік умовних позначень, символів, одиниць та термінів (за необхідності).
• Вступ (Актуальність теми. Зв'язок роботи з науковими програмами, планами, темами. Мета і завдання дослідження. Об'єкт дослідження. Предмет дослідження. Методи дослідження. Наукова новизна одержаних результатів. Практичне значення одержаних результатів. Особистий внесок здобувача. Апробація результатів досліджень**.** Публікації.)
• Огляд літератури за темою, обґрунтування необхідності проведення досліджень за обраною темою, вибір напрямків досліджень, мета дослідження, основні задачі дослідження
• Викладення результатів власних досліджень з висвітленням того нового, що пропонується
• Проект програмного забезпечення
• Результати досліджень та розробки проекту програмного забезпечення
• Організаційно-економічний розділ Розрахунок економічної ефективності від розробки та впровадження програмного забезпечення
• Розділи з охорони праці та охорони навколишнього середовища Охорона праці, Охорона навколишнього середовища
• Висновки
• Список використаних джерел
• Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма і методика випробувань програмного забезпечення)

5. Перелік графічного матеріалу

_____
_____
_____
_____
_____
_____

6. Консультанти розділів магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання___

КАЛЕНДАРНИЙ ПЛАН

Назва етапів кваліфікаційної (магістерської) роботи (МР)	Термін виконання	Примітка
1. Підготовка вступної частини МР	18.10.2020	
2. Підготовка розділу (ів) МР з огляду літератури за темою, обґрунтування необхідності проведення досліджень за обраною темою, вибір напрямів досліджень	19.10.2020	
3. Підготовка розділу (ів) МР з результатів власних досліджень	22.10.2020	
4. Підготовка розділу МР з проекту програмного забезпечення	16.11.2020	
5. Підготовка організаційно-економічного розділу	18.11.2020	
6. Підготовка розділу з охорони праці	20.11.2020	
7. Підготовка розділу з охорони навколишнього середовища	23.11.2020	
8. Підготовка розділу МР – Висновки	25.11.2020	
9. Оформлення списку використаних джерел	27.11.2020	
10. Оформлення додатків	30.11.2020	
11. Підготовка презентації МР та доповіді	01.12.2020	
12. Подання МР на попередній захист	01.12.2020	
13. Подання МР рецензенту	11.12.2020	
14. Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, разом з заявою щодо самостійності виконання роботи та ідентичності друкованої та електронної версії роботи	14.12.2020	
15. Подання на кафедру ПЗАС електронних версії наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК від 19.05.2020 р. за №287-уч); презентації доповіді	18.12.2020	
16. Подання на кафедру ПЗАС письмового відгуку та рецензії на кваліфікаційну роботу	18.12.2020	

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Литовченко Олексій Володимирович

Удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програмного забезпечення для його реалізації

Кваліфікаційна робота на здобуття освітнього рівня магістра зі спеціальності 121 – «Інженерія програмного забезпечення». Національний університет кораблебудування імені адмірала Макарова. Миколаїв, 2020 р.

Обсяг роботи: 104 стор., 18 табл., 20 рис., 16 використаних джерел, 5 додатків.

Актуальність теми. При розробці програмних проектів одного класу найбільший вплив на трудомісткість має їх розмір. Проте в інженерії програмного забезпечення не існує єдиних рекомендацій для оцінювання розміру програмних продуктів. Тому задача удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програмного забезпечення для його реалізації є актуальною та має практичну цінність.

Мета і завдання дослідження. Метою даної кваліфікаційної роботи є підвищення достовірності оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програми для її реалізації.

Для досягнення поставленої мети необхідно вирішити такі завдання: виконати аналіз існуючих методів та моделей оцінювання розміру ПЗ; обґрунтувати необхідність удосконалення рівняння регресії; нормалізувати отримані емпіричні дані; побудувати нелінійне рівняння регресії, довірчий інтервал та інтервал передбачення на основі нормалізованих даних без викидів; розробити програму для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin**.**

Об’єкт дослідження: процес оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Предмет дослідження: нелінійне рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Методи дослідження: для вирішення поставлених задач були застосовані методи теорії ймовірності, математичної статистики та регресійного аналізу.

Наукова новизна одержаних результатів: удосконалено рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Практичне значення отриманих результатів полягає в розробці алгоритму і ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin на основі нелінійного регресійного рівняння з нормалізуючим перетворенням у вигляді десяткового логарифму.

Ключові слова: ОЦІНЮВАННЯ РОЗМІРУ, НЕЛІНІЙНЕ РІВНЯННЯ РЕГРЕСІЇ, НОРМАЛІЗУЮЧЕ ПЕРЕТВОРЕННЯ

ABSTRACT

Lytovchenko Oleksii

Improving the regression equation for estimating the size of open source software on Kotlin and developing software for its implementation.

Qualification work for obtaining a master's degree in Specialty 121 - Software Engineering. Admiral Makarov National University of Shipbuilding. Mykolaiv, 2020.

The work contains: 104 pages, 18 tables, 20 figures, 16 references, 5 appendices.

Relevance of the topic: When developing software projects of one class, their size has the greatest impact on complexity. However, in software engineering there are no uniform recommendations for estimating the size of software products. Therefore, the task of improving the regression equation for estimating the size of open source software on Kotlin and developing software for its implementation is relevant and has practical value.

Purpose and tasks of the research. The purpose of this qualification work is to increase the reliability of estimating the size of open source software on Kotlin and develop a program for its implementation.

To achieve this goal it is necessary to solve the following tasks: perform an analysis of existing methods and models for estimating the size of software; justify the need to improve the regression equation; normalize the obtained empirical data; construct a nonlinear regression equation, confidence interval and prediction interval based on normalized data without emissions, develop a program to estimate the size of open source software on Kotlin.

Object of research: the process of estimating the size of open source software on Kotlin.

Subject of research: the process of estimating the size of open source software on Kotlin.

Research methods: methods of probability theory, mathematical statistics and regression analysis were used to solve the problems.

Scientific novelty of the obtained results: improved the regression equation for estimating the size of open source software on Kotlin.

The practical value of the results. The obtained results are the development of an algorithm and software for estimating the size of open source software on Kotlin on the basis of a nonlinear regression equation with a normalizing transformation in the form of a decimal logarithm.

Keywords: SIZE ESTIMATION, NONLINEAR REGRESSION EQUATION, NORMALIZING TRANSFORMATION

ЗМІСТ

ВСТУП

При розробці програмних проектів виконавці вирішують і задачі оцінювання. Перш за все, це задачі оцінювання трудомісткості і ресурсів, необхідних для успішної реалізації програмного забезпечення.

При розробці програмних проектів одного класу найбільший вплив на трудомісткість має їх розмір [1].

Проте в інженерії програмного забезпечення не існує єдиних рекомендацій для оцінювання розміру програмних продуктів. Серед існуючих методів та моделей, які використовуються для отримання такої оцінки, можна виділити [1]:

- метод конструктивної вартості;
- метод функціонально-бальної оцінки;
- експертне оцінювання;
- нейронні мережі;
- нечітку логіку;
- регресійний аналіз;
- комбіновані методи та інш.

Але більшість з них стають непридатними до використання через інший клас задач, відсутність даних, ресурсів або експертних навичок в цій галузі.

Тому тематика оцінювання розміру програмного забезпечення в повній мірі ще не розкрита. І тому задача удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програмного забезпечення для його реалізації є актуальною та має практичну цінність.

Мета і завдання дослідження. Метою даної кваліфікаційної роботи є підвищення достовірності оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програми для його реалізації.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– виконати аналіз існуючих методів та моделей оцінювання розміру ПЗ;

– обґрунтувати необхідність удосконалення рівняння регресії для оцінювання розміру ПЗ з відкритим кодом на Kotlin;

– нормалізувати отримані емпіричні дані, використовуючи перетворення у вигляді десяткового логарифму;

– перевірити дані на викиди;

– побудувати лінійне рівняння регресії , довірчий інтервал та інтервал передбачення для вихідних даних;

– побудувати нелінійне рівняння регресії, довірчий інтервал та інтервал передбачення на основі нормалізованих даних;

– розробити програму для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin**.**

Об’єкт дослідження: процес оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Предмет дослідження: нелінійне рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Методи дослідження. Для вирішення поставлених задач були застосовані методи теорії ймовірності, математичної статистики та регресійного аналізу.

Наукова новизна одержаних результатів: удосконалено рівняння для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Практичне значення отриманих результатів полягає в розробці алгоритму і ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin на основі нелінійного рівняння регресії з нормалізуючим перетворенням у вигляді десяткового логарифму.

Особистий внесок здобувача. Проведені дослідження та отримані результати, наведені в кваліфікаційній роботі, одержані здобувачем особисто.

1 ХАРАКТЕРИСТИКА МОВИ ПРОГРАМУВАННЯ KOTLIN ТА ОГЛЯД ІСНУЮЧИХ МЕТОДІВ І МОДЕЛЕЙ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Мова програмування Kotlin як мова для створення мобільних і веб-додатків

Kotlin – це статично типізована мова програмування від компанії JetBrains (2016). Kotlin можна використовувати для створення мобільних і веб-додатків.

Kotlin працює поверх віртуальної машини Java та при компіляції компілюється в байткод. Тобто додаток на Kotlin можна запускати всюди, де встановлена віртуальна машина Java. Також можна компілювати код в JavaScript і запускати в браузері. Програми на Kotlin можуть працювати під Windows, Linux, Mac OS, iOS, Android [2].

Kotin розвивається як opensource, вихідний код проекту можна подивитися в репозиторії на github за адресою https://github.com/JetBrains/kotlin/ [2].

Kotlin зазнав впливу таких мов: Java, Scala, Groovy, C #, JavaScript, Swift. Kotlin підтримує об'єктно-орієнтоване та процедурне програмування. Він має ясний і зрозумілий синтаксис і досить легкий для навчання [2].

Найпопулярнішим напрямком використання Kotlin є розробка під ОС Android. Компанія Google обрала Kotlin в якості пріоритетної мови для розробок під ОС Android. Вданій сфері він поступово замінює мову Java.

1.2 Метрики програмного забезпечення

Існує чимало різноманітних метрик для оцінювання програмного забезпечення. Метрика програмного забезпечення– це міра, що дозволяє отримати чисельне значення деякої властивості програмного забезпечення

або його специфікацій [3]. Часто метрики ПЗ порівнюють з методами та моделями для оцінювання розміру, витрат і зусиль. Оскільки кількісні методи добре зарекомендували себе в інших галузях, теоретики і практики інформатики намагаються перенести даний підхід і в розробку програмного забезпечення. Набір метрик може включати в себе [4]:

- кількість рядків коду;
- цикломатичну складність;
- аналіз функціональних точок;
- кількість помилок на 1000 рядків коду;
- ступінь покриття коду тестуванням;
- покриття вимог;
- кількість класів і інтерфейсів і ін.

Найпоширенішою метрикою вихідного коду, що відображає розмір програмного проекту, є показник кількості рядків коду. Ця метрика має багато похідних метрик (кількість пустих рядків; кількість рядків, що містять коментарі і т.д.), що дозволяють оцінити різні показники проекту [4]. Як правило, кількість рядків коду програми є негаусівською випадковою величиною.

1.3 Огляд існуючих методів і моделей оцінювання розміру програмного забезпечення

Існують різні класифікації методів та моделей оцінювання розміру програмного забезпечення. Згідно однієї з них методи та моделі оцінювання розміру програмного забезпечення відносяться до наступних класів [1]:

1) методи та моделі експертної оцінки;

2) методи оцінки по математичним моделям.

Експертні оцінки застосовуються при відсутності дискретних емпіричних даних. Фактично єдиним обгрунтуванням оцінки є авторитет експерта (або групи експертів) [1].

До методів оцінки по математичним моделям відносяться [1]:

- нейронні мережі;
- нечітка логіка;
- генетичні алгоритми;
- регресійний аналіз та інш.

Не дивлячись на існуюче різноманіття моделей, сьогодні їм бракує точності та універсальності.

Розглянемо регресійні моделі оцінювання розміру програмного забезпечення. Існують лінійні та нелінійні моделі оцінювання розміру. В лінійних моделях, використовуючи емпіричні дані, методом найменших квадратів знаходять коефіцієнти рівняння регресії. За допомогою отриманого рівняння виконують прогнозування значень розміру. Під час побудови нелінійних регресійних моделей використовуються два підходи. Або методом перебору знаходять нелінійну функцію регресії та на її основі будують нелінійну регресійну модель, або знаходять такі попередні перетворення випадкових змінних регресії (нормалізуючі перетворення), які дозволяють отримати близький до лінійного зв’язок між нормалізованими змінними.

1.4 Обгрунтування необхідності проведення досліджень

Розмір є важливим показником програмного забезпечення. Інформація, що отримується при оцінюванні розміру, може бути використана для оцінювання трудомісткості та ресурсів проектів на ранніх стадіях розробки.

Проблемі оцінювання розміру програмного забезпечення присвячено багато наукових праць. Та незважаючи на інтенсивні дослідження, залишається ряд невирішених аспектів проблеми.

Швидкий розвиток інформаційних технологій, підвищення складності та розміру програмного забезпечення, поява нових підходів до розробки програмного забезпечення, нові мови програмування, жорстка конкуренція, – все це висуває підвищені вимоги до точності та оперативності оцінювання програмних проектів на ранніх стадіях розробки.

Із аналізу методів та моделей для оцінювання розміру програмних проектів випливає, що існуючі на сьогоднішній день методи та математичні моделі мають ряд недоліків, що не дозволяє в повному обсязі оцінити розмір програмного забезпечення з одночасним урахуванням особливостей мови програмування Kotlin. Також в реальних проектах майже неможливо отримати дані, які будуть розподілені за нормальним законом, і зв’язки між цими даними не завжди є лінійними. Отже є необхідність удосконалити математичну модель оцінювання розміру програмного забезпечення, яка повинна враховувати нелінійність зв’язків між емпіричними даними та використовувати їх нормалізацію для покращення достовірності передбачення та швидкої і надійної оцінки розміру розробки на мові програмування Kotlin.

2 УДОСКОНАЛЕННЯ РІВНЯННЯ РЕГРЕСІЇ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВІДКРИТИМ КОДОМ НА KOTLIN

2.1 Один із підходів до розробки удосконаленого рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin

При побудові удосконаленого рівняння регресії був використаний наступний підхід:

– виконаний аналіз існуючих методів та моделей оцінювання розміру ПЗ;

– обґрунтована необхідність удосконалення рівняння регресії для оцінювання розміру ПЗ з відкритим кодом на Kotlin;

– досліджені різні джерела з відкритим вихідним кодом та зібрані дані для удосконалення рівняння;

– виконана нормалізація отриманих емпіричних даних, використовуючи перетворення у вигляді десяткового логарифму;

– виконана перевірка даних на викиди;

– побудуване лінійне рівняння регресії, довірчий інтервал та інтервал передбачення для вихідних даних;

– побудуване нелінійне рівняння регресії, довірчий інтервал та інтервал передбачення на основі нормалізованих даних.

Теоретична цінність роботи полягає у подальшому розвитку математично коректного, достатньо ефективного та перспективного підходу, що грунтується на застосуванні нелінійної регресії, для розв’язування задачі оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin. Практична цінність роботи полягає в можливості використання підходу до

оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin як ефективного засобу відшукання розміру програмного забезпечення.

Для отримання передбачення розміру програмного забезпечення, довірчих інтервалів та інтервалів передбачення будемо використовувати наступні емпіричні дані:

– кількість функцій;

– кількість строк коду.

В роботі визначаються метрики якості лінійного регресійного рівняння, отриманого для вихідних даних у припущенні про нормальність їх розподілу, метрики удосконаленого нелінійного регресійного рівняння та виконується порівняння отриманих результатів.

2.2 Удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin

Для побудови нелінійного регресійного рівняння для оцінювання розміру програмного забезпечення відкритим кодом на Kotlin за допомогою власної написаної програми та з використання сховища програмного забезпечення із відкритим кодом GitHub були зібрані вихідні дані для проведення дослідження, а саме: кількість функцій (X) та кількість строк коду (Y).

Розмір програмного забезпечення розраховується як функція від одного фактору:

$$Y = f(X) \tag{2.1}$$

Так як віхідні дані не підпорядковуються нормальному закону розподілу (визначили по χ^2), то для використання лінійної регресійної теорії виконується нормалізація [5,6]. В якості нормалізуючого перетворення обрано десятковий логарифм. В результаті було отримано нормалізовані величини Z_Y та Zx.

Емпіричні дані та їх нормалізовані значення без викидів наведені в таблиці 2.1.

Таблиця 2.1 – Емпіричні та нормалізовані дані

№	Y	X	Z_Y	Z_X
1	30	7	1,4771	0,8451
2	33	2	1,5185	0,3010
3	18	1	1,2553	0,0000
4	30	2	1,4771	0,3010
5	74	7	1,8692	0,8451
6	43	8	1,6335	0,9031
7	85	8	1,9294	0,9031
8	100	23	2,0000	1,3617
9	43	3	1,6335	0,4771
10	29	2	1,4624	0,3010
11	24	3	1,3802	0,4771
12	58	11	1,7634	1,0414
13	40	3	1,6021	0,4771
14	29	3	1,4624	0,4771
15	45	4	1,6532	0,6021
16	75	8	1,8751	0,9031
17	20	3	1,3010	0,4771
18	25	3	1,3979	0,4771
19	58	5	1,7634	0,6990
20	80	11	1,9031	1,0414
21	20	2	1,3010	0,3010
22	45	11	1,6532	1,0414
23	45	6	1,6532	0,7782
24	25	1	1,3979	0,0000
25	63	9	1,7993	0,9542
26	28	3	1,4472	0,4771
27	13	1	1,1139	0,0000
28	52	6	1,7160	0,7782
29	51	10	1,7076	1,0000
30	121	14	2,0828	1,1461

Перевірка нормалізованих даних на викиди здійснювалася з використанням квадрату відстані Махаланобіса D^2, критерію χ^2, тестової статистики T_S та критерію Фішера F [6].

Остаточні результати перевірки на викиди наведені в таблиці 2.2.

