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 регресійного рівняння для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури, та розробка програми для його реалізації

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

Ковальчук В.І.
(підпис, прізвище та ініціали)
Керівник Латанська Л.О.
(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

Ковальчуку Володимиру Івановичу
(прізвище, ім'я, по батькові)

1.Тема магістерської роботи: Удосконалення регресійного рівняння для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури, та розробка програми 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 на робочому місці програміста, 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

Ковальчук Володимир Іванович

«Удосконалення регресійного рівняння для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури, та розробка програми 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., 22 таблиці, 21 рисунок, 18 використаних джерел, 5 додатків.

Актуальність теми: Отримання ефективної системи оцінювання розміру програмного забезпечення в даний час є важливим завданням, що вимагає удосконалення існуючих методів. Адже саме ефективність оцінювання розміру програм може стати основою для успіху або невдачі проектів на ранніх етапах розробки.

Мета і завдання дослідження. Метою даної роботи є підвищення достовірності оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів. Реалізація мети включає задачі: аналіз існуючих методів та моделей оцінювання розміру програмного забезпечення; удосконалення регресійного рівняння; розробку програми для його реалізації.

Об'єкт дослідження: Процес оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням Java та мікросервісів.

Предмет дослідження: Рівняння регресії для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

Методи дослідження: Теорії ймовірності, математичної статистики та регресійного аналізу.

Наукова новизна одержаних результатів полягає в удосконаленні рівняння регресії для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів, що дозволило отримати більш точну оцінку розміру ПЗ.

Практичне значення одержаних результатів: ПЗ для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів, яке розроблене в рамках кваліфікаційної роботи, дозволяє автоматизувати та, відповідно, знизити трудомісткість розрахункових робіт.

Апробація результатів досліджень: Результати досліджень, викладених у роботі, були оприлюднені на VI міжнародній науково-технічній конференції «Комп'ютерне моделювання та оптимізація складних систем» (м. Дніпро, 04-06.11.2020).

Ключові слова: НЕЛІНІЙНЕ РЕГРЕСІЙНЕ РІВНЯННЯ, ОЦІНЮВАННЯ РОЗМІРУ, МІКРОСЕРВІСНА АРХІТЕКТУРА, ІНТЕРНЕТ-МАГАЗИНИ

ABSTRACT

Kovalchuk Volodymyr

«Improving the regression equation to estimate the size of software of online stores developed in Java on the basis of microservice architecture, and developing the software for its implementation»

Master's Degree in Master's Degree in Specialty 121 - Software Engineering. Admiral Makarov National University of Shipbuilding. Mykolaiv, 2020

The work contains: 104 pages, 22 tables, 21 figures, 18 references, 5 appendices.

Relevance of the topic: Obtaining an effective software size estimation system is currently an important task that requires improvement of existing methods. After all, it is the effectiveness of estimating the size of programs that can be the basis for the success or failure of projects in the early stages of development.

The purpose and objectives of the study. The purpose of this work is to increase the reliability of estimating the size of the software of online stores developed using Java language and microservices. Realization of the purpose includes tasks: the analysis of existing methods and models of the estimation of the size of the software; improvement of regression equation; development of a program for its implementation.

Object of research: The process of estimating the size of online store software developed using Java language and microservices.

Subject of research: Regression equation for estimating the size of software of online stores developed using Java language and microservices.

Research methods: The methods of probability theory, mathematical statistics and regression analysis were used in the research.

The scientific novelty of the obtained results is to improve the regression equation for estimating the size of software online stores developed using Java language and microservices, which allowed to obtain a more accurate estimate of the size of the software.

The practical value of the results: Software for estimating the size of software of online stores developed using Java language and microservices, which is developed as part of the qualification work, allows you to automate and, accordingly, reduce the complexity of computational work.

Approbation of study results: The results of the research presented in the paper were published at the 6th International Scientific and Technical Conference "Computer Modeling and Optimization of Complex Systems" (Dnipro, 04-06.11.2020).

Keywords: NONLINEAR REGRESSION EQUATION, SIZE ESTIMATION, MICROSERVICE ARCHITECTURE, ONLINE STORES

ЗМІСТ

ВСТУП

Актуальність теми. Розмір програмного забезпечення є одним з найважливіших внутрішніх показників програмного забезпечення. Інформація, яку отримуємо при оцінюванні розміру програмного забезпечення, корисна для прогнозування зусиль розробки програмного забезпечення.

Отримання ефективної системи оцінювання розміру програмного забезпечення в даний час є важливим завданням, що вимагає удосконалення існуючих методів. Адже саме ефективність оцінювання розміру програм може стати основою для успіху або невдачі проектів на ранніх етапах розробки.