Таблиця 2.2 – Результат перевірки на викиди

№	Zx	Zy	D^2	χ^2	T_S	F
1	0,8451	1,4771	1,5006	5,9915	0,7010	3,3404
2	0,3010	1,5185	2,3975	5,9915	1,1201	3,3404
3	0,0000	1,2553	4,3269	5,9915	2,0215	3,3404
4	0,3010	1,4771	2,1973	5,9915	1,0266	3,3404
5	0,8451	1,8692	1,3714	5,9915	0,6407	3,3404
6	0,9031	1,6335	1,6393	5,9915	0,7658	3,3404
7	0,9031	1,9294	1,5244	5,9915	0,7122	3,3404
8	1,3617	2,0000	5,1542	5,9915	2,4080	3,3404
9	0,4771	1,6335	1,5989	5,9915	0,7470	3,3404
10	0,3010	1,4624	2,1284	5,9915	0,9944	3,3404
11	0,4771	1,3802	0,8640	5,9915	0,4037	3,3404
12	1,0414	1,7634	2,3859	5,9915	1,1147	3,3404
13	0,4771	1,6021	1,4886	5,9915	0,6955	3,3404
14	0,4771	1,4624	1,0639	5,9915	0,4970	3,3404
15	0,6021	1,6532	1,1511	5,9915	0,5378	3,3404
16	0,9031	1,8751	1,5094	5,9915	0,7052	3,3404
17	0,4771	1,3010	0,7065	5,9915	0,3301	3,3404
18	0,4771	1,3979	0,9040	5,9915	0,4223	3,3404
19	0,6990	1,7634	1,2023	5,9915	0,5617	3,3404
20	1,0414	1,9031	2,1539	5,9915	1,0063	3,3404
21	0,3010	1,3010	1,4510	5,9915	0,6779	3,3404
22	1,0414	1,6532	2,6446	5,9915	1,2355	3,3404
23	0,7782	1,6532	1,1286	5,9915	0,5273	3,3404
24	0,0000	1,3979	5,3664	5,9915	2,5071	3,3404
25	0,9542	1,7993	1,7519	5,9915	0,8185	3,3404
26	0,4771	1,4472	1,0240	5,9915	0,4784	3,3404
27	0,0000	1,1139	3,4075	5,9915	1,5919	3,3404
28	0,7782	1,7160	1,1592	5,9915	0,5416	3,3404
29	1,0000	1,7076	2,1726	5,9915	1,0150	3,3404
30	1,1461	2,0828	2,6257	5,9915	1,2267	3,3404

Як бачимо, в наведеній таблиці викиди відсутні: $D^2 < \chi^2$ та $T_s < F$.

Розподіл вихідних даних для вибірки X представлений на рис. 2.1.а, а розподіл нормалізованих даних для вибірки X представлений на рис. 2.1.б.

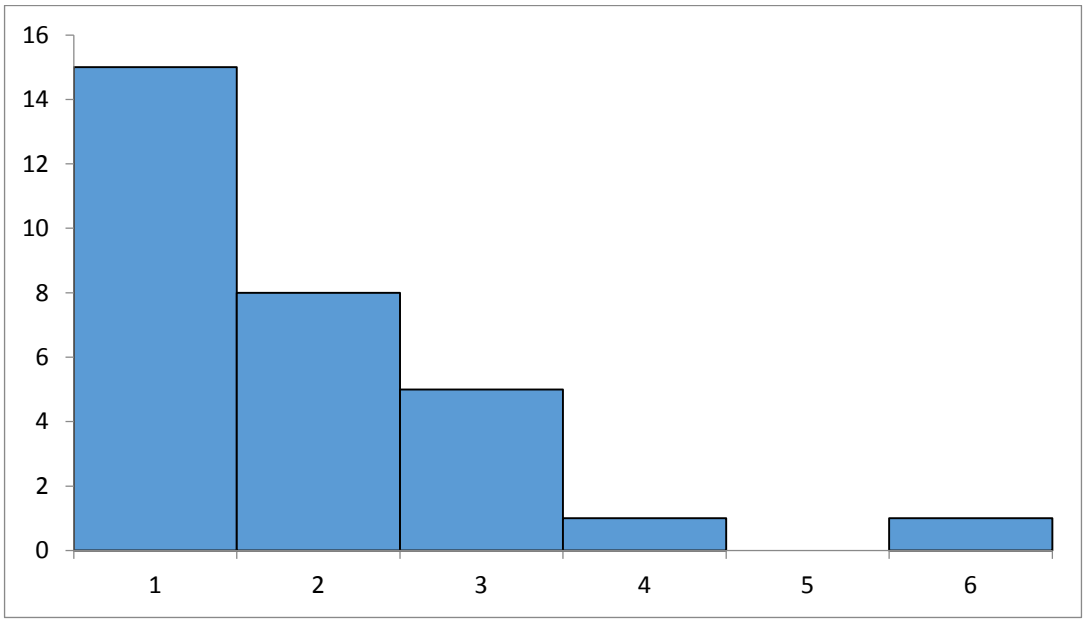

Рисунок 2.1.а – Розподіл вихідних даних для вибірки X

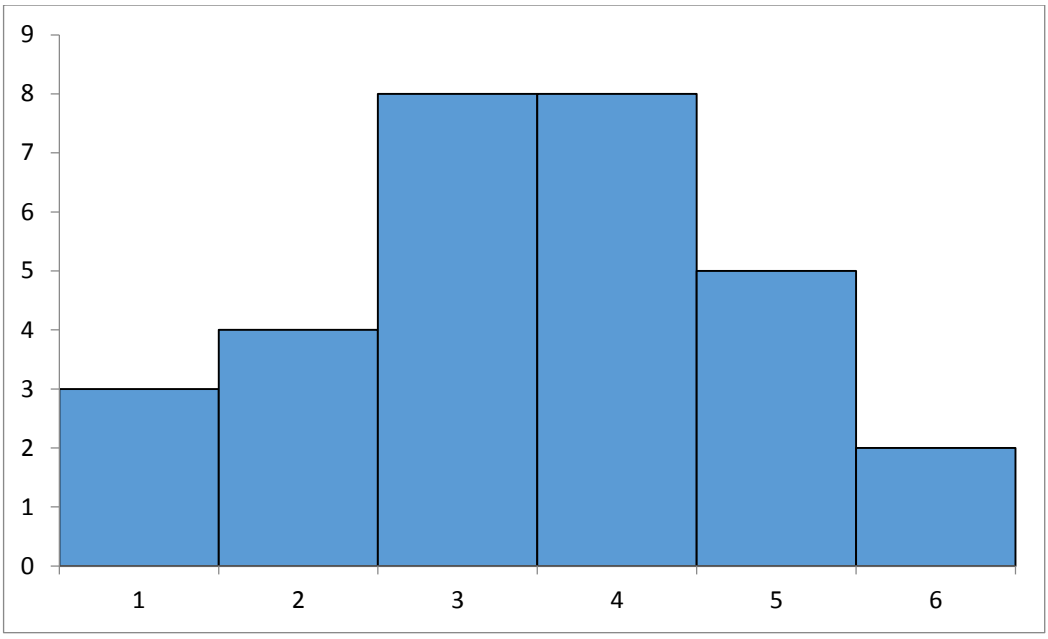

Рисунок 2.1.б – Розподіл нормалізованих даних для вибірки X

Розподіл вихідних даних для вибірки *Y* представлений на рис. 2.2.а, а розподіл нормалізованих даних для вибірки *2* представлений на рис. 2.2.б.

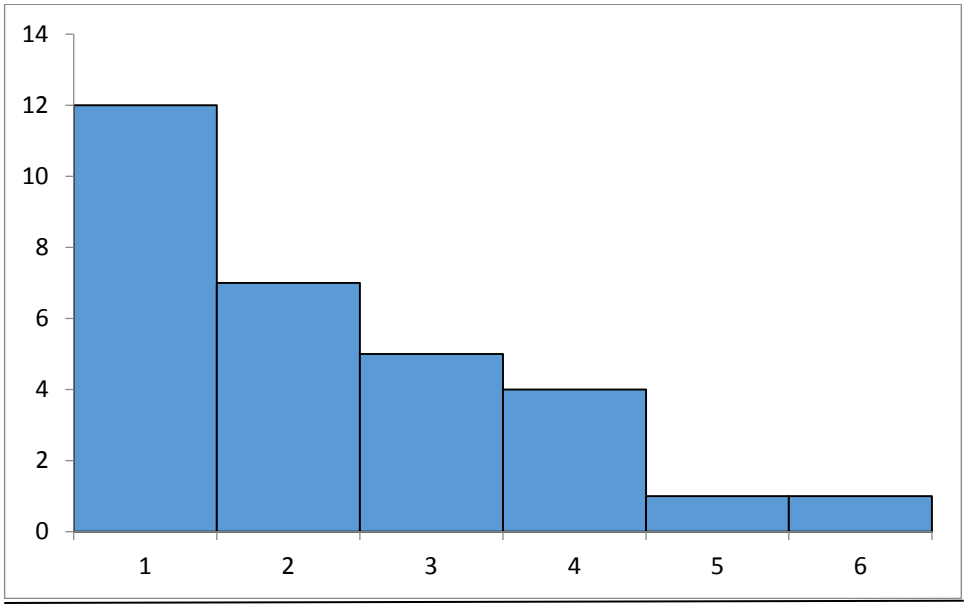

Рисунок 2.2.а – Розподіл вихідних даних для вибірки Y

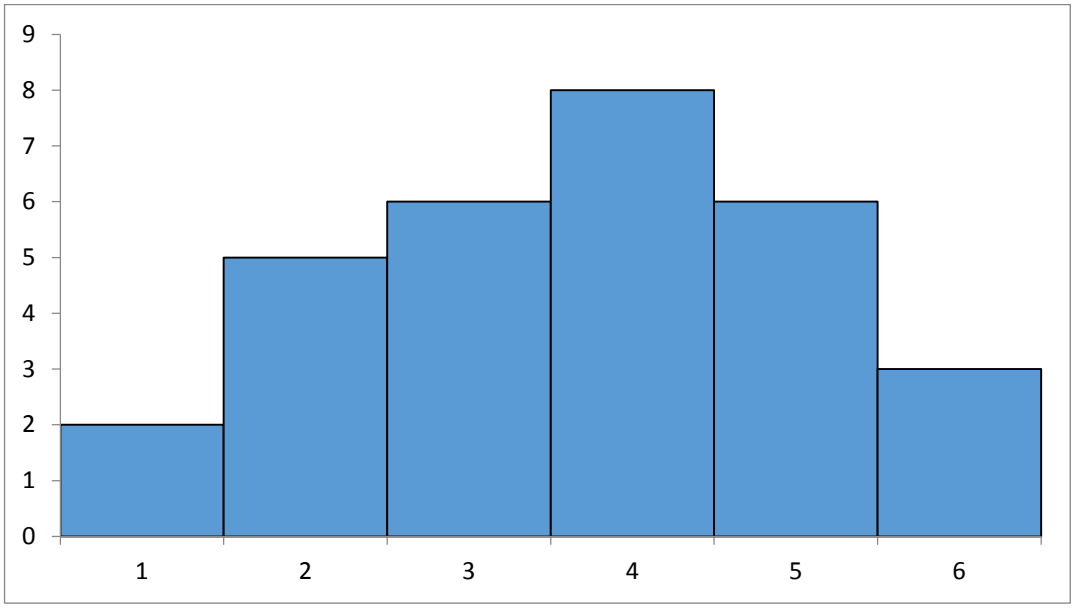

Рисунок 2.2.б – Розподіл нормалізованих даних для вибірки *Y*

Для нормілізованих X та Y використаємо лінійну регресійну теорію та визначимо коефіцієнти лінійного рівняння регресії та межі інтервалів прогнозування та предбачення.

Лінійне рівняння регресії з двома параметрами виглядає наступним чином [7, 8] (2.2):

$$Z_{\hat{Y}} = \mathrm{b}_0 + \mathrm{b}_1 \mathrm{Z}_\mathrm{X} \tag{2.2}$$

Значення коефіцієнтів були розраховані за методом найменших квадратів:

$$b_1 = 0{,}5702,\ b_0 = 1{,}2392.$$

Довірчий інтервал для отриманої регресії обчислюється згідно (2.3) [9, 10]:

$$\hat{Z}_y \pm t_{\alpha/2,n-2} \cdot S_{Z_y} \sqrt{\frac{1}{n} + \frac{(Z_x - \overline{Z_x})^2}{\sum_n^1 (z_{xi} - \overline{Z_x})^2}}, \tag{2.3}$$

де $\hat{Z}_y$ – значення, розраховані за рівнянням регресії;

$t_{\alpha/2,n-2}$ – квантіль t-розподілу Стьюдента;

α – рівень статистичної значущості;

n – кількість елементів в вибірці;

$$S_{Z_y} = \sqrt{\frac{1}{n-2} \cdot \sum_1^n \left(z_{yi} - \hat{z}_{y_i}\right)^2}.$$

Інтервал передбачення визначається згідно (2.4) [9, 10]:

$$\hat{Z}_y \pm t_{\alpha/2,n-2} \cdot S_{Z_y} \sqrt{1 + \frac{1}{n} + \frac{(Z_x - \overline{Z_x})^2}{\sum_n^1 (z_{x_i} - \overline{Z_x})^2}}. \tag{2.4}$$

де $S_{Z_y} = \sqrt{1 + \frac{1}{n-2} \sum_1^n (y_i - \hat{y}_i)^2}$;

$t_{\alpha/2,\vartheta}$ – квантіль t-розподілу Стьюдента.

де $S_{Z_y} = \sqrt{\frac{1}{n-2}\sum_1^n (y_i - \hat{y}_i)^2}$;

$t_{\alpha/2,\vartheta}$ – квантіль t-розподілу Стьюдента.

Нелінійна регресія будується на основі лінійного рівняння регресії з використанням зворотнього перетворення [11, 12]:

$$\hat{Y} = 10^{b_0} * X^{b_1}.$$

Тобто

$$\hat{Y} = 10^{1,2392} * X^{0,5702}$$

Також по формулам зворотнього перетворення знаходимо границі довірчого інтервалу та інтервалу передбачення для нелінійної регресії (переходимо до вихідних емпіричних даних).

На рисунку 2.3 наведені графіки прогнозного значення, довірчого інтервалу та інтервалу передбачення для нелінійної регресії.

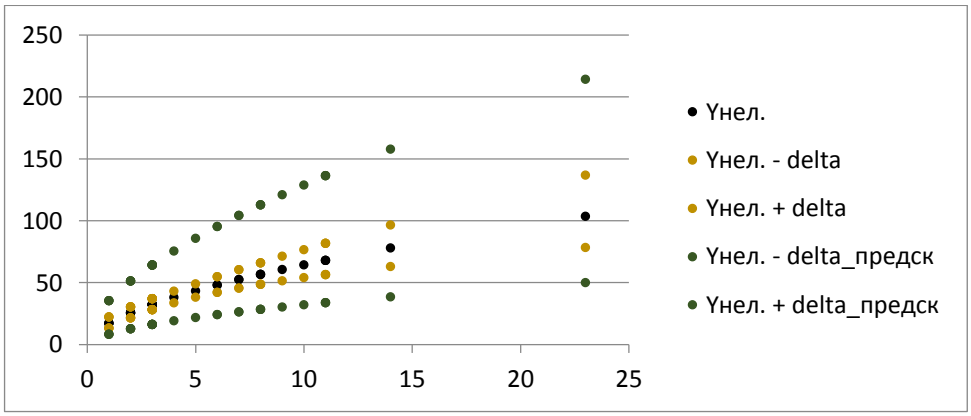

Рисунок 2.3 – Прогнозне значення, довірчий інтервал та інтервал передбачення для нелінійної регресії

Для порівняння було побудоване лінійне рівняння регресії для вихідних даних у припущенні про нормальність їх розподілу, та знайдені для нього межі довірчих інтервалів та інтервалів передбачення. Лінійне рівняння регресії має вигляд:

$$\hat{Y} = 4{,}4258X + 20{,}1784.$$

Для перевірки якості рівняння регресії існують наступні метрики [11, 13, 14]:

– коефіцієнт детермінації R^2 (розраховується за формулою 2.5):

$$R^2 = 1 - \left(\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 / \sum_{i=1}^{n} (y_i - \bar{y})^2 \right), \quad (2.5)$$

де y_i – емпіричне значення y;

$\hat{y}_i$ – розрахункове значення y;

$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$ – середнє значення випадкової величини y.

– сума квадратів відхилень (розраховується за формулою 2.6):

$$S_y = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \quad (2.6)$$

де y_i – фактичне значення випадкової величини y;

$\hat{y}_i$ – розрахункове значення.

– середня величина відносної похибки MMRE (розраховується за формулою 2.7):

$$MMRE = \frac{1}{n} \cdot \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (2.7)$$

де y_i – фактичне значення випадкової величини y;

$\hat{y}_i$ – розрахункове значення.

– рівень прогнозування $PRED(l)$ (розраховується за формулою 2.8):

$$PRED(l) = \frac{k}{n}, \quad (2.8)$$

де k – кількість значень $\left| \frac{y_i - \hat{y}_i}{y_i} \right| \leq l$ $(i = \overline{1,n})$.

В табл. 2.3 знаходяться результати виконаних розрахунків: прогнозовані значення, межі довірчих інтервалів та інтервалів передбачення для даних, нормалізованих за допомогою десяткового логарифму.

Таблиця 2.3 – Прогнозовані значення, межі довірчих інтервалів та інтервалів передбачення для даних, нормалізованих за допомогою десяткового логарифму.

№ проекту	Y прогн.	Y - delta	Y + delta	Y - delta передб.	Y + delta передб.
1	52,6108	45,7038	60,5616	26,5122	104,4008
2	25,7542	21,6900	30,5799	12,8874	51,4672
3	17,3460	13,4209	22,4191	8,4593	35,5686
4	25,7542	21,6900	30,5799	12,8874	51,4672
5	52,6108	45,7038	60,5616	26,5122	104,4008
6	56,7730	48,7802	66,0754	28,5427	112,9245
7	56,7730	48,7802	66,0754	28,5427	112,9245
8	103,6710	78,5083	136,8985	50,1579	214,2769
9	32,4530	28,3285	37,1780	16,3700	64,3370
10	25,7542	21,6900	30,5799	12,8874	51,4672
11	32,4530	28,3285	37,1780	16,3700	64,3370
12	68,0773	56,6142	81,8614	33,9555	136,4877
13	32,4530	28,3285	37,1780	16,3700	64,3370
14	32,4530	28,3285	37,1780	16,3700	64,3370
15	38,2380	33,7982	43,2609	19,3339	75,6258
16	56,7730	48,7802	66,0754	28,5427	112,9245
17	32,4530	28,3285	37,1780	16,3700	64,3370
18	32,4530	28,3285	37,1780	16,3700	64,3370
19	43,4263	38,3685	49,1508	21,9556	85,8935
20	68,0773	56,6142	81,8614	33,9555	136,4877
21	25,7542	21,6900	30,5799	12,8874	51,4672
22	68,0773	56,6142	81,8614	33,9555	136,4877
23	48,1839	42,2756	54,9180	24,3292	95,4279
24	17,3460	13,4209	22,4191	8,4593	35,5686
25	60,7168	51,5891	71,4596	30,4477	121,0775
26	32,4530	28,3285	37,1780	16,3700	64,3370
27	17,3460	13,4209	22,4191	8,4593	35,5686
28	48,1839	42,2756	54,9180	24,3292	95,4279
29	64,4763	54,1872	76,7191	32,2469	128,9174
30	78,1128	63,1219	96,6639	38,6447	157,8899

В табл. 2.4 знаходяться результати виконаних розрахунків: прогнозовані значення, межі довірчих інтервалів та інтервалів передбачення для емпіричних даних.

Таблиця 2.4 – Результати виконаних розрахунків: прогнозовані значення, межі довірчих інтервалів та інтервалів передбачення для емпіричних даних

№ проекту	Y прогноз.	Y - delta	Y + delta	Y - delta передб.	Y + delta передб.
1	51,1591	45,3007	57,0176	19,2180	83,1003
2	29,0301	21,5294	36,5307	-3,2883	61,3484
3	24,6043	16,2722	32,9363	-7,9383	57,1469
4	29,0301	21,5294	36,5307	-3,2883	61,3484
5	51,1591	45,3007	57,0176	19,2180	83,1003
6	55,5850	49,3633	61,8066	23,5680	87,6019
7	55,5850	49,3633	61,8066	23,5680	87,6019
8	121,9722	100,6289	143,3155	83,4281	160,5163
9	33,4559	26,6720	40,2398	1,3130	65,5988
10	29,0301	21,5294	36,5307	-3,2883	61,3484
11	33,4559	26,6720	40,2398	1,3130	65,5988
12	68,8624	60,5304	77,1944	36,3198	101,4050
13	33,4559	26,6720	40,2398	1,3130	65,5988
14	33,4559	26,6720	40,2398	1,3130	65,5988
15	37,8817	31,6600	44,1034	5,8648	69,8986
16	55,5850	49,3633	61,8066	23,5680	87,6019
17	33,4559	26,6720	40,2398	1,3130	65,5988
18	33,4559	26,6720	40,2398	1,3130	65,5988
19	42,3075	36,4491	48,1660	10,3664	74,2487
20	68,8624	60,5304	77,1944	36,3198	101,4050
21	29,0301	21,5294	36,5307	-3,2883	61,3484
22	68,8624	60,5304	77,1944	36,3198	101,4050
23	46,7333	41,0011	52,4656	14,8175	78,6492
24	24,6043	16,2722	32,9363	-7,9383	57,1469
25	60,0108	53,2269	66,7947	27,8679	92,1537
26	33,4559	26,6720	40,2398	1,3130	65,5988
27	24,6043	16,2722	32,9363	-7,9383	57,1469
28	46,7333	41,0011	52,4656	14,8175	78,6492
29	64,4366	56,9360	71,9372	32,1182	96,7550
30	82,1399	70,8943	93,3854	48,6429	115,6368

Довжини інтервалів без нормалізації більші, ніж довжини для нормалізованих даних. Результати передбачення рівняння лінійної регресії є негативними для кількох рядків. В той час, як усі результати прогнозування за нелінійним рівнянням регресії є позитивними.

Для нелінійної та лінійної регресій були пораховані R^2, MMRE та PRED(25), які представлені в таблиці 2.5.

Таблиця 2.5 – R^2, MMRE та PRED(25) для нелінійної та лінійної моделей

№ п/п	Назва моделі	Коефіцієнт детермінації R^2	Середня величина відносної похибки MMRE	Рівень прогнозування PRED(25)
1	Нелінійна (log)	0,6923	0,2478	0,5333
2	Лінійна	0,6703	0,2736	0,5000

Звідси бачимо, що в випадку нормалізованих даних всі показники кращі, ніж в вихідних даних.

2.3 Постановка задачі на розробку ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin

Виходячи з аналізу існуючих методів і моделей для оцінювання розміру програмного забезпечення та побудувавши удосконалене нелінійне регресійне рівняння для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin, сформулюємо наступну постановку задачі на розробку програми.

В даній роботі необхідно розробити програмне забезпечення, основними функціями якого будуть:

1) надавати можливість вносити дані про програмні проекти;

2) виконувати нормалізацію внесених даних;

3) будувати нелінійне рівняння регресії;

3) будувати довірчі інтервали на основі нормалізованих даних;

4) будувати інтервали передбачення на основі нормалізованих даних;

6) розраховувати метрики якості.

7) оцінювати розмір програмних проектів.

Детальні вимоги до програми наведені у додатку А.

3 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПЗ З ВІДКРИТИМ КОДОМ НА KOTLIN

При розробці проекту ПЗ був використаний об’єктно-орієнтований підхід та уніфікована мова моделювання (UML).

На основі розробленого технічного завдання була виконана поетапна розробка проекту програмного забезпечення: ескізне проектування, технічне, розробка робочого проекту [15].

3.1 Ескізній проект програмного забезпечення для оцінювання розміру програмного забезпечення

В рамках ескізного проектування були розроблені: діаграма варіантів використання, специфікації варіантів використання, діаграми діяльності та проект користувальницького інтерфейсу.

3.1.1 Розробка діаграми варіантів використання програмного забезпечення

Основні вимоги до програмного продукту, що відображені у технічному завданні (додаток А), можна відобразити на діаграмі варіантів використання.

Даний проект передбачає наявність лише одного актора – користувача програми, який буде повноцінно використовувати всі функції програмного продукту. Виявлені актори програмного забезпечення представлені в таблиці 3.1.

Таблиця 3.1 – Виявлені актори програмного забезпечення

Актор	Короткий опис
Користувач	Особа, яка працює з програмним забезпеченням та має можливість ввести вихідні дані, виконати нормалізацію за допомогою десяткового логарифму, обчислити довірчий інтервал та інтервал передбачення, визначити показники якості, оцінити розмір

В таблиці 3.2 представлений короткий опис варіантів використання програмного забезпечення.

Таблиця 3.2 – Короткий опис варіантів використання програмного забезпечення.

Основні актори	Найменування	Формулювання
Користувач	Побудувати рівняння регресії	Будує рівняння регресії для оцінювання розміру програмного забезпечення
Користувач	Отримати довірчий інтервал	Визначає довірчий інтервал
Користувач	Отримати інтервал передбачення	Визначає інтервал передбачення
Користувач	Визначити метрики якості	Визначає метрики якості
Користувач	Внести вихідні дані	Вносить вихідну інформацію
Користувач	Нормалізувати дані	Виконує нормалізацію внесених даних
Користувач	Виконати оцінювання розміру та побудувати інтервал передбачення цієї оцінки	Виконує оцінювання розміру на основі нелінійної регресії та будує її інтервал передбачення

Діаграма варіантів використання програмного забезпечення зображена на рисунку 3.1.

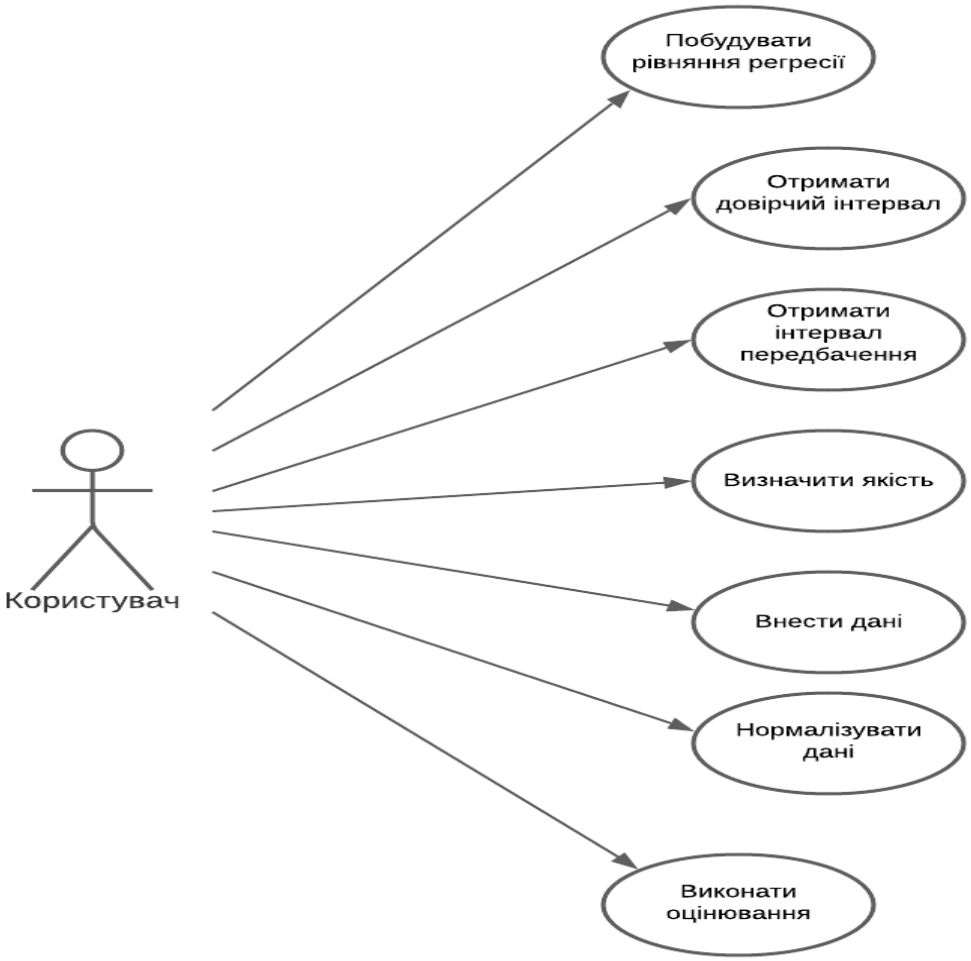

Рисунок 3.1 – Діаграма варіантів використання програмного забезпечення

Використовуючи діаграму варіантів використання (рисунок 3.1), формалізуємо функціональні вимоги до ПЗ, які наведені нижче у специфікаціях варіантів використання.

3.1.2 Специфікації варіантів використання ПЗ для оцінювання розміру ПЗ

В таблиці 3.3 представлені специфікації віріантів використання ПЗ

Таблиця 3.3 – Специфікації варіантів використання ПЗ

Побудувати рівняння регресії	
1	2
Короткий опис	Система підраховує необхідні коефіцієнти для функції
Дійові особи прецеденту	Користувач
Передумови	Користувач має попередньо ввести дані
Базовий потік	Система здійснює підбір коефіцієнтів для шуканої функції
Альтернативний потік	Відсутній
Постумови	Відсутні
Отримати довірчий інтервал	
Короткий опис	Система підраховує довірчі інтервали для даних, нормалізованих методом десяткового логарифму, та відображає їх на екрані
Дійові особи прецеденту	Користувач
Передумови	Користувач має попередньо ввести дані
Базовий потік	1. Система здійснює підрахунок довірчих інтервалів 2. Користувач натискає кнопку «Показати інтервали» 3. Довірчі інтервали для даних, нормалізованих методом десяткового логарифму, і даних без нормалізації відображаються на екрані

Продовження табл. 3.3

1	2
Альтернативний потік	Відсутній
Постумови	Відсутні
Визначити якість	
Короткий опис	Система підраховує якість, а саме R^2, MMRE та PRED(25) для кожного регресійного рівняння та відображає їх на екрані
Дійові особи прецеденту	Користувач
Передумови	Користувач має попередньо ввести дані
Основний потік	1. Система здійснює підрахунок метрик 2. Система відображає значення метрик
Альтернативний потік	Відсутній
Після умови	Відсутні
Отримати інтервал предбачення	
Короткий опис	Система підраховує інтервали передбачення для даних, нормалізованих методом десяткового логарифму, та відображає їх на екрані
Дійові особи прецеденту	Користувач
Передумови	Користувач має попередньо ввести дані
Базовий потік	1. Система підраховує інтервали передбачення 2. Користувач натискає кнопку «Показати інтервали» 3. Інтервали передбачення для даних, нормалізованих методом десяткового логарифму відображаються на екрані
Альтернативний потік	Відсутній
Постумови	Відсутні

Продовження табл. 3.3

1	2
Виконати оцінювання	
Короткий опис	Система виконує оцінювання розміру за введеною кількістю функцій
Дійові особи прецеденту	Користувач
Передумови	Внесені емпіричні дані
Базовий потік	1.Користувач вводить дані для оцінювання. 2. Система зчитує дані та розраховує прогноз і інтервали передбачення. 3. Система виводить на екран результати оцінювання
Альтернативний потік	Відсутній
Постумови	Відсутні
Внести дані	
Опис прецеденту	Користувач запускає програму та натискає кнопку для вибору шляху до файлу з даними
Дійові особи прецеденту	Користувач
Передумови	Немає
Базовий потік	1. Користувач натискає кнопку «Вибрати файл» 2. Користувач в файловій системі знаходить необхідний файл та обирає його 3. Система зчитує дані з файлу та аналізує їх
Альтернативний потік	1. Користувач натискає кнопку «Вибрати файл» 2. Користувач в файловій системі знаходить необхідний файл та обирає його 3. Система зчитує дані з файлу, але через неправильний формат даних відображає повідомлення про те, що формат даних є невірним .

Продовження табл. 3.3

1	2
Постумови	Якщо користувач ввів шлях до файлу з даними у відповідному форматі, то дані зчитуються з файлу і з ними виконуються подальші підрахунки
Нормалізувати дані	
Опис прецеденту	Система виконує нормалізацію даних за допомогою десяткового логарифму
Дійові особи прецеденту	Користувач
Передумови	Користувач має попередньо ввести дані
Базовий потік	1. Система отримує дані з файлу 2. Система здійснює їх нормалізацію за допомогою десяткового логарифму
Альтернативний потік	Відсутні
Постумови	Відсутні

3.1.3 Діаграми діяльності для основних варіантів використання

Для більш зрозумілого сприйняття діаграми варіантів використання, наведеної на рисунку 3.2, та специфікацій варіантів використання, необхідно привести діаграми діяльності.

Діаграма діяльності для варіанту використання «Визначити якість» зображена на рисунку 3.3.

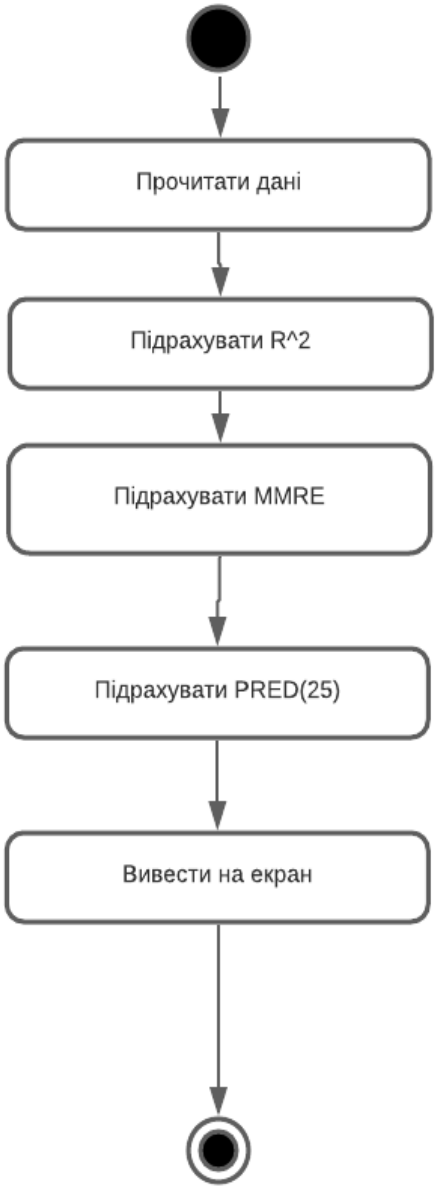

Рисунок 3.2 – Діаграма діяльності ПЗ для оцінювання розміру

Діаграма діяльності для варіанту використання «Отримати інтервал передбачення» представлена на рисунку 3.3.

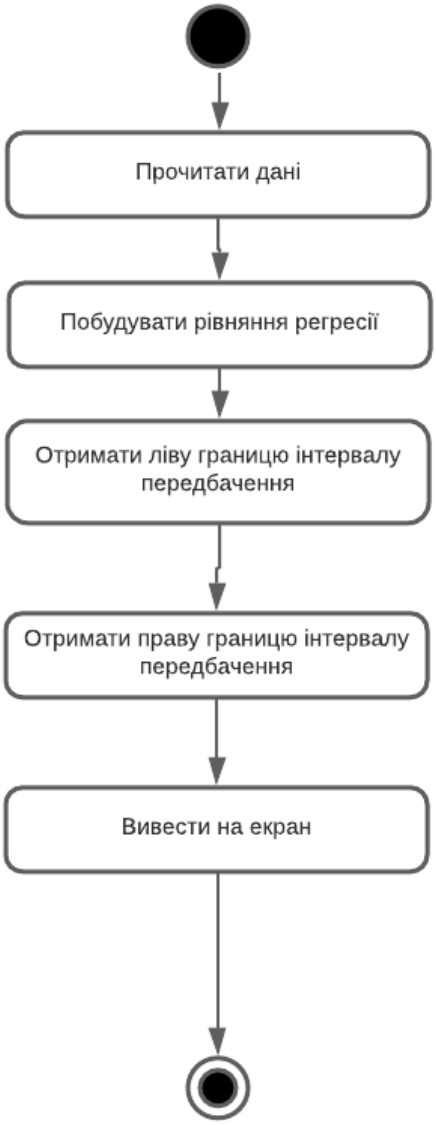

Рисунок 3.3 – Діаграма діяльності для варіанту використання «Отримати інтервал передбачення»

3.1.4 Інформаційна модель даних ПЗ для оцінювання розміру ПЗ

На підставі технічного завдання з’являється необхідність створення бази даних. Визначимо логічні зв’язки між сутностями та розробимо схему даних у вигляді концептуальної моделі даних [15].

При побудові концептуальної моделі необхідно визначити: сутності, атрибути сутностей та зв’язки між сутностями.

У предметній галузі виявлені наступні сутності: емпіричні дані, нормалізовані дані, коефіцієнти рівняння та інтервали. В базі даних є зв’язки між сутностями один до одного та один до багатьох. Атрибути сутностей представлені на концептуальній моделі.

Концептуальна модель даних програми для оцінювання розміру програмного забезпечення представлена у вигляді ER-діаграми на рисунку 3.4.

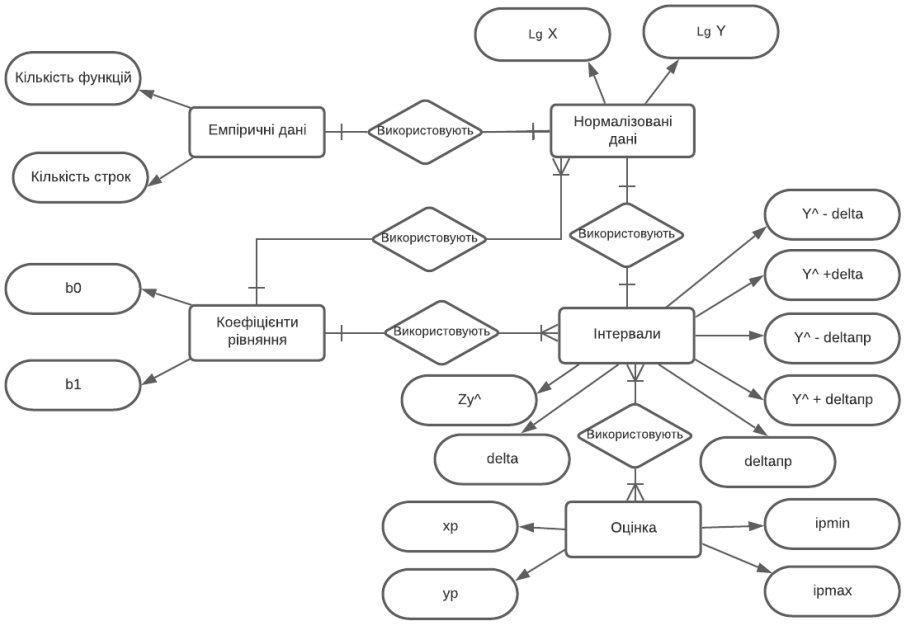

Рисунок 3.4 – ER-діаграма програмного забезпечення для оцінювання розміру програми

3.1.5 Проектування інтерфейсу ПЗ для оцінювання розміру програмного забезпечення

Виконаємо проектування користувальницького інтерфейсу розроблювального ПЗ. На основі описаних вище варіантів використання визначимо відповідні головні інтерфейсні компоненти.