Існує чимало методів та моделей, які використовуються для оцінювання розміру програмного забезпечення. Добре відомі та застосовуються на практиці методи побудови рівнянь, довірчих інтервалів та інтервалів прогнозування у випадку лінійної регресії. Проте існують припущення, які виправдовують використання моделей лінійної регресії, одним з яких є нормальність розподілу емпіричних даних. Але це припущення в задачах оцінювання розміру програмного забезпечення справедливе лише в окремих випадках. Тому необхідно використовувати нелінійні рівняння регресії, в тому числі і для оцінки розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

Мета і завдання дослідження. Метою даної роботи є підвищення достовірності оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів. Реалізація мети включає наступні завдання дослідження:

– аналіз існуючих методів та моделей оцінювання розміру програмного забезпечення та обгрунтування необхідності досліджень;

– удосконалення нелінійного регресійного рівняння для оцінювання розміру програмного забезпечення;

– розробку програми для оцінювання розміру програмного забезпечення з використанням удосконаленого рівняння.

Об’єкт дослідження: Процес оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

Предмет дослідження: Рівняння регресії для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

Методи дослідження. Поставлені завдання магістерської роботи будуть вирішуватися з використанням методів теорії ймовірностей, математичної статистики та регресійного аналізу.

Наукова новизна одержаних результатів полягає в удосконаленні рівняння регресії для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів, що дозволило отримати більш точну оцінку розміру ПЗ.

Практичне значення одержаних результатів: ПЗ для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів, яке розроблене в рамках кваліфікаційної роботи, дозволяє автоматизувати та, відповідно, знизити трудомісткість розрахункових робіт.

Особистий внесок здобувача. Кваліфікаційна робота є самостійно виконаною працею. Усі результати, викладені у роботі, отримані автором особисто. У тезах доповіді [1], написаних у співавторстві, здобувачу належить аналіз методів оцінювання розміру програмного забезпечення.

Апробація результатів досліджень. Результати досліджень, викладених у роботі, були оприлюднені на VI міжнародній науково-технічній конференції «Комп’ютерне моделювання та оптимізація складних систем» (м. Дніпро, 04-06.11.2020).

Публікації. Основні результати магістерської роботи викладено у 1 науковій праці – тезах конференції [1].

1 МІКРОСЕРВІСНА АРХІТЕКТУРА – ОСОБЛИВИЙ АРХІТЕКТУРНИЙ СТИЛЬ РОЗРОБКИ ПРОГРАМНИХ ДОДАТКІВ

1.1 Монолітний та мікросервісний архітектурні стилі

Існують різні підходи до побудови складних програмних систем. Визначальним фактором якісної системи є її архітектура. Провідними архітектурами в даний час являються монолітна та мікросервісна.

Монолітна архітектура полягає в тому, що програмна система розміщується на одному комп'ютері, запускається як єдине ціле та виконує всі функції системи. Монолітній архітектурі притаманні простота, узгодженість та міжмодульний рефакторінг [2].

Для мікросервісної архітектури характерним є те, що один додаток розробляється як набір невеликих, не тісно пов'язаних сервісів. Розробка кожного сервісу може вестися незалежно від інших [3].

Зазвичай монолітну архітектуру використовують невеликі команди на початку роботи над проектом. Однак при значному збільшенні коду та необхідності внесення незначних змін робота з додатком ускладнюється. Згідно зі статистикою, в даний час більшість комп'ютерних фірм досліджують або впроваджують мікросервіси [4].

1.2 Переваги та недоліки мікросервісів

Мікросервісна архітектура почала активно розвиватися та використовуватися разом з розвитком гнучкої методології.

Мікросервісна архітектура полягає в тому, що один додаток розробляється як набір невеликих, не тісно пов'язаних сервісів. Те, що сервіси невеликі та нетісно пов'язані, полегшує та прискорює їх розробку, налагодження, тестування та передачу замовнику. Сервіси працюють

незалежно. Тому внесення змін до одного з них мінімально впливає на роботу іншого. Це дуже важливо в умовах великих проектів та частих змін. Також помилка в одному сервісі може навіть і не вплинути на роботу інщих сервісів. Це полегшує тестування та налагодження програм та дає можливість надавати замовнику більш якісне програмне забезпечення та в більш стислі строки. Кожен з сервісів вирішує конкретну задачу. Сервіси можуть бути написані на різних мовах та використовувати різні підходи до збереження даних [3]. Також характерною є наявність у кожного сервісу власної бази даних. Для передачі інформації між сервісами використовуються HTTP, gRPC, AMQP та інші.

Основним недоліком використання мікросервісної архітектури є підвищення складності системи.

Але, незважаючи на такий вагомий недолік, мікросервісна архітектура активно розвивається. Досить часто цю архітектуру використовують для веб-проектів. Як приклад використання мікросервісної архітектури можна навести:Amazon, eBay, Twitter, Stripe, PayPal, Uber та Medium.

Однією з найпопулярніших мов для створення мікросервісів є Java. Програми на Java транслюються в байт-код. Байт-код являється незалежним від операційної системи та технічного забезпечення. Тому програми на Java-можуть виконуватися на довільному пристрої, для якого існує віртуальна машина Java.

Для створення мікросервісів на Java використовуються бібліотеки Spring framework, Spark framework, Restlet. На Java розроблені системи багатьох відомих компаній, наприклад Netflix, Amazon.

2 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ І МОДЕЛЕЙ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ, ТА ОБГРУНТУВАННЯ НЕОБХІДНОСТІ ПРОВЕДЕННЯ ДОСЛІДЖЕННЯ ЗА ДАНОЮ ТЕМОЮ

2.1 Метрики розміру програмного забезпечення

В даний час в програмної інженерії ще не сформувалася остаточно система метрик. Діють різні підходи до визначення їх набору і методів вимірювання. Розглянемо одну із класифікацій метрик програмного забезпечення. Згідно з однією із класифікацій, метрики програмного забезпечення можна поділити на групи: кількісні метрики, об'єктно-орієнтовані метрики, метрики складності потоку керування програми, гібридні метрики, метрики складності потоку керування даними, метрики складності потоку керування і даних програми [5]. Розглянемо більш детально кількісні та об'єктно-орієнтовані метрики.

До найбільш вживаних кількісних метрик відносяться [5]:

– Кількість рядків коду. Це найбільш поширена метрика вихідного коду, що відображає розмір програмного проекту, і дорівнює загальній кількості рядків коду SLOC.

Існує чимало різновидів цієї метрики, наприклад [5]:

- кількість пустих рядків;
- кількість рядків з коментарями;
- середня кількість рядків для функцій (методів);
- середня кількість рядків для модулів;
- середня кількість рядків для класів та інш.

Метрику SLOC можна використовувати на ранніх стадіях роботи над проектом для отримання прогнозного значення трудомісткості розробки.

– Стилістика програми. Використовується для визначення щільності коментарів.

– Метрики Холстеда. Обчислюються, враховуючи кількість рядків і синтаксичних елементів вихідного коду програми.

До об'єктно-орієнтованих метрик відносяться [5]:

– загальне число класів;

– загальне число методів;

– загальне число атрибутів та інш.

Кількість рядків коду використовується при підрахунку різних характеристик ПЗ та залежить від інших метрик програми.

2.2 Аналіз методів і моделей оцінювання розміру програмного забезпечення

У сучасних реаліях, коли розробка програмних продуктів ведеться в умовах значно обмеженого часу та фінансових ресурсів, максимально точне оцінювання необхідних трудових витрат на розробку є однією з основних проблем при управлінні проектами. Більшість підходів до оцінювання трудових витрат базуються на розмірі програмного забезпечення [6].

На сьогоднішній день методи та моделі оцінювання розміру програмного забезпечення можна поділити на дві основні групи: алгоритмічні та неалгоритмічні. Методи та моделі першої групи базуються на оцінюванні та аналізі кількісних характеристик програми. До них відносяться: метод PERT, регресійні моделі, нейронні мережі, генетичні алгоритми та інші.

Методи та моделі другої групи базуються на думці кваліфікованих експертів або на використанні попередньо визначених схем та принципів [6]. До них відносяться: експертне оцінювання, оцінка по аналогії та інші.

Неалгоритмічні методи оцінювання:

– Експертне оцінювання. Експертне оцінювання використовують, якщо не існує достатньої кількості емпіричних даних для алгоритмічного методу, або необхідні досить приблизні оцінки. Основними недоліками методів експертного оцінювання є суб'єктивність, упередженість та відсутність достатньої кваліфікації експертів.

– Метод оцінювання по аналогії. Це різновидність експертного оцінювання [7]. Для отримання оцінки використовуються дані по попередньо завершеним проектам.

Алгоритмічні методи використовують математичні моделі. Математичні моделі постійно удосконалюються та підвищується їх точність.

До алгоритмічних відносяться:

– Регресійні методи. Використовують лінійну та нелінійну, однофакторну та багатофакторну регресію.

– Нейронні мережі. Базуються на навчанні з використанням штучного інтелекту.

– Генетичні алгоритми. Базуються на використанні оператора схрещування, використовують механізми природного відбору та спадкоємства. Генетичні алгоритми можуть використовуватися самостійно, або спільно з нейронними мережами як комбіновані методи.

– Метод PERT. Дозволяє швидко та без зайвих витрат отримати досить оптимістичні оцінки.

Базується на наступній формулі:

$$E_i = (P_i + 4M_i + O_i) / 6,$$

де M_i – найбільш ймовірна оцінка;

O_i – оптимістична оцінка; жоден ризик не реалізувався;

P_i – песимістична оцінка; всі ризики реалізувалися;

E_i – середня оцінка по кожному проекту.