Спочатку виконаємо проектування інтерфейсу головного вікна ПЗ (рис.3.5).

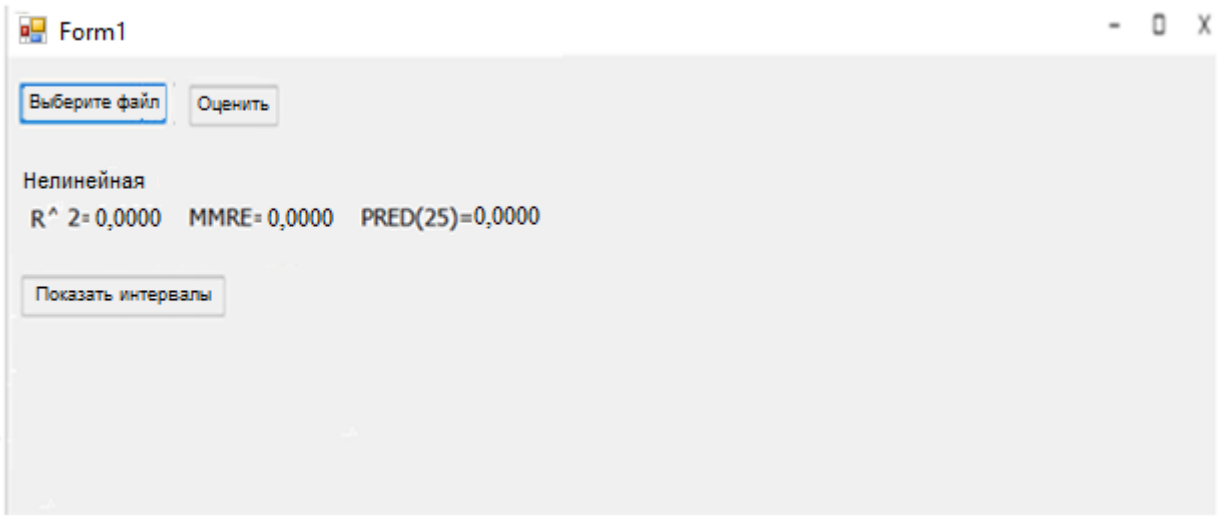

Рисунок 3.5 – Проектування інтерфейсу головного вікна ПЗ

Як бачимо з рисунку 3.4, програма повинна реалізовувати введення емпіричних даних, розрахунок метрик якості регресійного рівняння, розрахунок довірчих інтервалів та інтервалів передбачення, оцінювання розміру нового компоненту.

3.2 Технічний проект ПЗ для оцінювання розміру програмного забезпечення

На основі результатів ескізного проектування ПЗ оцінювання розміру програмного забезпечення переходимо до технічного проектування [15].

В рамках технічного проекту розробляються статична та динамічна модель ПЗ.

3.2.1 Розробка статичної моделі ПЗ для оцінювання розміру програмного забезпечення

Діаграму моделі аналізу, яка була розроблена у ескізному проекті, необхідно довести до технічного рівня. Для цього розробимо статичну модель ПЗ (діаграму класів).

На рисунку 3.6 представлена діаграма класів ПЗ для оцінювання розміру програмного забезпечення.

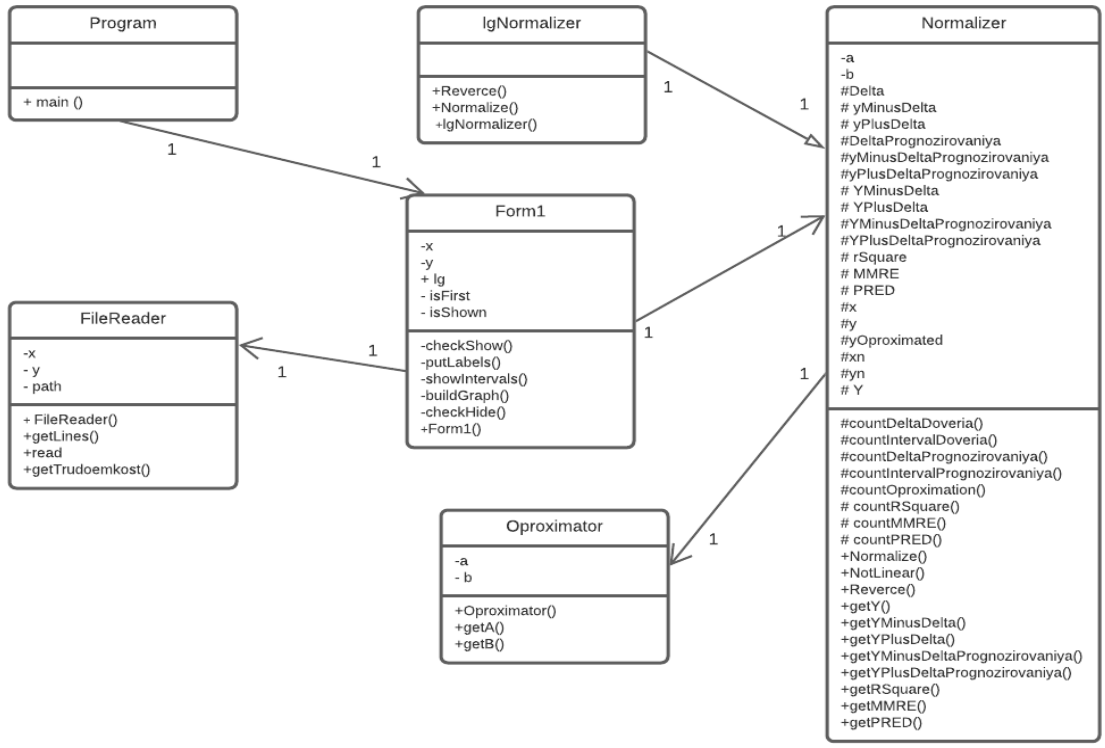

Рисунок 3.6 – Діаграма класів ПЗ для оцінювання розміру програми

Для більш детального сприйняття ієрархії класів приведемо специфікацію головних класів програмного забезпечення.

Клас Form1

Клас призначений для створення графічного інтерфейсу програми.

Включає в себе наступні атрибути:

x – масив значень функцій

y – масив значень розміру

isShown – зміна яка показує чи потрібно відображувати таблицю з значеннями інтервалів

lg – зміна призначена для зберігання екземпляру класа lnNormalizer

Включає в себе наступні методи:

Form1() – конструктор класу

buildGraph() – метод для побудови і відображення графіків інтервалів

putLabels() – метод для відображення метрик якості

showIntervals() – метод призначений для відображення порахованих інтервалів

checkShow() – метод для перевірки того, що в даний момент інтервали відображаються на формі

checkHide() – метод для перевірки того, що в даний момент інтервали не відображаються на формі

Клас fileReader

Клас призначений для зчитування даних з файлу.

Включає в себе наступні атрибути:

x – призначений для зберігання id даної програми

y – призначений для зберігання ім’я даної програми

path – призначений для зберігання шляху до файлу з даними

Включає в себе наступні методи:

fileReader() – конструктор класу

getLines() – метод для отримання масиву зі значеннями строк коду

get Razmer() – метод для отримання масиву зі значенями розміру

read() – метод для зчитування даних з файлу

Клас Program

Клас призначений для запуску програми

Включає в себе наступні методи:

Main() – метод для запуску програми

Клас lgNormalizer

Клас призначений для нормалізації даних за допомогою десяткового логарифму і для подальшого зворотнього перетворення.

Включає в себе наступні методи:

lgNormalizer() – конструктор класу

Normalize() – метод для нормалізації даних за допомогою десяткового логарифму

Reverse() – метод для зворотнього перетворення

Клас Program

Клас призначений для запуску програми

Включає в себе наступні методи:

Main() – метод для запуску програми

Клас Oproximator

Клас призначений для апроксимації функції.

Включає в себе наступні атрибути:

a – значення коефіцієнту a

b – значення коефіцієнту b

Включає в себе наступні методи:

Oproximator() – конструктор класу

getA() – метод призначений для отримання коефіцієнту a

getB() – метод призначений для отримання коефіцієнту b

Клас Normalizer

Клас відповідаючий за основні розрахунки

Включає в себе наступні атрибути:

x – масив значень строк коду

y – масив значень розміру

yOproximated – масив апроксимованих y

xn – нормалізовані x

yn – нормалізовані y

delta – Δ для довірчого інтервала

yMinusDelta – нижня границя довірчого інтервалу

yPlusDelta – верхня границя довірчого інтервалу

deltaPrognozirovaniya – Δ інтервалу передбачення

yMinusDeltaPrognozirovaniya – нижня границя інтервалу передбачення

yPlusDeltaPrognozirovaniya – верхня границя інтервалу передбачення

Y – y після зворотнього перетворення

YMinusDelta – нижня границя довірчого інтервалу після зворотнього перетворення

YPlusDelta – верхня границя довірчого інтервалу після зворотнього перетворення

YMinusDeltaPrognozirovaniya – нижня границя інтервалу передбачення після зворотнього перетворення

YPlusDeltaPrognozirovaniya – верхня границя інтервалу передбачення після зворотнього перетворення

a – значення коефіцієнту a

b – значення коефіцієнту b

rSquare – значення R^2

MMRE – значення MMRE

PRED – значення PRED(25)

Включає в себе наступні методи:

Normalizer() – конструктор класу

countOproximation() – метод для підрахунку апроксимованих значень y

countDeltaDoveria() – метод для підрахунку Δ довірчого інтервалу

countIntervalDoveria() – метод для підрахунку довірчого інтервалу

countDeltaPrognozirovaniya() – метод для підрахунку Δ інтервалу передбачення

countIntervalPrognozirovaniya() – метод для підрахунку інтервалу передбачення

countRSquare() – метод для підрахунку R^2

countMMRE() – метод для підрахунку MMRE

countPRED() – метод для підрахунку PRED(25)

getY() – метод для отримання значень y після зворотнього перетворення

getYMinusDelta() – метод для отримання нижньої границі довірчого інтервалу після зворотнього перетворення

getYPlusDelta() – метод для отримання верхньої границі довірчого інтервалу після зворотнього перетворення

getYMinusDeltaPrognozirovaniya() – метод для отримання нижньої границі для інтервалу передбачення після зворотнього перетворення

getYPlusDeltaPrognozirovaniya() – метод для отримання верхньої границі для інтервалу передбачення після зворотнього перетворення

getRSquare() – метод для отримання R^2

getMMRE() – метод для отримання MMRE

3.2.2 Динамічна модель програмного забезпечення

Для моделювання взаємодії системи використовуються діаграми послідовності і комунікацій.

Діаграма послідовності – діаграма, яка служить для подання взаємодії елементів моделі у формі послідовності повідомлень і відповідних подій на лініях життя об'єктів.

Діаграма комунікацій – діаграма, яка призначена для представлення взаємодії в контексті внутрішньої архітектури системи і переданих повідомлень.

Діаграма послідовності для процесу обробки емпіричного набору даних (нормалізації, побудови лінійної моделі, інтервалів прогнозування та довіри та обчислення метрик якості) представлена на рисунку 3.7.

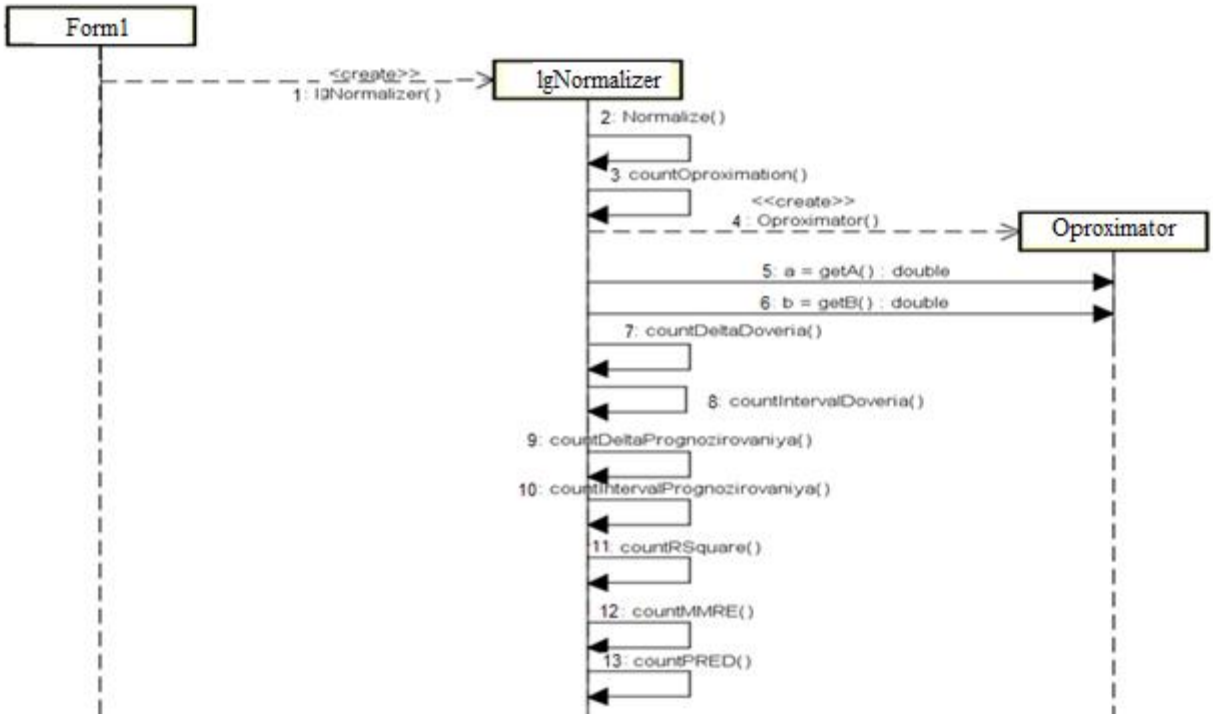

Рисунок 3.7 –Діаграма послідовності для процесу обробки емпіричного набору даних

Діаграма послідовності варіанту використання «Ввести дані» представлена на рисунку 3.8.

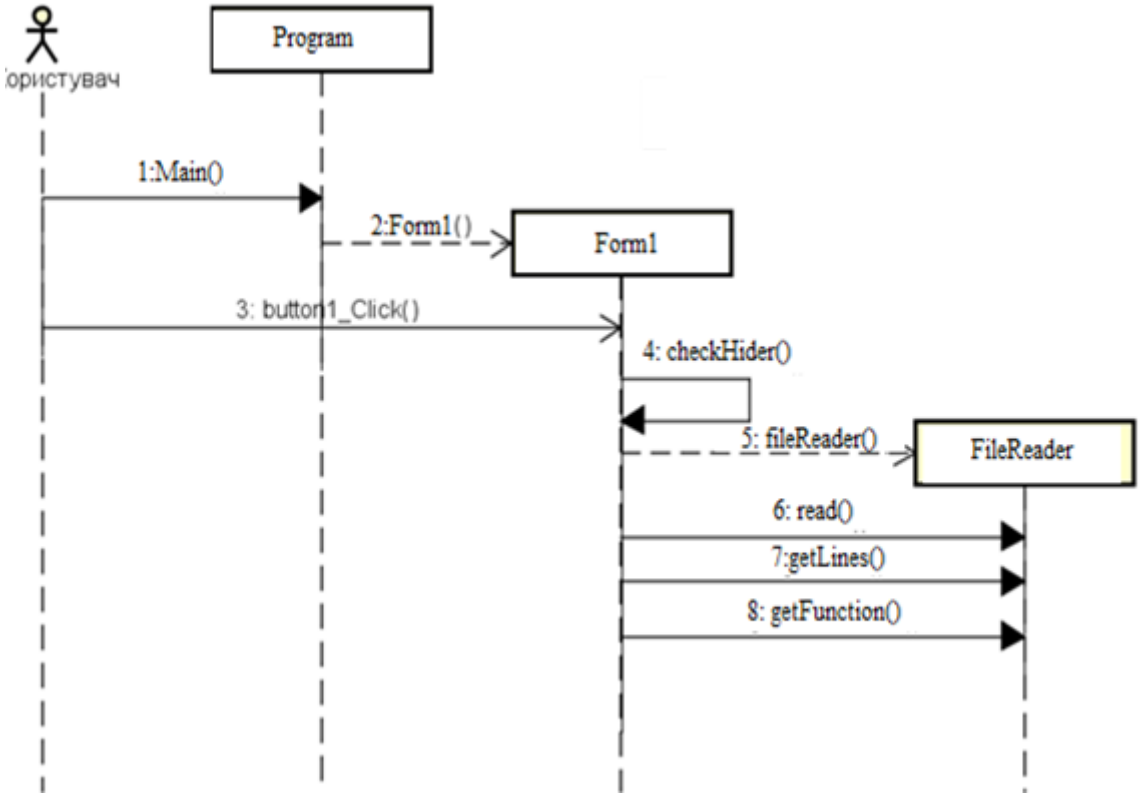

Рисунок 3.8 – Діаграма послідовності варіанту використання «Ввести дані»

3.3 Робочий проект ПЗ для оцінювання розміру програмного забезпечення

На основі результатів розробки технічного проекту ПЗ перейдемо до розробки робочого проекту ПЗ. В робочому проекті виконується вибір мови програмування, розробка діаграми компонентів та фізичної моделі даних ПЗ, тестування та випробування.

3.3.1 Вибір мови програмування

В якості мови для реалізації проекту була обрана мова C#.

Розвиток комп'ютерних технологій приводить до появи нових мов програмування. Мова C# з’явилася в 2001 році та була розроблена фірмою Microsoft. Розглянемо основні переваги C#:

– компонентно-орієнтована;

– враховує всі можливості каркасу Framework.Net;

– об'єктно-орієнтована;

– наслідує мови C/C++;

– більш проста та надійна, ніж C/C++;

– має такі ж переваги роботи з віртуальною машиною, що і Java;

– має досить обширну бібліотеку. Ця бібліотека дозволяє легко будувати веб-додатки, зручно працювати з базою даних.

3.3.2 Розробка діаграми компонентів ПЗ для оцінювання розміру програмного забезпечення

Діаграма компонентів забезпечує узгоджений перехід від логічного уявлення до конкретної реалізації проекту у формі програмного коду [16]

На рисунку 3.9 представлено діаграму компонентів програмного забезпечення для оцінювання розміру.