Більшість з наведених вище методів та моделей мають недоліки. З іншої сторони, з'являються нові та удосконалюються існуючі інформаційні

технології. Все це потребує удосконалення методик обчислення розміру програмного забезпечення, в тому числі і програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

2.3 Перевірка якості рівняння регресії для оцінювання розміру програмного забезпечення

Для регресійних рівнянь існує набір метрик, який дозволяє оцінити якість побудованого рівняння.

Найбільш часто при вимірюванні якості оцінки використовують середнє відхилення відносної похибки *MMRE* та якість передбачення *PRED(x)*. Обидві оцінки базуються на значенні відхилення відносної похибки *MRE* [8].

Відносна помилка *MRE* визначається за формулою (2.1)

$$MRE = \frac{|\hat{Y}_i - Y_i|}{Y_i}, \quad (21)$$

де $\hat{Y}_i$ – прогнозоване значення випадкової величини Y, Y_i – фактичне значення випадкової величини Y.

Середнє відхилення відносної помилки MMRE визначається як [7]:

$$MMRE = \frac{1}{n}\sum MRE, \quad (2.2)$$

де n – кількість проаналізованих проектів.

Якість (рівень) передбачення *PRED(x)* визначається як [8]:

$$PRED(x) = \frac{k}{n}, \quad (2.3)$$

де k – кількість даних, для яких $MRE \leq x$.

Модель повинна бути в межах 25% точності у 75% даних та *MMRE* менше 0,25 [8].

Для того щоб оцінити регресійні моделі, можна використовувати такий критерій як сума квадратів відхилень:

$$SSE = \sum_{i=1}^{n}\left(Y_i - \hat{Y}(X_i)\right)^2, \quad (2.4)$$

де n – це кількість емпіричних даних; Y_i – фактичне значення випадкової величини Y; $\hat{Y}(X_i)$ – прогнозоване значення випадкової величини Y.

Кращою є та модель, для якої значення *SSE* є меншим [9].

Також для оцінювання регресійних моделей можна використовувати значення коефіцієнта детермінації R^2:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}(X_i))^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}, \quad (2.5)$$

де n – це кількість емпіричних даних; Yi – фактичне значення випадкової величини Y; $\hat{Y}(X_i)$ – прогнозоване значення випадкової величини Y; $\bar{Y}$ – середнє значення випадкової величини Y, див. формулу (2.6).

$$\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i, \quad (2.6)$$

Кращою є та модель, для якої значення R^2 є більшим [9].

2.4 Обгрунтування необхідності проведення досліджень

Розмір програмного забезпечення є одним з найважливіших внутрішніх показників програмного забезпечення. Інформація, яку отримуємо при оцінюванні розміру програмного забезпечення, корисна для прогнозування зусиль розробки програмного забезпечення.

Отримання ефективної системи оцінювання розміру програмного забезпечення в даний час є важливим завданням, що вимагає удосконалення існуючих методів. Адже саме ефективність оцінювання розміру програм може стати основою для успіху або невдачі проектів на ранніх етапах розробки.

Існує чимало методів та моделей, які використовуються для оцінювання розміру програмного забезпечення. Добре відомі та застосовуються на практиці методи побудови рівнянь, довірчих інтервалів та інтервалів прогнозування у випадку лінійної регресії. Проте існують припущення, які виправдовують використання моделей лінійної регресії,

одним з яких є нормальність розподілу емпіричних даних. Але це припущення в задачах оцінювання розміру програмного забезпечення справедливе лише в окремих випадках. Тому необхідно використовувати нелінійні рівняння регресії, в тому числі і для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів [1, 10, 11].

Вище викладене підтверджує актуальність проведення дослідження щодо підвищення достовірності оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів, та розробки програми для її реалізації [1].

3 УДОСКОНАЛЕННЯ РЕГРЕСІЙНОГО РІВНЯННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ, ТА РОЗРОБКА ПРОГРАМИ ДЛЯ ЙОГО РЕАЛІЗАЦІЇ

3.1 Побудова удосконаленого регресійного рівняння оцінювання розміру програмного забезпечення

Для побудови нелінійної регресійної моделі будемо використовувати наступні емпіричні дані: кількість методів (*X*); кількість строк коду (*Y*). Емпіричні дані були зібрані з реально завершених проектів.

З допомогою підрахунку χ^2 для емпіричних даних, було встановлено, що емпіричні дані не підпорядковуються закону нормального розподілу ($\chi^2 > \chi^2_{крит}$). Це означає, що необхідно виконати нормалізацію даних [10]. В якості нормалізуючого перетворення будемо використовувати десятковий логарифм. В результаті отримуємо нормалізовані величини Z_Y та Zx.

Для побудови регресійного рівняння після вилучення викидів залишилося 30 даних. Викиди були вилучені з використанням квадрату відстані Махаланобіса, тестової статистики та критерію Фішера.

Розмір програмного забезпечення буде обчислюватися як функція від одного фактору.

$$Y = f(X) \tag{3.1}$$

Емпіричні та нормалізовані дані, які використовуються для виконання розрахунків, наведені в таблиці 3.1.

Таблиця 3.1 – Емпіричні дані для виконання розрахунків

№ проекту	*X*(кількість методів)	*Y* (кількість строк коду)	*Zx* (кількість методів норм.)	Z_Y (кількість строк коду норм.)
1	6294	40446	3,798927	4,606876
2	18954	69436	4,277701	4,841585
3	7198	43898	3,857212	4,642445
4	4943	38519	3,693991	4,585675
5	6487	36943	3,812044	4,567532
6	2898	28374	3,462098	4,452921
7	7845	37812	3,894593	4,577630
8	8577	39931	3,933335	4,601310
9	3290	24015	3,517196	4,380483
10	4661	37857	3,668479	4,578146
11	1498	18707	3,175512	4,272004
12	7684	49170	3,885587	4,691700
13	5211	26671	3,716921	4,426039
14	7345	43802	3,865992	4,641494
15	5687	35862	3,754883	4,554635
16	3649	31604	3,562174	4,499742
17	11398	79515	4,056829	4,900449
18	8405	47528	3,924538	4,676950
19	6487	39096	3,812044	4,592132
20	10204	50985	4,008770	4,707442
21	6017	40939	3,779380	4,612137
22	13298	69940	4,123786	4,844726
23	3156	29362	3,499137	4,467786
24	4284	38340	3,631849	4,583652
25	3754	30552	3,574494	4,485040
26	14572	50876	4,163519	4,706513
27	9333	59237	3,970021	4,772593
28	4008	28753	3,602928	4,458683
29	3679	34979	3,565730	4,543807
30	8109	40218	3,908967	4,604420

Емпіричний розподіл для кількості строк коду Y представлено на рис. 3.1.а, а її нормалізований розподіл представлений на рис. 3.1.б.