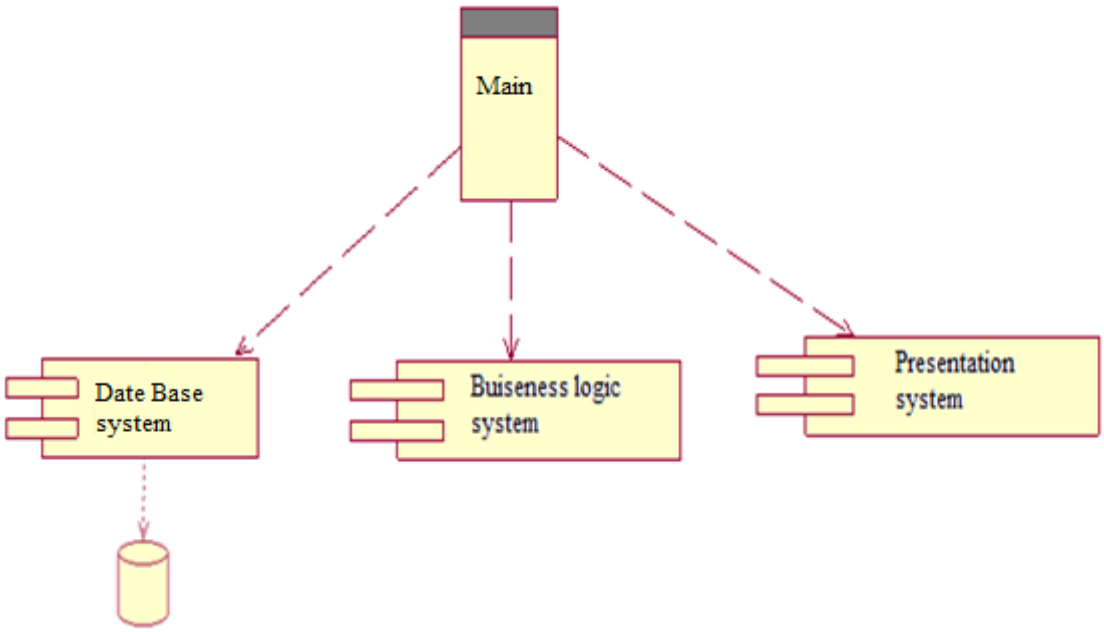

Рисунок 3.9 – Діаграма компонентів

3.3.3 Реалізація та тестування програмного забезпечення

В якості мови програмування була обрана мова C#. Текст програмного забезпечення знаходиться в додатку Г.

Тестування

В ході виконання тестування програмного продукту були протестовані всі компоненти. Нижче наведено тестування функцій введення даних та оцінювання методом «чорної скриньки».

В таблиці 3.4 представлені класи еквівалентності вхідних даних для функції «Введення даних для оцінювання».

Таблиця 3.4 – Класи еквівалентності вхідних даних

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Кількість функцій	1 Ціле число>0	2 Пустий рядок 3 Не число 4 Число<0

В таблиці 3.5 представлені тести з правильних класів еквівалентності.

Таблиця 3.5 – Тести з правильних класів еквівалентності

№ тесту	№ покритого класу	Вхідні умови	Результат
1	1	3	Інформація про новий проект була записана до бази даних

Отже, за результатами правильних класів еквівалентності програма працює правильно і без збоїв.

В таблиці 3.6 представлені тести з неправильних класів еквівалентності.

Таблиця 3.6 – Тести з неправильних класів еквівалентності

№ тесту	№ покритого класу	Вхідні умови	Результат
1	2		Повідомлення про помилку
2	3	вв	Повідомлення про помилку
3	4	-3	Повідомлення про помилку

Отже, за результатами тестування неправильних класів еквівалентності – програма працює правильно.

Розглянемо тестування функції «Введення даних з файлу».

В таблиці 3.7 представлені класи еквівалентності вхідних даних для функції «Введення даних з файлу».

Таблиця 3.7 – Класи еквівалентності вхідних даних

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Дані з файлу	1 Дані правильного формату	2 Файл пустий 3 Дані неправильного формату

В таблиці 3.8 представлені тести з правильних класів еквівалентності.

Таблиця 3.8 – Тести з правильних класів еквівалентності

№ тесту	№ покритого класу	Вхідні умови	Результат
1	1	3 5	Інформація про новий проект була записана до бази даних

Отже, за результатами правильних класів еквівалентності програма працює правильно і без збоїв.

В таблиці 3.9 представлені тести з неправильних класів еквівалентності.

Таблиця 3.9 – Тести з неправильних класів еквівалентності

№ тесту	№ покритого класу	Вхідні умови	Результат
1	2		Повідомлення про помилку
2	3	Вв аа	Повідомлення про помилку

Отже, за результатами тестування неправильних класів еквівалентності – програма працює правильно.

3.3.4 Випробування програмного забезпечення

Випробування зводилися до послідовного вводу тестових наборів даних й аналізу отриманих результатів. Випробування програмного забезпечення було проведено відповідно до програми й методики випробувань (Додаток Б), що містить уточнення вимог технічного завдання для даного ПЗ і гарантує їхню коректну перевірку.

Випробування можливості введення даних з файлу

- Натиснули кнопку «Вибрати файл».
- Відкрилося вікно, в якому обрали шлях до файлу.
- Файл відкрився і поблизу кнопки з’явилася назва файлу

Після натискання кнопки «Вибрати файл» файл додався. Результат відповідає очікуваному.

Перевірка можливості продивитися довірчі інтервали та інтервали передбачення.

- Ввели дані.
- Натиснули кнопку «Показати інтервали».
- Інтервали відобразилися в таблиці.
- Кнопка «Показати інтервали» змінила текст на «Сховати інтервали».

Після натискання кнопки «Показати інтервали» на головній формі додалася таблиця, яка містить ці інтервали.

Перевірка можливості сховати інтервали

– Вели дані та натиснув кнопку «Показати інтервали».
– Натиснули кнопку «Сховати інтервали».
– Таблиця з інтервалами зникла із основної форми.
– Кнопка «Сховати інтервали» змінила текст на «Показати інтервали».

Дані випробування показали, що розроблене програмне забезпечення функціонує нормально.

4 РЕЗУЛЬТАТИ РОЗРОБКИ ПЗ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Під час виконання кваліфікаційної роботи були поставлені завдання, які було виконано у повному обсязі:

– виконаний аналіз існуючих методів та моделей оцінювання розміру ПЗ;

– обґрунтована необхідність удосконалення рівняння регресії для оцінювання розміру ПЗ з відкритим кодом на Kotlin;

– досліджені різні джерела з відкритим вихідним кодом та зібрані дані для удосконалення рівняння регресії;

– нормалізовані отримані емпіричні дані, використовуючи перетворення у вигляді десяткового логарифму;

– перевірені дані на викиди;

– побудоване лінійне рівняння регресії, довірчий інтервал та інтервал передбачення для вихідних даних;

– побудуване нелінійне рівняння регресії, довірчий інтервал та інтервал передбачення на основі нормалізованих даних;

– розроблене ПЗ для оцінювання розміру програмного забезпечення.

На основі десятичного логарифму була підвищена достовірність оцінювання розміру програмного забезпечення.

Технічне завдання на розробку програмного забезпечення представлене в додатку А, Програма та методика випробувань – в додатку Б, Опис програмного забезпечення – в додатку В, Текст програмного забезпечення – в додатку Г, Інструкція користувача – в додатку Д.

Розмір файлу з програмою становить 340 КБ.

ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin призначене для використання на персональних комп'ютерах, що працюють під управлінням наступних операційних систем: ОС MS Windows XP/7/8/8.1/10.

Програмне забезпечення успішно пройшло тестування та випробування, після яких були внесені необхідні корективи до програмного забезпечення.

5 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Вступ

Дана робота присвячена розробці програмного забезпечення для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin Програмне забезпечення можна використати в галузі управління проектами розробки програмного забезпечення.

Створення програмного забезпечення вимагає одноразових витрат на його розробку, придбання необхідних технічних засобів і поточних витрат на функціонування. Економія від функціонування програмного забезпечення визначається з врахуванням витрат на його експлуатацію. Відношення цієї економії до витрат на створення програмного забезпечення характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються по діючим на момент розрахунку оптовим цінам, тарифам і ставкам заробітної плати.

5.2 Розрахунок витрат на створення й експлуатацію програмного забезпечення

Витрати на розробку ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin складаються з витрат: на оплату праці розробника, на амортизацію комп‘ютера, на якому відбувається розробка, на експлуатацію цього комп‘ютера, на засоби розробки та витрат на виробничі запаси.

Розробка виконується програмістом із місячним окладом 10 000 грн. Додаткова заробітна плата складає 20% від основної. Таким чином, основна і додаткова заробітна плата програміста складає 12 000 грн. Вартість ноутбука складає 8 000 грн. При вартості кіловат-години електроенергії у 1,7166 грн.

(II клас напруги до 27,5 кВт, разом з ПДВ), розраховується вартість ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin. Витрати на допоміжні матеріали наведені в табл. 5.1., на розробку – в табл. 5.2.

Таблиця 5.1 – Витрати на допоміжні матеріали

Вид допоміжних матеріалів	Вартість
Офісний папір	100,00
Заправлення картриджа до принтеру	100,00
Флеш-накопичувач даних	250,00
Разом допоміжні матеріали	450,00

Вартість програмного забезпечення визначимо за формулою:

$$Впз=(Вопт+Всз+ЗВВ+Ве)*Т+Вдм, \quad (5.1)$$

де Впз – вартість програмного забезпечення, грн; Вопт – витрати на оплату праці (основна та додаткова), грн; Всз – відрахування на соціальні заходи або єдиний соціальний внесок (36,76% від основної і додаткової заробітної плати), грн.; ЗВВ – загально-виробничі витрати (10% від основної заробітної плати), грн.; Зе – витрати на електроенергію, грн.; Т – тривалість розробки, міс.; Вдм – витрати на допоміжні матеріали, грн.

Витрати на електроенергію за умови споживання 0,5 кВт, тривалості роботи за місяць, дорівнює 22*8=176 годин, що у підсумку складає:

$$Ве=176*1,7166*0,5=151,06 \text{ грн.}$$

Таблиця 5.2 – Витрати на розробку програмного забезпечення

Стаття витрат	Сума витрат, грн.
Витрати на оплату праці	12000,00
Відрахування на соціальні заходи	4411,20
Загальновиробничі витрати	1000,00
Витрати на допоміжні матеріали	450,00
Витрати на електроенергію	151,06

Відповідно до формули (5.1) вартість програмного забезпечення складає:

Впз=(12000+4411,2+1000+151,06)*2+450=17562,26*2+450=35 574,52 грн

Амортизаційні відрахування на основні засоби (ноутбук) складають 25% від первісної вартості в рік:

Вамор=8000*0,25=2 000,00 грн

У масштабах підприємства річні витрати на матеріали визначаються у розмірі 5% вартості основних засобів і складають:

Вм=8000*0,05=400,00 грн.

Річне споживання електроенергії ноутбуком у годинах визначається в такій спосіб:

$$Чн=264,5*Тз, \quad (5.2)$$

де Тз – середнє місячне завантаження устаткування (близько 4 годин); 264,5 – середня кількість робочих днів на рік.

Отже річне споживання електроенергії ноутбуком складає:

Чн=264,5*4=1058 годин.

Витрати на споживання електроенергії при 1058 годин роботи ноутбука за рік становить:

Ве=1058*1,7166*0,5=908,08 грн.

Експлуатаційні витрати для ноутбука за рік складають:

Взр=2000+400+908,08=3 308,08 грн.

Отже за перший рік витрати на створення й експлуатацію програми складають:

Все=35 574,52+3 308,08=35 574,52грн.

5.3 Економічна ефективність розробки і впровадження

Основним показником економічної ефективності функціонування програмного забезпечення є зниження трудомісткості розрахункових робіт щодо оцінювання розміру програмного забезпечення з відкритим кодом на

Kotlin. До числа основних факторів, що визначають приріст прибутку в зв‘язку з впровадженням програмного забезпечення, відносяться:

– підвищення продуктивності праці;

– вивільнення робочого часу.

Визначимо економічну ефективність, ґрунтуючись на тому, що впровадження програмного забезпечення вивільняє 0,5 працівника (за експертною оцінкою фахівців підприємства).

Оплата праці 0,5 працівника за рік складає:

10000*12*0,5=60 000,00 грн.

Річний економічний ефект визначається за формулою:

$$Е_{рік} = \Delta С_п - Е_н * К, \quad (5.3)$$

де $\Delta С_п$ – вивільнені кошти після впровадження програмного забезпечення = 60 000,00-3 308,08 =56 691,92

$Е_н$ – нормативний коефіцієнт економічної ефективності (дорівнює коефіцієнту амортизації 0,25%);

К – капітальні витрати на впровадження програмного забезпечення 35 574,52 грн.

Річний економічний ефект становить:

$Е_н$ = 56 691,92 - 35 574,52*0,25 = 47 798,29 грн.

Строк окупності капітальних вкладень визначається за наступною формулою:

$$T = \frac{K}{\Delta C_n}, \quad (5.4)$$

Строк окупності становить:

Т = 35 574,52/56 691,92 = 0,63 року

Отже строк окупності ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin складає приблизно 7,5 місяців.

5.4 Висновки

За отриманим значенням річного економічного ефекту і строку окупності можна зробити висновок, що розробка ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin є економічно вигідною та обґрунтованою.

6 ОХОРОНА ПРАЦІ

Охорона праці – система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів і засобів, направлених на збереження і працездатності людини в процесі трудової діяльності.

Дана система включає в себе:

– аналіз шкідливих та небезпечних факторів, які впливають на людину;

– розробка заходів щодо усунення шкідливого впливу на людину та створення нормальних умов праці;

– правила з техніки безпеки та виробничої санітарії;

– норми з охорони праці жінок, неповнолітніх та осіб з низькою працездатністю;

– спеціальні норми охорони праці осіб, які працюють в тяжких, шкідливих та небезпечних виробничих умовах.

Безпека людини на виробництві залежить від багатьох факторів. Здебільшого вона визначається кваліфікацією та відповідальністю робітників управління.

Велике значення для підвищення безпеки праці у виробництві має система стандартів безпеки праці, яка включає значну кількість взаємопов’язаних стандартів, розроблених для попередження ушкоджень і захворювань працюючих.

Закон України “Про охорону праці” визначає положення щодо реалізації права на охорону їх життя і здоров’я в процесі трудової діяльності, регулює за участю відповідних органів відносини між власником підприємства, установи і організації і робітником з питань безпеки, гігієни праці і виробничої санітарії, встановлює єдиний порядок організації охорони праці на Україні.

Поліпшення умов праці, підвищення їх безпеки нешкідливості має велике економічне значення. Воно впливає на економічні результати виробництва, на продуктивність праці, якість і собівартість продукції, що виготовляється.

Підвищення умов праці призводить до таких соціальних результатів, як покращення здоров’я працюючих, ріст ступеня задоволення працею, зміцнення трудової дисципліни, підвищення престижу ряду професій, ріст виробничої активності і поліпшення ряду інших показників, які характеризують більш високий ступінь розвитку працюючих.

Охорона природи, оскільки очищення і знешкодження стічних вод і викидів в атмосферу, боротьба із шумом і вібрацією, захист від електромагнітних полів та іонізуючих випромінювань служать не тільки цілям охорони праці, але й одночасно сприяють охороні середовища мешкання людини.

6.1 Аналіз небезпечних і шкідливих факторів у відділі розробки програмного забезпечення

При роботі співробітники відділу зазнають впливу небезпечних і шкідливих факторів, як-то:

– недостатня освітленість робочого місця;

– підвищений рівень шуму і вібрації від роботи вентиляторів, принтерів;

– висока ймовірність виникнення пожеж в зв’язку з наявністю великої кількості паперу;

– підвищена напруга в електричній мережі, замикання якої може відбутися через тіло людини;

– вплив неіонізуючих електромагнітних випромінювань, електростатичних і магнітних полів, які виникають при роботі комп’ютерів;

– недостатній рівень параметрів мікроклімату і температури, відносної вологості повітря, швидкості руху повітря.

Робота співробітника вимагає від нього значної зорової уваги, і як наслідок, у приміщенні повинно бути правильно спроектоване і розташоване освітлення.

В залежності від часу доби і джерела світла на підприємстві використовується освітлення трьох видів: природне, штучне і комбіноване.

У денний час і ясну погоду використовується природне освітлення через світлові віконні пройми. Для нього характерна висока розсіяність і це є сприятливим фактором для здорової діяльності.

У передвечірній час і за несприятливих погодних умовах використовується комбіноване освітлення. Оскільки денне світло не забезпечує достатню освітленість робочого місця, то використовуються настільні лампи місцевого освітлення.

У вечірній час взимку використовується штучне освітлення. В якості джерел використовуються лампи накалювання.

Рекомендована і дійсна освітленість у приміщенні відділу товарно-матеріальних цінностей складає 750лк.

Одним із найшкідливіших факторів, що чинить негативний вплив на працездатність робітників, є шум. Джерелами шуму у відділі є вентилятори, ЕОМ.

Для забезпечення нормативного рівня шуму у виробничому приміщенні відділу і на робочих місцях використовуються шумопоглинаючі засоби: важкогорючі спеціальні перфоровані плити, мінеральна вата з максимальним коефіцієнтом звукопоглинання в межах частот 31,5-8000Гц.

Рівень вібрації при виконанні робіт з використанням ЕОМ не повинен перевищувати допустимих значень. Оскільки в бюро використовуються 2 ПК з одним струйним принтером, то рівень вібрації, який створюють один принтер і два процесори, знаходиться в межах допустимих норм.

У відділі є меблі, які являються швидкозагораючим предметом. Існує ризик у випадку виникнення пожежі, швидке її поширення по приміщенню.

Електричний струм, проходячи через тіло людини справляє тепловий, хімічний, біологічний вплив, порушуючи нормальну життєдіяльність організму. Будь-який з цих впливів може привести до електротравм.

У відділі є холодильник, електрочайник, кондиціонер, касовий апарат. Електровмикання здійснюється через трьохштирьові штепсельні вилки, які заземлені на штатну систему заземлення.

Робота комп'ютера супроводжується електромагнітним і електростатичним випромінюванням. Навколо електростатично-зарядженого монітору підвищується концентрація пилу. Такий електризований пил може викликати запалення шкіри, а електромагнітне випромінювання може призвести до зниження загальної продуктивності співробітника.

Потужність дози рентгенівського випромінювання у відділі на відстані 0,05м від екрану не перевищує 7,74*10-12А/кг, а вміст озону в повітрі робочої зони відповідає еквівалентній дозі 0,1мг/м3, окислів азоту – 5мг/м3, вміст пилу – 4мг/м3.

Мікрокліматичні параметри (температура, відносна вологість повітря, швидкість руху повітря) впливають у першу чергу на продуктивність праці і самопочуття людини, а також на надійність роботи обчислювальної техніки. Забезпечення достатньої роботи системи вентиляції і опалення також підтримують працездатність співробітника.

Приміщення відділу обладнане системою опалення, кондиціонування повітря. Параметри мікроклімату, які діють у відділу, відповідають встановленим нормам.

6.2 Розрахунок освітленості у відділі розробки програмного забезпечення

До сучасного виробничого освітлення висуваються наступні вимоги:

– відповідність рівня освітленості робочих місць характеру роботи, що здійснюється;

– досить рівномірний розподіл яскравості на робочих місцях поверхнях і у навколишньому просторі;

– неперервність освітленості у часі;

– оптимальна направленість світлового потоку, який випромінюють світлові прилади;

– довговічність, економічність, електро- і вогненебезпека, естетичність, зручність і простота експлуатації.