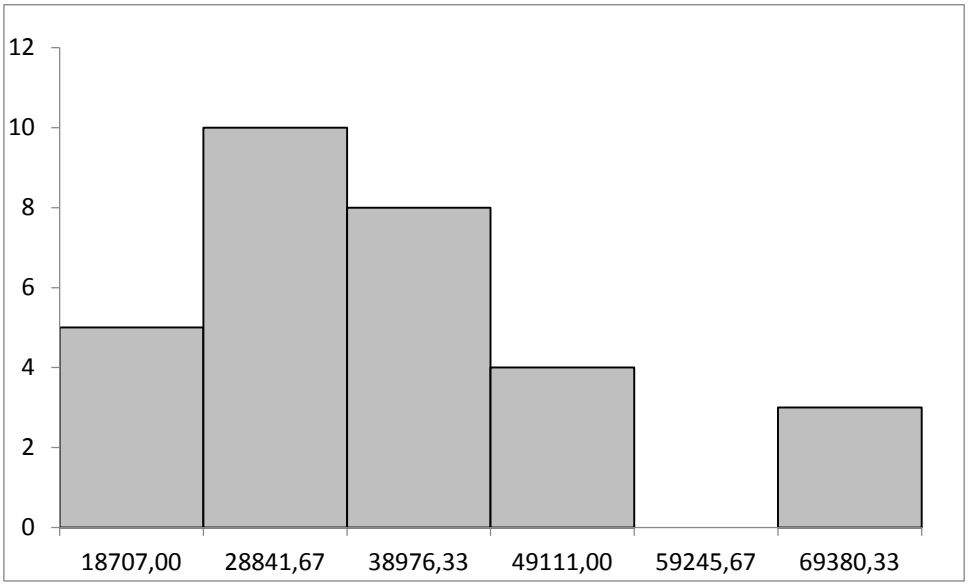

Рисунок 3.1.а – Емпіричний розподіл для кількості строк коду Y

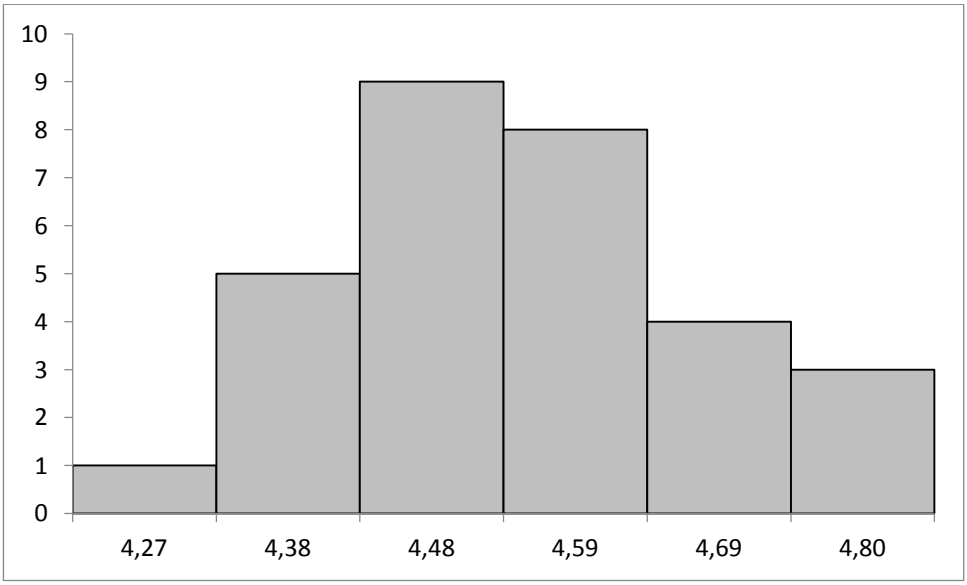

Рисунок 3.1.б – Нормалізований розподіл для кількості строк коду Y

Емпіричний розподіл для кількості методів X представлено на рис.3.2.а, а її нормалізований розподіл представлений на рис. 3.2.б.

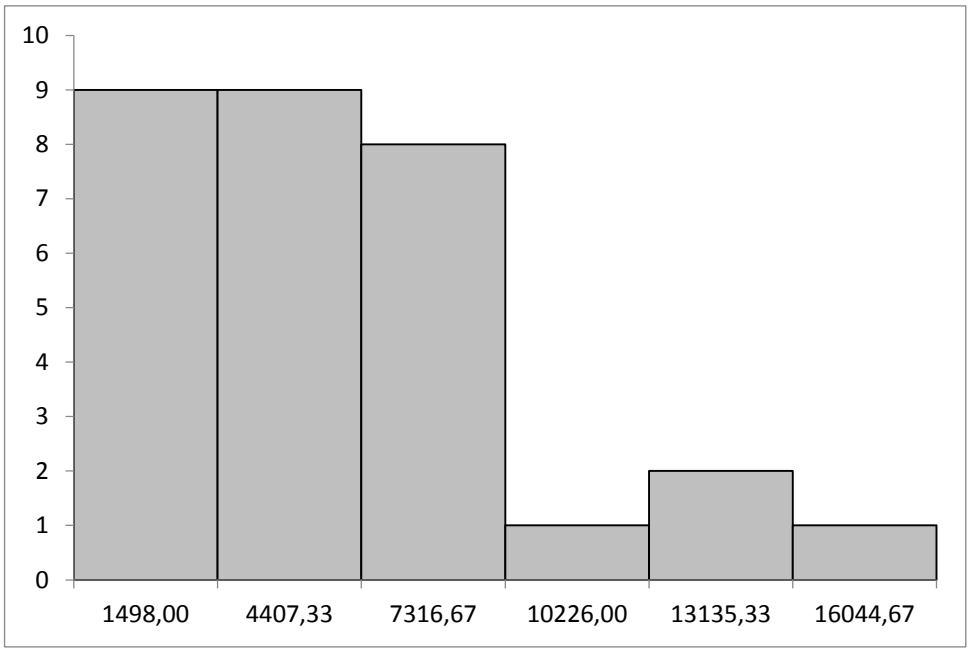

Рисунок 3.2.а – Емпіричний розподіл для кількості методів X

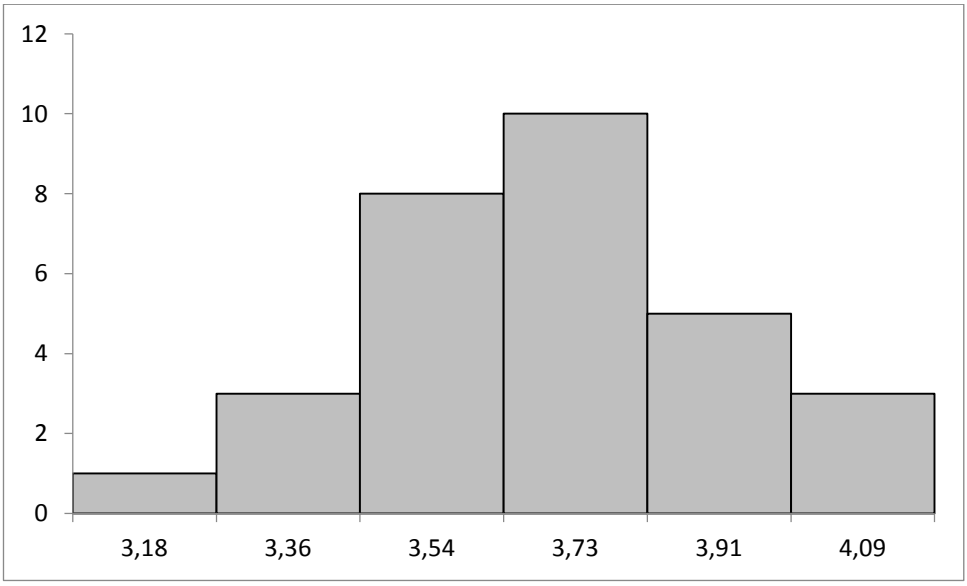

Рисунок 3.2.б – Нормалізований розподіл для кількості методів X

Лінійне рівняння регресії для нормалізованих даних з двома параметрами виглядає наступним чином [11]:

$$Z_{\hat{Y}} = b_0 + b_1 Z_X \quad (3.2)$$

Для знаходження коефіцієнтів використовували метод найменших квадратів: b_0=2,5908 ; b_1=0,5300.