Штучне освітлення за конструктивним виконанням може бути загальним і місцевим.

При загальному освітленні всі робочі місця отримують освітлення від загальної освітлюваної установки. Комбіноване освітлення поряд із загальним включає місцеве освітлення, що зосереджує світловий потік безпосередньо на робочих місцях. Використання лише місцевого освітлення не допустимо, оскільки виникає необхідність постійної адаптації зору, створюються тіні та інші негативні фактори.

Для штучного освітлення приміщень використовують люмінесцентні лампи. В них висока світлова віддача і тривалий термін служби. Разом із тим необхідно врахувати недоліки: висока пульсація світлового потоку, необхідність використання спеціальної апаратури, складність їх утилізації із-за наявності у лампах парів ртуті.

Розрахунок освітлення здійснюється для приміщень завдовжки 5м і завширшки 6м методом коефіцієнту використання світлового потоку наступними етапами:

1. Визначення світлового потоку (F), що падає на поверхню, здійснимо за формулою (6.1):

$$F = \frac{E_{min} \times k_з \times S \times z}{\eta} \quad \text{(Лм)}, \tag{6.1}$$

де E_{min} – нормована мінімальна освітленість, Лк (визначається за таблицею). В нашому випадку E_{min} =300Лк;

$k_з$– коефіцієнт запасу, який враховує зменшення світлового потоку лампи в результаті забруднення світильників в процесі експлуатації. В нашому випадку $k_з$ =1,5;

S – площина приміщення, яке освітлюється (S=30м^2);

z – коефіцієнт використання світлового потоку з урахуванням коефіцієнту відображення стелі =70% і стін =50%; виражається відношенням світлового потоку, який падає на розрахункову поверхню, до сумарного потоку всіх ламп і виражається у долях одиниці.

Значення η визначається за таблицею коефіцієнтів використання. Для цього розрахуємо індекс приміщення (i) за формулою (6.2):

$$i = \frac{S}{h \times (A+B)}, \tag{6.2}$$

де h – розрахункова висота підвісу, h=2,7м;

A – ширина приміщення, A=5м;

B – довжина приміщення, B=6м.

$$i = \frac{30}{2.7 \times (5+6)} = 1.01$$

За таблицею знаходимо коефіцієнт використання люмінесцентних світильників η=0,32.

Підставимо у формулу і отримаємо:

$$F = \frac{300 \times 1.5 \times 30 \times 1.1}{0.3} = 49500\text{Лм}$$

Для освітлення приміщення даного відділу обираємо люмінесцентні лампи типу ЛДУ 80, світловий потік яких дорівнює 4070Лм.

2. Розрахуємо необхідну кількість світильників для приміщення з комп’ютерами при загальному рівномірному освітленні.

Кількість необхідних ламп N розраховується за формулою (6.3):

$$N = \frac{F}{F_л}, \tag{6.3}$$

де $F_л$– світловий потік, 4070Лм.

$$N = \frac{49500}{4070} = 13{,}6 \approx 14\text{шт}$$

Із розрахунку освітленості виробничого приміщення виходить, що для правильної подачі світла і правильної освітленості робочих місць відділу необхідно використовувати світильники типу УПМ. Кожний світильник комплектувати 2 лампами. Розмістити світильники двома паралельними рядами по три в кожному ряді і один світильник перпендикулярно двом паралельним у середині приміщення.

6.3 Розробка заходів щодо зменшення впливу небезпечних і шкідливих факторів

Для зменшення впливу небезпечних і шкідливих факторів у приміщенні необхідно здійснити ряд заходів.

1.Для правильної подачі світла і правильної освітленості робочих місць слід використовувати світильники типу УПМ. Кожний світильник комплектувати 2 лампами. Розташувати світильники двома паралельними рядами по три в кожному ряді і один світильник перпендикулярно двом паралельним усередині приміщення.

2.Зниження впливу шуму на робітників відділу можна досягти шляхом більш раціональною розташування джерел шуму у приміщенні або зменшення рівня шуму у джерелі його виникнення, використання додаткової звукоізоляції і звуко погашення, впровадження малошумного обладнання(більш сучасного) і правильного планування режиму праці і відпочинку робітників.

3.Зменшити ймовірність виникнення пожежі у відділі можна шляхом зменшення документообігу, наприклад, повністю автоматизувати обліково-аналітичний процес, при якому значна кількість документів буде залишатись в електронній формі.

4.Для забезпечення безпечної для робітників відділу безаварійної роботи електроприладів необхідно поряд з їх положенням виконанням і обладнанням захисними засобами так організувати їх експлуатацію, щоб повністю виключити будь-яку ймовірність помилок з боку облікових робітників. Для організації безпечної експлуатації електроприладів необхідно підвищити технічну грамотність і свідому дисципліну робітників, які зобов’язані чітко дотримуватись організаційних і технічних заходів, а також прийомів і черговості виконання експлуатаційних операцій у відповідності з Правилами технічної експлуатації електроприладів споживачами і Правилами техніки безпеки при експлуатації електроприладів споживачами.

5. Знизити вплив електромагнітних випромінювань на співробітників при роботі на комп’ютері можна шляхом встановлення захисних екранів. Такий спосіб захисту є дуже ефективним і може бути досягнутий шляхом раціонального вибору конструкції екрану і матеріалу для екранування.

6. Наявність пилу, озону, можна знищити шляхом обладнання робочого приміщення системою кондиціонування повітря.

Всі ці заходи у сукупності сприяють злагодженій, безперебійній роботі відділу розробки програмного забезпечення, дозволять підвищити продуктивність праці робітників.

7 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

7.1 Сучасний екологічний стан Миколаївщини

Напружена екологічна ситуація у багатьох районах і містах країни свідчить про те, що незважаючи на посилення останнім часом уваги до цих питань і значні витрати на їх вирішення, вжиті заходи не досить ефективні і не зумовлюють змін у тенденції погіршення стану довкілля.

Структура промислового виробництва, що склалася в Україні, характеризується інтенсивним споживанням енергії, сировинних, водних і земельних ресурсів, а також збільшенням навантаження на довкілля.

Так, в Україні в 1996 році було викинуто в атмосферу близько 6,34 млн. тонн забруднюючих речовин, в тому числі 4,76 млн. тонн – зі стаціонарних джерел, 1,58 млн. тонн – з пересувних. За період 1992-1996 рр. загальний обсяг викиду забруднюючих речовин в атмосферне повітря скоротився зі стаціонарних джерел на 45 відсотків, з пересувних – на 12 відсотків. Станом на 1998 рік було викинуто в атмосферу 85 відсотків свинцю, 49 відсотків окису вуглецю та 31 відсоток вуглеводнів.

На сьогодні екологічну ситуацію як в м. Миколаєві, так і в Україні, можна охарактеризувати як кризову. Забруднення довкілля досягло такого рівня, коли воно негативно впливає на здоров'я населення.

Охорона навколишнього середовища від негативного впливу діяльності людей є першочерговою проблемою для міста.

До основних екологічних проблем міста Миколаєва відносяться:

– критичний стан атмосфери, зростання до небезпечних меж концентрації ряду хімічних речовин (оксидів азоту, сірки, діоксиду вуглеводу тощо) у повітряному просторі міста, неприпустимий рівень забруднення повітря;

– забруднення до критичних рівнів стічними водами та шкідливими викидами гідросфери, небезпечні забруднення не тільки поверхневих, але і підземних вод;

– техногенне забруднення літосфери внаслідок відсутності ефективних технологій утилізації хімічних, промислових та побутових відходів.

Основними передумовами, що збільшують виникнення екологічної загрози, є:

– невиконання природоохоронного законодавства, відсутність контролю за газоочисним устаткуванням та системою їх експлуатації;

– проникнення технологій, що не забезпечують безпеку природи та людини;

– застосування речовин, які завдають шкоди навколишньому середовищу;

– збільшення кількості автотранспорту з високим рівнем забруднення у вихлопних газах.

Реалізація екологічних загроз може призвести до збільшення числа захворювань серед мешканців міста, до скорочення тривалості життя, до загострення протиріч між виробництвом та природоохоронною діяльністю.

Як свідчать результати агрохімічного обстеження грунтів області, якісні показники їх родючості значно погіршилися. Вміст гумусу зменшився на 0,3 відсотка і становить 3,28 відсотка. Вміст рухомого азоту, фосфору, обмінного калію знизився на 8,14 відсотків і становить відповідно 14; 88; 152 мг/кг ґрунту. Збільшилися площі солонцюватих та змитих ґрунтів. Розораність сільськогосподарських угідь становить 84,7 відсотка. Внесення органічних добрив скоротилося з 78 до 0,5 т/га, а мінеральних майже в 20 разів. В області налічується близько 6,7 тис.га слабокислих ґрунтів, які поширені головним чином у південній та південно-східній частині регіону.

В результаті кризових явищ екстенсивне використання землі призвело до дисбалансу в землеробстві. Розширення ріллі до розмірів, неприпустимих

для розвинутих країн світу, погіршило її якісний стан, екологічну рівновагу навколишнього середовища, збільшило енергоспоживання.

Висока концентрація промислових підприємств в обласному центрі призвела до значного рівня забруднення річок Південний Буг та Інгул. Щороку у річки на території міста надходять стоки з 9 об'єктів, з них без очищення або з очищенням, що не відповідає санітарно-гігієнічним вимогам, 5 випусків господарсько-побутових стічних вод та 20 випусків промислових стічних вод.

Залишається незадовільним стан водопровідних мереж. Проби питної води не відповідають вимогам державних стандартів за органолептичними властивостями, загальною мінералізацією, вмістом хімічних речовин, а також за мікробіологічними показниками. Щороку близько 20 відсотків досліджених проб з поверхневих водоймищ I категорії за санітарно-хімічними та до 1-2% за мікробіологічними показниками не відповідають встановленим нормам.

Існуючими технологіями знезаражування питної води передбачене широке застосування хлору, внаслідок чого в ній утворюються шкідливі хлорорганічні сполуки.

Відомо також, що через високий ступінь мінералізації питної води, який спостерігається в Матвіївні, Тернівці, Великій та Малій Коренихах, збільшується кількість захворювань на хвороби шлунково-кишкового тракту, у тому числі гастрити, жовчнокам'яну та сечокам'яну хвороби.

На сьогодні в місті немає відповідної інфраструктури для забезпечення поводження з токсичними, промисловими, радіоактивними відходами. У місті немає полігону для поховання промислових відходів першого і другого класу небезпечності, які б повністю відповідали технічним та санітарно-гігієнічним вимогам. Немає також заводу з переробки токсичних промислових відходів.

Полігон для твердих побутових відходів експлуатується з порушенням санітарно-гігієнічних вимог, без запобіжних заходів щодо забруднення

підземних вод та повітряного басейну. Земельні ресурси полігону майже вичерпані. Усе це призводить до посилення соціального напруження серед місцевого населення.

Звалища побутових відходів є постійними джерелами забруднення довкілля, місцями розмноження комах, гризунів, бродячих тварин, що переносять збудників інфекцій, а також спричиняють зростання захворюваності на інфекційні хвороби.

Після зупинки багатьох миколаївських підприємств викиди в атмосферне повітря від стаціонарних джерел зменшилися.

За даними проведеного санепідемслужбою міста аналізу лабораторних досліджень у 2018 році перевищення граничнодопустимих концентрацій забруднюючих речовин виявлено в 69 дослідженнях (7,4%) із загальної їх кількості 928.

А у 2017 році не спостерігалося перевищень гранично допустимих концентрацій (ГДК) по пилу, діоксиду сірі, оксиду азоту, фтористому водню. Кількість перевищень ГДК забруднюючих речовин в атмосфері склала: по оксиду вуглецю – 27; по діоксиду азоту – 258; по формальдегіду –20.

Проте 66% всіх викидів доводиться на пересувні джерела. Деякі автомобілі в Миколаєві працюють на етілірованому бензині; багато машин старі з невідрегульованими двигунами. У місті є вулиці, де допустимі концентрації по свинцю, іншим важким металам, бензапірену, вуглекислому газу, оксидам азоту і сірки перевищені в десятки разів. Це – Пушкінська, пр. Героїв Сталінграду, пр. Леніна, Космонавтів та інші.

Транзитний транспорт, що йде з Одеси у бік Херсона і Києва, проїжджає практично через центр міста. Це сприяє сильному забрудненню атмосферного повітря в Центральному районі міста.

У 2017 році з 791 виміру по шуму на різних ділянках міста перевищували допустимі значення 159 вимірів (або 20,1%). На магістралях міста перевищення допустимого рівня шуму встановлене в 71,1% вимірів. Перевищення відмічене на перехрестях: вулиць Артилерійській –

Потьомкінській – в 75% вимірів; пр. Миру – вул. Космонавтів – 75%; пр. Леніна – пр. Жовтневий – 83,3%; пр. Жовтневий – пр. Корабелів - 83%; пр.Жовтневий – вул. Торгова – 83,3%; вул. Пушкінська – Нікольска, вул. Пушкінська – пр. Леніна, вул. В.Морська – Нікольська – в 100% вимірів. Таким чином, критичний стан атмосфери, неприпустимий рівень забруднення повітря, забруднення до критичних рівнів стічними водами та шкідливими викидами гідросфери, небезпечні забруднення поверхневих і підземних вод, техногенне забруднення літосфери характеризує екологічний стан міста Миколаєва і Миколаївщини взагалі як критичний. Ситуація, що склалася, потребує втручання на державному, регіональному і місцевому рівнях в усі сфери життєдіяльності суспільства культури і духовності, освіти та інформування, політики, економіки, законодавства, охорони здоров'я.

7.2 Забруднення навколишнього середовища комп'ютерною компанією

Комп'ютерна компанія – це офісна компанія, яка займається наданням послуг з розробки та супроводження програмного забезпечення. Хоча компанія не займається промисловим виробництвом з викидом шкідливих речовин у атмосферу, воду та ґрунт, це не виключає існування джерел забруднення довкілля. До таких джерел, насамперед, належать:

– дизельна електростанція;

– твердопаливні котли;

– побутове сміття;

– електромагнітне випромінювання;

– відходи від застарілого обладнання та оргтехніки;

Одночасно в офісі компанії можуть працювати до 50 комп’ютерів, стаціонарних телефонів та велика кількість серверної техніки, яка обслуговує роботу цього обладнання, аварійне чи планове відключення електроенергії

призводить до значних перешкод у роботі персоналу. Для того, щоб уникнути таких ситуацій, на території компанії знаходиться дизельна електростанціє. Вона працює як в аварійному, так і в резервному режимі.

Результатом роботи дизельної електростанції є продукти згоряння дизельного палива та шумове забруднення.

Твердопаливні коти використовуються в холодну пору року в системі опалення офісних приміщень. Для опалення використовуються деревні пелети. Хоча таке джерело тепла має менший відсоток шкідливих викидів в атмосферу, ніж, наприклад, вугілля чи мазут, цей вид опалення не можна вважати екологічно чистим. Порівняльні характеристики продуктів згоряння палива наведені в табл. 7.1.

Таблиця 7.1 – Рівні викидів забруднюючих речовин в атмосферу при спалюванні різних видів палива

Вид палива	Викиди забруднюючих речовин в атмосферне повітря без систем очищення, тонн на 1 тис. тонн нат. палива				
	CO2	NO_2	SO_2	Тверді частинки (пил неорг.)	РАЗОМ
Природний газ	1,18	3,52	0,00	0,00	4,70
Древні брикети, пелети	4,68	9,31	0,28	4,11	17,69
Деревина дров'яна	4,9	9,4	0,3	4,3	18,9
Тирса деревна	5,0	9,6	0,5	5,0	20,0
Деревні відходи, обрізки	5,2	9,9	0,4	5,2	20,7
Швидкозростаюча деревина	4,8	9,5	0,0	8,4	22,7
Тріска, сучки, кора	5,6	11,4	0,8	13,4	31,3
Мазут	5,20	5,20	35,30	0,30	45,90
Брикет торф'яний	8,04	26,81	3,00	13,02	50,87
Кам'яне вугілля	9,58	63,56	9,20	65,32	147,66

Побутові відходи – це залишки речовин і предметів, які утворюються в результаті побутової та господарської діяльності людини і які не можуть бути використані на місці утворення, а їх накопичення і зберігання порушують санітарний стан навколишнього середовища.

Всі побутові відходи, які утворюються в процесі функціонування компанії, можна розділити на рідкі та тверді. До рідких побутових відходів належать нечистоти з вигребів туалетів, помиї (від миття посуду, підлоги, тощо) і стічні води (побутові, зливові). До твердих побутових відходів можна віднести сміття (побутові відходи) та кухонні відходи. Зокрема, твердими відходами є папір, картон, скло, пластмаса, продукти харчування.

Неймовірні темпи розвитку технологій привели до того, що утилізація оргтехніки стала особливо актуальною сьогодні. Комп`ютери та інші гаджети не просто стають малопотужними, але і застарівають морально. Нам доводиться йти в ногу з часом, яке змушує купувати нові ноутбуки і ПК.

До складу оргтехніки входить маса деталей, які можна піддати переробці. Утилізація оргтехніки та обладнання дозволяє зберегти стан навколишнього середовища і є досить прибутковим бізнесом. Тим більше, сьогодні ця послуга затребувана особливо сильно. Вже сьогодні кількість випущених комп`ютерів перевалило за один мільярд. Додайте сюди друковані та копіювальні машини - актуальність переробки відпрацьованих пристроїв зростає з їх числом.

Видалення техніки має на увазі складний ланцюг робіт. Але для початку потрібно визначитися з поняттям. За КВЕД і ОКПД комп`ютерний лом, переробка якого проводиться в спеціальних організаціях, ділиться на:

- монітори;
- системні блоки;
- друковано - копіювальні машини;
- шредери, плоттери;
- флеш – накопичувачі;
- жорсткі диски;
- мережеве обладнання;
- клавіатури, миші.

Всі вони мають свої особливості, від яких залежить подальша переробка електронної техніки. Хоча в цілому складові кожного пристрою

схожі, проте, утилізація моніторів відрізняється від процесу утилізації картриджів для принтерів. Всі вони вимагають спеціальної підготовки і обробки.

Найчастіше, коли мова заходить про старе комп`ютерне обладнання, згадують про зміст дорогоцінних металів. Але мало хто говорить про що до складу техніки входять сполуки ртуті, цинку, свинцю і кадмію. Поки вони знаходяться в складі оргтехніки, вони є безпечними, але потрапляючи на звалища і взаємодіючи з вологою, вони розпадаються на з`єднання, які отруюють грунт і воду. Тому дуже важливо правильно утилізувати комп`ютерну техніку.

Питання з дорогоцінними металами теж є не таким простим. Обладнання, що містить золото, срібло і інші елементи, знаходиться на балансі підприємства. Керівники не можуть просто взяти і викинути комп`ютерний лом на звалище, інакше підприємство отримає солідний штраф. Для цього потрібна наявність певної документації та сертифікатів.

Генератором електромагнітного забруднення є, в першу чергу, велика кількість комп’ютерної техніки на території офісу. Більшість із них працює 8 годин на добу, а деякі не вимикаються цілодобово. Випромінювачами в даному випадку є і процесор, і монітор. Випромінювання останнього значно вищі, особливо його бічні і задні стінки, адже вони не мають спеціального захисного покриття, як у лицьовій частині монітора. Крім того, в офісі є дві кухні, кожна з яких обладнана 5 мікрохвильовими печами.

Електромагнітне випромінювання має несприятливий вплив на організм людини. Його наслідками можуть бути головний біль, порушення сну, перевтома і, навіть, у деяких випадках стенокардія.

7.3 Розробка заходів щодо зменшення забруднення

Для того, щоб зменшити рівень забруднення навколишнього середовища, варто вжити певних заходів, які мінімізують цей вплив, якщо його неможливо усунути цілком. Серед таких заходів:

1) Очищення димових газів у мокрих золоуловлювачах.

Електрофільтри на електростанціях застосовуються для досягнення найбільш глибокого очищення димових газів в основному на великих енергоблоках потужністю 300 МВт та більше. Мокрі золоуловлювачі, які працюють при маленьких питомих витратах води та невеликих перепадах тиску, встановлюються за котлоагрегатами середньої паропродуктивності. У мокрих золоуловлювачах уловлювання часток золи на плівці води, яка стікає по його стінках, здійснюється за рахунок відцентрової сили, яка діє на частки. Ефективність апарата не перевищує 90%.

2) Використання природного газу для опалення офісних приміщень замість деревних пелетів.

Згідно з показниками, наведеними в табл. 7.1, рівень шкідливих викидів від згоряння природного газу є найнижчим серед перерахованих видів палива. Проте зміну джерела енергії слід проводити, враховуючи економічну ефективність, оскільки вартість цих видів палива не є однаковою.

3) Раціональне позбавлення від побутових відходів.

Деякі побутові відходи можна безпечно спалювати для отримання енергії. Вторинну сировину (макулатуру, скло, пластмаси) можна відправляти на переробку, а не вивозити на сміттєзвалища. Крім того, зменшити кількість побутових відходів допоможе повторне їх використання (наприклад, скляних чи пластикових пляшок та контейнерів для їжі), скорочення споживання та використання меншої кількості упаковки.

4) Зменшити рівень електромагнітного забруднення та захиститися від його надмірного впливу.

Зрозуміло, що неможливо повністю обійтися без електроприладів та комп’ютерів, проте можна дещо зменшити їх негативний вплив. Для цього слід вимикати з електромережі комп’ютери, телефони та обладнання, з якими ніхто не працює. Також варто час від часу виходити на прогулянки і робити перерви в роботі з комп'ютером.

Що стосується мікрохвильових печей, то їх потужність може змінюватись, тому час від часу треба звертатися до майстра, щоб контролювати рівень випромінювання.

5) Утилізація застарілої ортехніки та комп’ютерного обладнання.

У принципі, будь-який комп'ютер чи оргтехніку можна переробити і пустити у вторинне використання. При грамотній утилізації близько 95% відходів техніки здатні повернутися до нас в тому чи іншому вигляді, і приблизно 5% відправляються на звалища або заводи з переробки твердих побутових відходів.

Співвідношення ручної та автоматизованої праці на фабриках по переробці комп'ютерної техніки залежить від її типу. Для монітора це співвідношення приблизно 50 на 50 - розбирання старих кінескопів є досить трудомістким заняттям. Для системних блоків та оргтехніки частка автоматичних операцій вища.

HP вперше запропонувала переробку відслужив свій термін продукції ще в 1981 році. Сьогодні HP володіє інфраструктурою зі збору та переробки використаних ПК і оргтехніки в 50 країнах світу. У рік утилізації піддається близько 2,5 млн. одиниць продукції. В одному тільки 2007 HP переробив близько 100 тис. тонн списаного обладнання та витратних матеріалів, - майже в півтора рази більше, ніж роком раніше.

Будь-яка якісна переробка комп`ютерної техніки по КВЕД вимагає попередньої підготовки:

- Жорсткі диски проходять низкоуровневую Форматизация.
- Утилізація картриджів вимагає попереднього очищення від тонера, до складу якого входять шкідливі хімічні елементи.

В цілому весь електро лом обробляється, розбирається і сортується. Пластикові, металеві та скляні деталі оргтехніки переробляються окремо.

Перший етап завжди проводиться вручну. Це - видалення всіх небезпечних компонентів. У сучасних настільних ПК і принтерах таких компонентів практично немає. Але переробці піддаються, як правило, комп'ютери і техніка, випущені в кінці 90-х - самому початку 2000-х років, коли плоских рідкокристалічних моніторів просто не існувало. А в кинескопних моніторах міститься чимало сполук свинцю. Інша категорія продукції, містить небезпечні елементи, - ноутбуки. В акумуляторах і екранах застарілих моделей є певна кількість ртуті, яка також дуже небезпечна для організму. Важливо відзначити, що в нових моделях ноутбуків від цих шкідливих компонентів позбулися.

Потім видаляються всі великі пластикові частини. У більшості випадків ця операція також здійснюється вручну. Пластик сортується в залежності від типу і подрібнюється для того, щоб надалі його можна було використовувати повторно. Решту відправляють у великій подрібнювач-шредер, і всі подальші операції є автоматизованими. Багато в чому технології переробки запозичені з гірської справи - приблизно таким же способом витягують цінні метали з породи.

Подрібнені в гранули залишки комп'ютерів піддаються сортуванню. Спочатку за допомогою магнітів витягуються всі залізні частини. Потім приступають до виділення кольорових металів, яких в ПК значно більше. Алюміній добувають з брухту за допомогою електролізу. У сухому залишку виходить суміш пластику і міді. Мідь виділяють способом флотації - гранули поміщають в спеціальну рідину, пластик спливає, а мідь залишається на дні. Сама ця рідина не отруйна, проте, робітники на заводі використовують захист органів дихання - щоб не вдихати пил.

ВИСНОВКИ

Метою даної кваліфікаційної роботи було підвищення достовірності оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програми для його реалізації

Для досягнення поставленої мети були вирішені наступні завдання:

– виконаний аналіз існуючих методів та моделей оцінювання розміру ПЗ;

– обґрунтована необхідність удосконалення рівняння регресії для оцінювання розміру ПЗ з відкритим кодом на Kotlin;

– досліджені різні джерела з відкритим вихідним кодом та зібрані дані для удосконалення рівняння регресії;

– нормалізовані отримані емпіричні дані, використовуючи перетворення у вигляді десяткового логарифму;

– перевірені дані на викиди;

– побудоване лінійне рівняння регресії, довірчий інтервал та інтервал передбачення для вихідних даних;

– побудуване нелінійне рівняння регресії, довірчий інтервал та інтервал передбачення на основі нормалізованих даних;

– розроблене ПЗ для оцінювання розміру програмного забезпечення.

ПЗ для оцінювання розміру програмного забезпечення, яке розроблено в рамках магістерської роботи, дозволить автоматизувати та скоротити час відповідних розрахунків.

З урахуванням усього вищенаведеного, можна сказати, що цілі, поставлені на початку проектування, були повністю виконані.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Котов, С.Л. Информационно-аналитическая система оценивания трудозатрат и стоимости создания программных средств [Текст] / С.Л. Котов, А.А. Демирский // Программные продукты и системы. – 2017. – Т. 30, № 3. – С. 469-473

2. Введение в язык Kotlin [Електронний ресурс]. – Режим доступу: https://metanit.com/kotlin/tutorial/1.1.php – Назва з екрану

3.Табунщик, Г.В. Інженерія якості програмного забезпечення:навчальний посібник [Текст] / Г.В Табунщик, Р.К. Кудерметов, Т.І. Брагіна. – Запоріжжя: ЗНТУ, 2013. – 180 с.

4. Титов, А.И. Выбор метрики размера проекта в модели оценки трудоемкости разработки программ [Текст] / А.И. Титов // Интеллектуальные технологии на транспорте. –2016. – № 1. – С. 31-38.

5. Магнус, Я.Р. Эконометрика. Начальный курс: Учеб. – 6-е изд., перераб. и доп. [Текст] / Я.Р. Магнус, П.К. Катышев, А.А. Пересецкий. – М.: Дело, 2004. – 576 с.

6. Prykhodko, S. Multivariate outlier detection technique based on normalizing transformations for non-Gaussian data / S. Prykhodko, N. Prykhodko, L. Makarova, K. Pugachenko // «Інформаційні технології та комп`ютерне моделювання»: матеріали статей Міжнародної науково-практичної конференції, м. Івано-Франківськ, 15-20 травня 2017 року. – Івано-Франківськ: п. Голіней О.М., 2017. – С. 170-174.

7. Кожевніков, І.Г. Емпіричні моделі оцінки вартості розробки програмного забезпечення [Текст] / І.Г. Кожевніков // Економічний простір. – 2014. – №83. – С.195-208.

8. Нелінійна регресія. вибір і порівняння нелінійних регресійних моделей [Електронний ресурс]. – Режим доступу: http://dn.khnu.km.ua/dn/k_default.aspx?M=k1314&T=02&lng=1&st=0 – Назва з екрану

9. Chatterjee, Samprit. Handbook of Regression Analysis [Text] / Samprit Chatterjee, Jeffrey S. Somonoff. Wiley, 2012. – 240 p.

10. Приходько С.Б. Методичні вказівки до виконання лабораторних робіт з дисципліни "Обробка експериментальних даних на ЕОМ" [Текст] / С.Б. Приходько – Миколаїв: НУК , 2005. – 52 с.

11. Дрейпер, Н. Прикладной регрессионній анализ: В 2-х кн. Кн. 2 / Пер. с англ. – 2-е изд., перераб. и доп. [Текст] / Н. Дрейпер, Г. Смит – М.: Финансы и статистика, 1987. – 351 с.

12. Регресійний аналіз [Електронний ресурс]. – Режим доступу: https://pidruchniki.com/17280924/ekonomika/regresiyniy_analiz – Назва з екрану

13. Кобзарь, А.И. Прикладная математическая статистика. Для инженеров и научных работников [Текст] / А.И. Кобзарь. – М.: ФИЗМАТЛИТ, 2006. – 816с.

14. Айвазян, С.А. Прикладная статистика. Основы эконометрики: Учебник для вузов: В 2 т. 2-е изд., испр. – Т.1: Теория вероятностей и прикладная статистика [Текст] / С.А. Айвазян, В.С. Мхитарян. – М.: ЮНИТИ-ДАНА, 2001. – 656 с.

15. Вендров, А.М. Проектирование программного обеспечения экономических информационных систем [Текст] / А.М. Вендров. – М.: «Финансы и статистика», 2002. –349 с.

16. Леоненков, А.В. Самоучитель UML [Текст] / А.В. Леоненков. – СПб: БХВ, 2002. – 304 с.

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ ПЗ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВІДКРИТИМ КОДОМ НА KOTLIN

Темою роботи є «Удосконалення рівняння регресії для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin та розробка програмного забезпечення для його реалізації».

Коротка характеристика області застосування: ПЗ планується використовувати в комп’ютерних фірмах для автоматизації оцінювання розміру програмного забезпечення на Kotlin.

1 Підстава для розробки

Підставою для розробки програмного забезпечення є наказ на затвердження тем кваліфікаційних (магістерських) робіт кафедри ПЗАС НУК імені адмірала Макарова.

2 Призначення розробки програмного забезпечення

2.1 Функціональне призначення програмного забезпечення

Програмне забезпечення повинне автоматизувати процес оцінювання розміру програмного забезпечення на Kotlin.

2.2 Експлуатаційне призначення програмного забезпечення

Розроблюване програмне забезпечення повинно надавати змогу розв'язувати ряд проблем, що виникають в комп’ютерних фірмах при оцінюванні розміру програмного забезпечення на Kotlin, з метою їх спрощення, прискорення та підвищення достовірності.

3 Вимоги до програмного забезпечення

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу виконуваних функцій

Програмне забезпечення повинне виконувати наступні функції:

1) надавати можливість вносити дані про програмні проекти;
2) виконувати нормалізацію внесених даних;
3) будувати довірчі інтервали на основі нормалізованих даних;
4) будувати інтервали передбачення на основі нормалізованих даних;
5) визначати коефіцієнти нелінійного рівняння регресії;
5) розраховувати метрики його якості.
6) оцінювати розмір програмних проектів.

3.1.2 Вимоги до організації вхідних та вихідних даних

Вхідною інформацією для даної системи є база даних, яка постійно змінюється та поповнюється, а також інші дані, необхідні для розрахунків, які повинні вводитися користувачем у діалогових формах.

Вихідною інформацією є розраховані метрики якості, довірчі інтервали та інтервали передбачення, а також результати оцінювання.

Документи повинні виводитись як на монітор користувача, так і на пристрої друку.

3.2 Вимоги до надійності

3.2.1 Вимоги до надійного функціонування програмного забезпечення (перезавантаження, копіювання, відновлення)

У разі виникнення збою в роботі апаратури, відновлення нормальної роботи програмного забезпечення повинно виконуватися після перезавантаження операційної системи, повторного виконання дій, втрачених після останнього збереження інформації на зовнішньому носії. Програма повинна забезпечувати коректну обробку виняткових ситуацій. Для

забезпечення надійної роботи програмного забезпечення необхідно передбачити можливість резервного копіювання інформації. Для виконання резервного копіювання слід використовувати вбудовані можливості операційної системи.

3.2.2 Вимоги до контролю вхідної та вихідної інформації

Програма повинна забезпечувати правильне введення інформації за рахунок використання там, де це доцільно, шаблонів введення, процедурного блокування введення некоректної інформації, списків і автопідстановок.

Обробка виняткових ситуацій, пов’язаних з доступом до дисків, пристрою введення-виведення інформації повинна виконуватися програмою з виведенням відповідних інформаційних повідомлень та не призводити до блокування роботи програми.

3.2.3 Вимоги до часу відновлення після відмови

Час відновлення після відмови, не пов’язаною з роботою програми, повинен складатися з часу перезапуску операційної системи, часу повторного введення або зчитування з носіїв втрачених даних.

3.3 Умови експлуатації

Умови експлуатації даної розробки повинні відповідати умовам експлуатації персональних ЕОМ. Обслуговування повинно проводитись людьми, що відповідають за введення, корегування даних, а також мають знання основ предметної галузі.

3.4 Вимоги до складу та параметрам технічних засобів

Для роботи програмного забезпечення необхідна наявність у користувача персонального комп'ютера, що має такі системні характеристики:

- ПК з процесором не нижче: Intel Core i3-6100;

– обсяг оперативної пам‘яті – не менше 4 Гб;

– необхідний обсяг на жорсткому диску – не менше 500 Мб;

– монітор;

– клавіатура;

– миша;

– принтер.

3.5 Вимоги до інформаційної та програмної сумісності

Для роботи з програмним забезпеченням необхідно установити на комп'ютер наступні програмні продукти:

– операційну систему MS Windows XP/7/8/8.1/10.

– пакет Microsoft Office;

– наявність .NET Framework (4.5 і вище версії).

3.6 Вимоги до маркування й пакування

Для зберігання даного програмного продукту необхідно використовувати лазерні компакт-диски, отже, вимоги до маркування та упаковки пред’являлися відповідно аналогічних вимог для компакт-дисків.

3.7 Вимоги до транспортування та зберігання

Програма має зберігатися та транспортуватися, як усі CD-DVD диски згідно ДСТУ.

Програма повинна зберігатися у вигляді двох маркованих дискових копій: еталонної та робочої. При маркуванні дисків вказується:

– номер версії програмного забезпечення;

– дата запису програмного забезпечення;

– дата його наступного перезапису;

Періодичний перезапис інформації на дисках повинен здійснюватися відповідно до нанесеного маркування.

Для забезпечення коректної робот програма встановлюється на жорсткий диск. Для нормального функціонування програмного засобу в приміщенні мають бути забезпечені наступні параметри: температура навколишнього середовища в діапазоні від +15^{o}C до +35^{o}C , відносна вологість 10-75%.

4 Вимоги до програмної документації

Документація до програмного засобу повинна містити наступні документи:

- технічне завдання;
- опис програмного забезпечення;
- текст програмного забезпечення;
- програму та методику випробувань;
- інструкцію користувача.

5 Техніко - економічні показники

Розрахунок техніко-економічних показників розробки даного програмного забезпечення та введення його в дію представлений в 5 розділі кваліфікаційної роботи.

6 Стадії та етапи розробки

Стадії та етапи розробки програмного забезпечення та терміни їх виконання повинні відповідати затвердженому графіку виконання кваліфікаційної роботи та зводитися до таких, що наведені у таблиці А1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Стадії розробки	Етапи робіт	Вміст робіт	Контрольні точки
Технічне завдання	Обумовлення необхідності розробки	Постановка задачі. Збір початкових матеріалів.	З 08.09.2020 до 12.09.2020
	Розробка та затвердження технічного завдання	Визначення вимог до ПЗ. ВОбговорення та затвердження технічного завдання	З 13.09.2020 до 20.09.2020
Ескізний проект	Розробка ескізного проекту	Попередня розробка структури вхідних та вихідних даних, уточнення методів рішення завдань, загальний опис алгоритму	З 21.09.2020 до 21.10.2020
	Затвердження ескізного проекту	Розробка пояснювальної записки. Узгодження і затвердження ескізного проекту	З 22.10.2020 до 24.10.2020
Технічний проект	Розробка технічного проекту	Уточнення структури вхідних та вихідних даних, розробка алгоритму рішення задачі, розробка структури ПЗ	З 25.10.2020 до 15.11.2020
	Затвердження технічного проекту	Розробка пояснювальної записки. Узгодження і затвердження технічного проекту	З 16.11.2020 до 18.11.2020
Робочий проект	Розробка ПЗ	Програмування та налагодження програми	З 19.11.2020 до 29.11.2020
	Розробка програмної документації	Розробка програмної документації відповідно до вимог ГОСТ 19.101-77	З 30.11.2020 до 10.12.2020
	Випробування програми	Розробка, узгодження та затвердження програми і методики випробувань, коригування програми і програмної документації за результатами випробувань	З 11.12.2020 до 14.12.2020

7 Порядок контролю та приймання

Контроль здійснюється замовником на кожній стадії розробки ПЗ. Приймання здійснюється замовником за програмою та методикою випробувань. Випробування проводиться в зазначений термін.

Хід проведення приймально-здавальних випробувань документується за допомогою протоколу проведення випробувань. На підставі протоколу проведення випробувань виконувач сумісно з замовником підписують акт приймання-здачі програми в експлуатацію.

ДОДАТОК Б

ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ ПЗ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВІДКРИТИМ КОДОМ НА KOTLIN

1 Об’єкт випробувань

Об’єктом випробовувань є ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin. Функціонує під управлінням операційної системи MS Windows XP/7/8/8.1/10.

2 Мета випробувань

Програма випробувань програмного забезпечення переслідує цілі:

– перевірка придатності розробленого програмного забезпечення до використання;

– перевірка придатності розробленого програмного забезпечення для розв’язання задач у предметній галузі, для якої даний програмний продукт був створений;

– перевірка досягнутих характеристик програмного продукту.

Метою випробувань є оцінити відповідність програмного забезпечення вимогам, сформульованим у технічному завданні, наведеному у Додатку А і виявити помилки проектування. Якщо прийнятий рівень відповідності програмного забезпечення досягнутий, а також відсутні помилки і грубі недоліки, то програмне забезпечення приймається в експлуатацію.