Звідси

$$Z_{\hat{Y}} = 2{,}5908+0{,}5300Z_X.$$

Довірчий інтервал для лінійної регресії обчислюємо за формулою (3.3) [12]:

$$Z_y = Z_{\hat{Y}} \pm t_{\alpha/2,\vartheta}\ S_{Z_y}\sqrt{\frac{1}{n}+\frac{(x-\overline{x})^2}{\sum_n^1(x_i-\overline{x})^2}}, \tag{3.3}$$

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.

Інтервал передбачення – за формулою (3.4) :

$$Z_y = Z_{\hat{Y}} \pm t_{\alpha/2,\vartheta}\ S_{Z_y}\sqrt{1+\frac{1}{n}+\frac{(x-\overline{x})^2}{\sum_n^1(x_i-\overline{x})^2}}, \tag{3.4}$$

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.

Через зворотнє перетворення переходимо до нелінійної регресії, довірчого інтервалу та інтервалу передбачення для неї [13, 14].

Формула зворотнього перетворення для функції має вигляд

$$\hat{Y} = 10^{Z_{\hat{Y}}} \tag{3.5}$$

Нелінійне рівняння регресії оцінювання розміру програмного забезпечення:

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

Для наведених емпіричних даних нелінійне рівняння регресії, у випадку використання нормалізуючого перетворення десятковий логарифм, має вигляд:

$$\hat{Y} = 10^{2{,}5908} * X^{0{,}5300}.$$

Також в роботі отримали лінійне рівняння регресії у припущенні про нормальність вихідних даних. Обчисливши коефіцієнти за методом найменших квадратів, отримали: b_0=19857,6350 ; b_1=3,0999.

$$\hat{Y} = 19857{,}635 + 3{,}0999X.$$

Для отриманого лінійного рівняння регресії обчислюємо за формулами (3.3), (3.4) довірчий інтервал та інтервал передбачення.

Графіки прогнозного значення, довірчого інтервалу та інтервалу передбачення для лінійної регресії без нормалізації наведені на рисунку (рис.3.3).

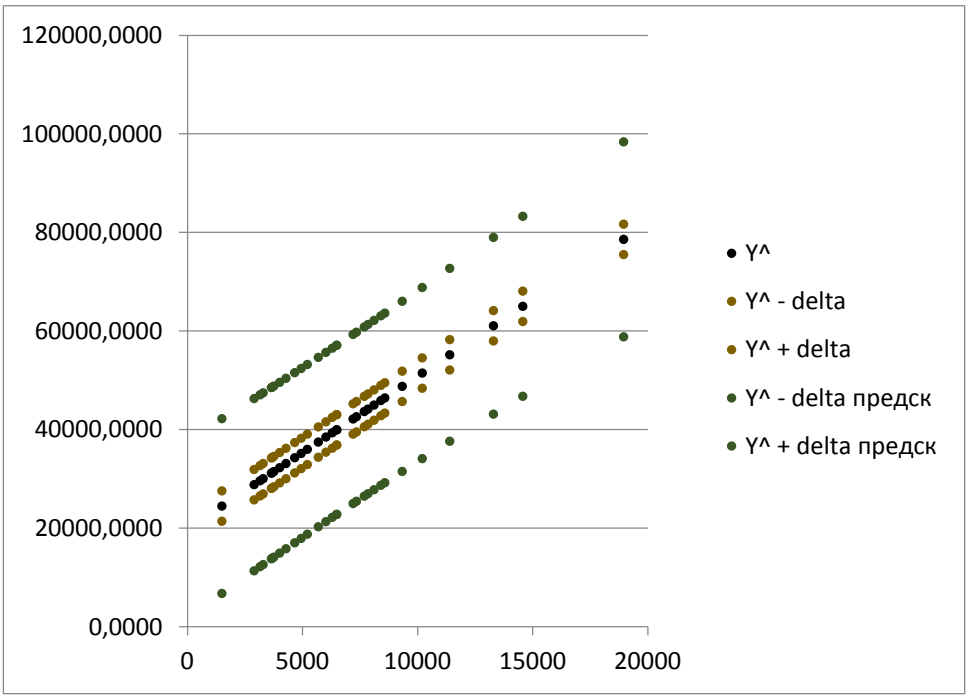

Рисунок 3.3 – Довірчий інтервал, інтервал передбачення та прогнозні значення для лінійної регресії без нормалізації

Графіки прогнозного значення, довірчого інтервалу та інтервалу передбачення для нелінійної регресії з нормалізацією представлені (рис.3.4).