3 Вимоги до програми

При проведенні випробувань функціональні характеристики програми підлягають перевірці на відповідність вимогам, викладеним у пункті «Вимоги до функціональних характеристик» Технічного завдання.

4 Вимоги до програмної документації

У комплект програмної документації повинні входити наступні документи:

- технічне завдання;
- інструкція користувача;
- текст програми;
- програма й методика випробувань.

5 Засоби та порядок випробувань

Засоби випробувань.

Для роботи з програмним забезпеченням необхідно установити на комп'ютер наступні програмні продукти:

– операційну систему MS Windows XP/7/8/8.1/10.

– пакет Microsoft Office;

– наявність .NET Framework (4.5 і вище версії).

Для роботи із програмою необхідний наступний склад технічних засобів з мінімальними параметрами:

– ПК з процесором не нижче: Intel Core i3-6100;

– обсяг оперативної пам‘яті – не менше 4 Гб;

– необхідний обсяг на жорсткому диску – не менше 500 Мб;

– монітор;

– клавіатура;

– миша;

– принтер.

Системні програмні засоби, використовувані програмою, повинні бути представлені ліцензійною локалізованою версією операційної системи.

Порядок проведення випробувань:

– виконуються інструкції до кожної функції;

– робиться аналіз отриманих результатів і встановлюється відповідність програмного продукту вимогам і системним документам.

При виявлені помилок у роботі системи складається їх перелік і обговорюється термін їх виправлення розробником. Після цього замовник проводить повторне тестування в повному обсязі (можливе використання нових чи додаткових тестів).

6 Методи випробувань

Випробування будемо проводити за стратегією «чорного ящика». За рахунок дуже великої кількості функцій, які потрібно випробувати, представимо тільки декілька результатів випробувань функцій програми. Всі функції програми будуть проходити випробування.

Складемо програму випробувань:

Випробування можливості введення даних з файлу

– Натиснути кнопку «Вибрати файл».

– Відкрити вікно, в якому обрати шлях до файлу.

– Файл повинен відкритися і поблизу кнопки повинна з’явитися назва файлу.

Після натискання кнопки «Вибрати файл» файл повинен додатися.

Перевірка можливості продивитися довірчі інтервали та інтервали передбачення.

– Ввести дані.

– Натиснути на кнопку «Показати інтервали».

– Інтервали повинні відобразитися в вигляді таблиці.

– Кнопка «Показати інтервали» повинна змінити текст на «Сховати інтервали».

Після натискання кнопки «Показати інтервали» на головній формі має додатися таблиця, яка містить ці інтервали.

Перевірка можливості сховати інтервали

– Ввести дані та натиснути кнопку «Показати інтервали».

– Натиснути кнопку «Сховати інтервали».

– Таблиця з інтервалами повинна зникнути із основної форми.

– Кнопка «Сховати інтервали» має змінити текст на «Показати інтервали».

ДОДАТОК В

ОПИС ПЗ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВІДКРИТИМ КОДОМ НА KOTLIN

1 Загальні відомості

1.1 Позначення і найменування програми

Найменування ПЗ: ПЗ для оцінювання розміру програмного забезпечення з відкритим кодом на Kotlin.

Позначення: «S_Kotlin».

1.2 Програмне забезпечення, необхідне для функціонування програми

Програмне забезпечення «S_Kotlin» є крос-платформним і основною необхідною вимогою для функціонування виробу є наявність .NET Framework (4.5 і вище версії).

1.3 Мови програмування

ПЗ «S_Kotlin» реалізовано на мові високого рівня C#. Модулі, реалізовані на інших мовах програмування, не використовуються.

2 Функціональне призначення

2.1 Класи вирішуваних завдань і призначення програми

ПЗ «S_Kotlin» повинно забезпечувати можливість виконання нижче перерахованих функцій:

1) надавати можливість вносити дані про програмні проекти;

2) виконувати нормалізацію внесених даних;

3) будувати довірчі інтервали на основі нормалізованих даних;

4) будувати інтервали передбачення на основі нормалізованих даних;

5) визначати коефіцієнти нелінійного рівняння регресії;

5) розраховувати метрики його якості.

6) оцінювати розмір програмних проектів.

2.2 Функціональні обмеження

Файли повинні бути формату EXE.

3 Використовувані технічні засоби

ПЗ «S_Kotlin» призначене для використання на персональних комп'ютерах, що працюють під управлінням наступних операційних систем: MS Windows XP/7/8/8.1/10

Для ПЗ «S_Kotlin» пред'являються такі апаратні вимоги до ПЕОМ:

– ПК з процесором не нижче: Intel Core i3-6100;

– обсяг оперативної пам‘яті – не менше 4 Гб;

– необхідний обсяг на жорсткому диску – не менше 500 Мб;

– монітор;

– клавіатура;

– миша;

– принтер.

Швидкість роботи ПЗ «S_Kotlin» на конкретному комп'ютері залежить також від характеристик окремих його комплектуючих (процесора, оперативної пам'яті і ін.).

4 Виклик і завантаження

Запуск ПЗ в графічному режимі здійснюється двома способами:

1) за допомогою файлу запуску: «S_Kotlin.exe»;

2) з використанням командного рядку: «S_Kotlin.exe».

5 Вхідні і вихідні дані

Вхідні дані програми (функції та строки коду) повинні знаходитись в окремому текстовому файлі у два стовбці. Вихідні дані програми мають відображатися на екрані користувача.

ДОДАТОК Г

ТЕКСТ ПРОГРАМИ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Oproximator.cs

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace WindowsFormsApp1
{
 class Oproximator
   {
      private double a;
      private double b;

      public Oproximator(double[] x, double[] y, bool check )
      {
         double sumx = 0;
         double sumy = 0;
         double sumx2 = 0;
         double sumxy = 0;

         int n = x.Length;

         for (int i = 0; i < n; i++)
         {
```

```
            sumx += x[i];
            sumy += y[i];
            sumx2 += x[i] * x[i];
            sumxy += x[i] * y[i];
        }

        a = (n * sumxy - sumx * sumy) / (n * sumx2 - sumx * sumx);
        b = (sumy - a * sumx) / n;
        if (check == true)
        {
            b += 0.025;
        }
    }

    public double getA()
    {
        return a;
    }

    public double getB()
    {
        return b;
    }

  }
}
```

fileReader.cs

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
using System.Diagnostics;

namespace WindowsFormsApp1
{
   class FileReader
   {
      private double[] x;
      private double[] y;
      private string path;

      public void read()
      {
         using (var reader = new StreamReader(path))
         {
            string row;
            int i = 0;

            var lineCount = File.ReadLines(path).Count();

            x = new double[lineCount];
            y = new double[lineCount];

            while ((row = reader.ReadLine()) != null)
            {
               try
               {
                                    x[i] = Double.Parse(row.Split(new[] { ' ' },
                           StringSplitOptions.RemoveEmptyEntries)[0]);

               }
               catch (Exception e)
               {
                                    throw new Exception("Неправильный
                           формат количества функций");
               }

               if (x[i] <= 0)
               {
                  throw new Exception("Количество функций меньше
                                 или равно 0");
```

```
                }

                try
                {
                                y[i] = Double.Parse(row.Split(new[] { ' ' },
                        StringSplitOptions.RemoveEmptyEntries)[1].Trim
                        Start());

                }
                catch (Exception e)
                {
                   throw new Exception("Неправильный формат
                                количества строк кода");
                }

                if (y[i] <= 0)
                {
                                throw new Exception("Значение строк кода
                        меньше или равно 0");
                }

                i++;

            }

        }
    }

    public double[] GetLines()
    {
       return x;
    }
```

lgNormalizer.cs

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace WindowsFormsApp1
```

```
{

    class LgNormalizer:Normalizer
    {

        public LgNormalizer(double[] x, double[] y) : base(x, y, true)
        {
                }

        public override void Reverse()
        {
            for (int i = 0; i < length; i++)
            {
                Y[i] = Math.Pow(10, Yoproximated[i]);
                YMinusDelta[i] = Math.Pow(10, yMinusDelta[i]);
                YPlusDelta[i] = Math.Pow(10, yPlusDelta[i]);
                                YMinusDeltaPrognozirovaniya[i] = Math.Pow(10,
                        yMinusDeltaPrognozirovaniya[i]);
                                YPlusDeltaPrognozirovaniya[i] = Math.Pow(10,
                        yPlusDeltaPrognozirovaniya[i]);
            }
        }

        public override void Normalize()
        {
            for(int i = 0; i < length; i++)
            {
                xn[i] = Math.Log10(x[i]);
                yn[i] = Math.Log10(y[i]);
            }
```

```
        }

    }
}
```

Normalizer.cs

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WindowsFormsApp1
{
    abstract class Normalizer
    {
        public const double t = 2.3534;
        private double S;
        private double ZxAverage;
        private double sum;
        protected int length;
        protected double[] x;
        protected double[] y;
        protected double[] Yoproximated;
        protected double[] xn;
        protected double[] yn;
        protected double[] delta;
        protected double[] yMinusDelta;
        protected double[] yPlusDelta;
        protected double[] deltaPrognozirovaniya;
```

```
protected double[] yMinusDeltaPrognozirovaniya;
protected double[] yPlusDeltaPrognozirovaniya;
protected double[] Y;
protected double[] YMinusDelta;
protected double[] YPlusDelta;
protected double[] YMinusDeltaPrognozirovaniya;
protected double[] YPlusDeltaPrognozirovaniya;
private double a;
private double b;
protected double rSquare;
protected double MMRE;
protected double PRED;

public abstract void Normalize();
public abstract void Reverse();

protected void countOproximation()
{
   Oproximator oproximator = new Oproximator(xn, yn);
   a = oproximator.getA();
   b = oproximator.getB();
   for (int i = 0; i < length; i++)
   {
      Yoproximated[i] = a * xn[i] + b;
   }
}

protected void countDeltaPrognozirovaniya()
{
   for (int i = 0; i < length; i++)
```

```
    {
                    deltaPrognozirovaniya[i] = t * S * Math.Sqrt( 1 +
                (double)1 / (double)length + (xn[i] - ZxAverage) * (xn[i]
                - ZxAverage) / sum);
    }
}

protected void countIntervalPrognozirovaniya()
{
    for (int i = 0; i < length; i++)
    {
                    yMinusDeltaPrognozirovaniya[i] =
                Yoproximated[i] - deltaPrognozirovaniya[i];
                    yPlusDeltaPrognozirovaniya[i] = Yoproximated[i]
                + deltaPrognozirovaniya[i];
    }
}

protected void countDeltaDoveria()
{
    double sumZx = 0;

    for(int i = 0; i < length; i++)
    {
        sumZx += xn[i];
    }

    ZxAverage = sumZx / length;

    //count sum (Zx - Zxcp)^2
```

```
    sum = 0;
    for (int i = 0; i < length; i++)
    {
        sum += (xn[i] - ZxAverage) * (xn[i] - ZxAverage);
    }

    for (int i = 0; i < length; i++)
    {
                        delta[i] = t * S * Math.Sqrt( (double)1 /
                (double)length + xn[i] - ZxAverage) * (xn[i] -
                ZxAverage) / sum );
    }

}

protected void countIntervalDoveria()
{
    for (int i = 0; i < length; i++)
    {
        yMinusDelta[i] = Yoproximated[i] - delta[i];
        yPlusDelta[i] = Yoproximated[i] + delta[i];
    }
}

protected void countS()
{
    double sum = 0;

    for(int i = 0; i < length; i++)
    {
```

```
                    sum += (yn[i] - Yoproximated[i]) * (yn[i] -
                Yoproximated[i]);
    }

    S = Math.Sqrt(sum/(length-2));
}

public Normalizer(double[] x, double[] y, bool check)
{
    this.x = x;
    this.y = y;
    this.length = x.Length;
    this.check = check;

    Yoproximated = new double[length];
    xn = new double[length];
    yn = new double[length];
    delta = new double[length];
    yMinusDelta = new double[length];
    yPlusDelta = new double[length];
    deltaPrognozirovaniya = new double[length];
    yMinusDeltaPrognozirovaniya = new double[length];
    yPlusDeltaPrognozirovaniya = new double[length];
    Y = new double[length];
    YMinusDelta = new double[length];
    YPlusDelta = new double[length];
    YMinusDeltaPrognozirovaniya = new double[length];
    YPlusDeltaPrognozirovaniya = new double[length];
```

ДОДАТОК Д

ІНСТРУКЦІЯ КОРИСТУВАЧА ПЗ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА KOTLIN

1. Програма запускається файлом «S_Kotlin.exe». Головне вікно програми має вигляд (рис. Д.1):

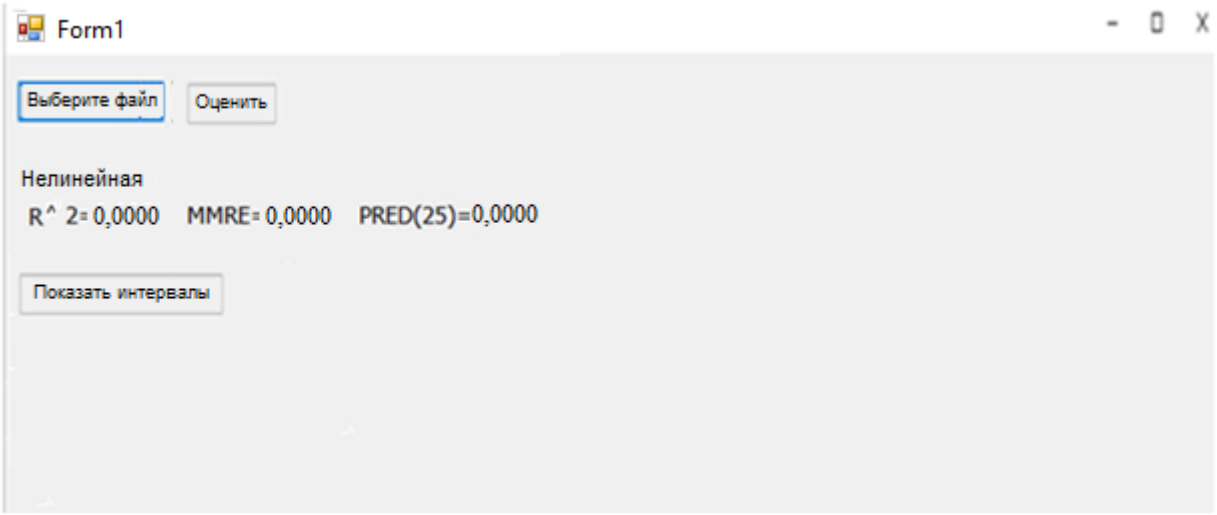

Рисунок Д.1 – Головна форма

2. Для введення емпіричного набору даних з файлу необхідно натиснути на кнопку «Виберите файл» (рис. Д.2).

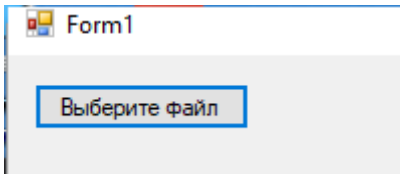

Рисунок Д.2 – Введення емпіричних даних з файлу

3. У вікні, що відкриється, необхідно обрати потрібний файл.

4. Після вибору файлу у вікні з’являться пораховані метрики якості (рис. Д.3).

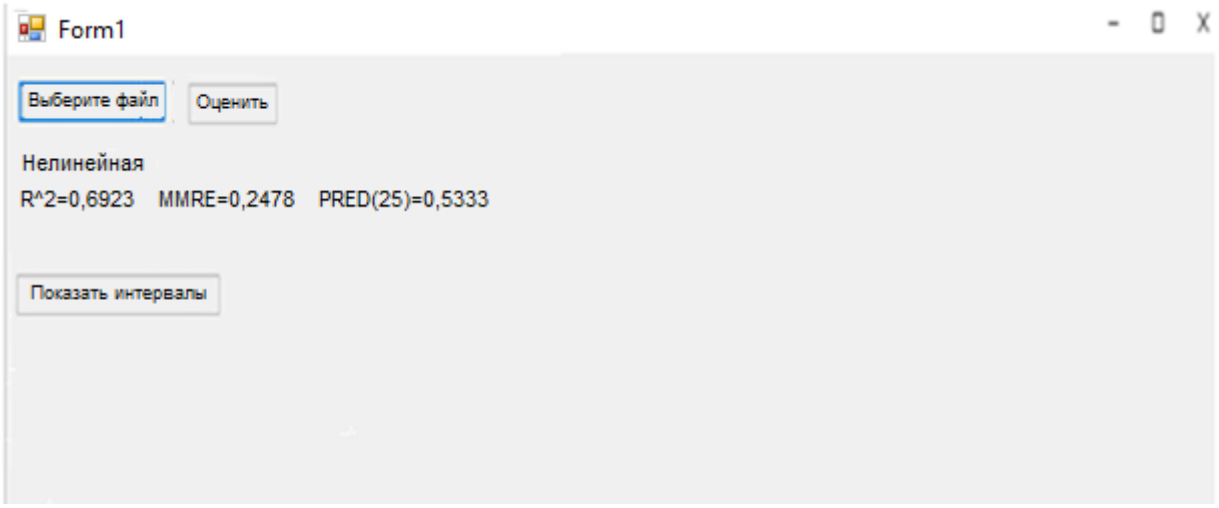

Рисунок Д.3 – Розраховані метрики

5. Для перегляду інтервалів довіри та прогнозування необхідно натиснути на кнопку «Показать интервалы» (рис. Д.4).

Рисунок Д.4 – Кнопка «Показать интервалы»

6. Для того щоб сховати пораховані інтервали необхідно натиснути на кнопку «Скрыть интервалы» (рис. Д.5)

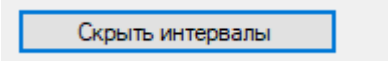

Рисунок Д.5 – Кнопка «Скрыть интервалы»

7. При неправильному форматі даних в файлі з’являється повідомлення про помилку (рис. Д.6)

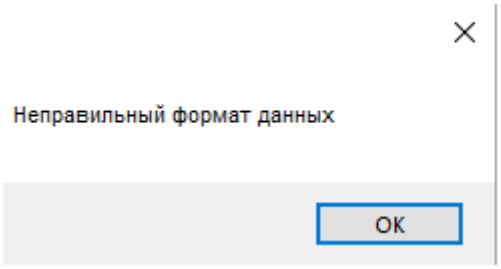

Рисунок Д.6 – Повідомлення про помилку

8. Для того щоб оцінити новий проект, необхідно натиснути на кнопку «Оценить» (рис. Д.7)

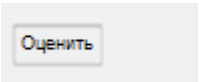

Рисунок Д.7 – Кнопка «Оценить»

9. З’явиться можливість ввести кількість функцій для оцінювання (рис. Д.8)

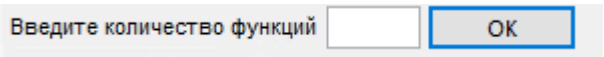

Рисунок Д.8 – Вікно для введення даних для оцінювання

Після вводу інформації та натиснення «ОК» отримаємо результат оцінювання.