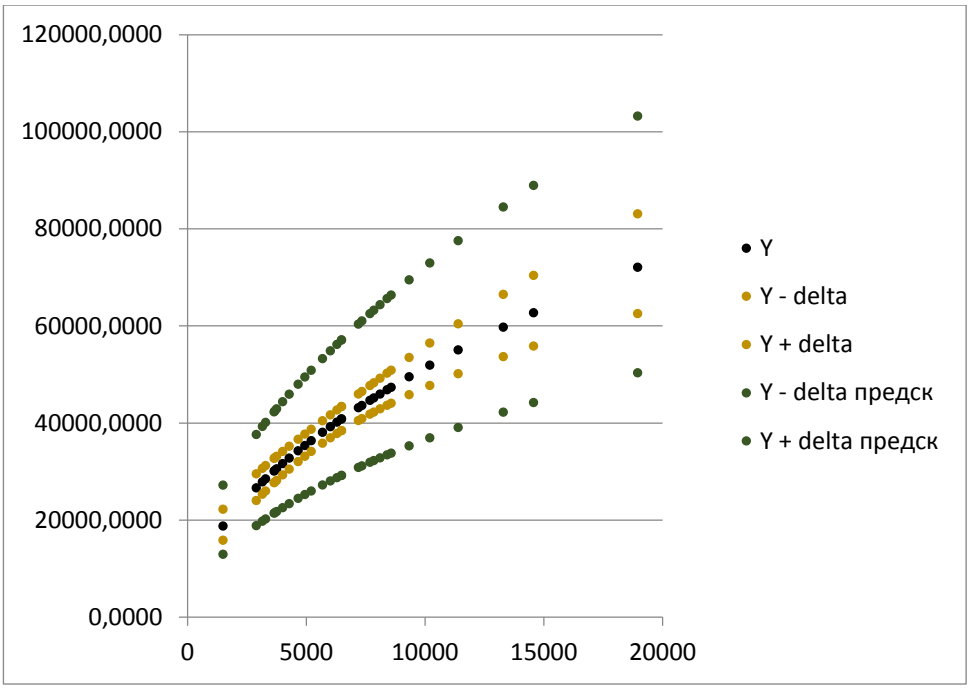

Рисунок 3.4 – Довірчий інтервал, інтервал передбачення та прогнозні значення для нелінійної регресії з нормалізацією

Зведені результати для довірчого інтервалу для лінійної та нелінійної регресії представлені в таблиці 3.2.

Зведені результати для інтервалу передбачення для лінійної та нелінійної регресії представлені в таблиці 3.3.

З таблиць видно, що довжини довірчих інтервалів та інтервалів прогнозування значно менші для переважної кількості нормалізованих даних, ніж відповідні довжини для ненормалізованих даних.

Таблиця 3.2 – Зведені результати для довірчого інтервалу для вихідних та нормалізованих даних

№	Довірчий ліва (ненорм)	Довірчий права (ненорм)	Довірчий інтервал (ненорм)	Довірчий ліва (норм)	Довірчий права (норм)	Довірчий інтервал (норм)
1	36238,5924	42497,6973	6259,1048	37841,3898	42695,1938	4853,8041
2	75531,4469	81693,2598	6161,8129	62550,8841	83097,5583	20546,6742
3	39088,6657	45252,1681	6163,5024	40514,8894	45973,8192	5458,9298
4	32099,3186	38261,1534	6161,8349	33153,8693	37721,1580	4567,2887
5	36885,3084	43047,5266	6162,2182	38439,4688	43398,2478	4958,7790
6	25760,1165	31921,9343	6161,8178	24040,0369	29538,6350	5498,5981
7	41095,0597	47256,9910	6161,9314	42253,2561	48293,3224	6040,0664
8	43364,1979	49526,0457	6161,8478	44080,3115	50882,4019	6802,0904
9	26975,2604	33137,0795	6161,8191	26002,7705	31239,6693	5236,8988
10	31225,1611	37386,9907	6161,8297	32051,7658	36662,8657	4611,0999
11	21420,3158	27582,1312	6161,8153	15859,9047	22246,3182	6386,4135
12	40595,9528	46757,9434	6161,9906	41832,4955	47718,6375	5886,1420
13	32930,0769	39091,9192	6161,8423	34159,5278	38717,9607	4558,4330
14	39544,8714	45707,3209	6162,4495	40921,3413	46503,0758	5581,7345
15	34405,5963	40567,4652	6161,8689	35849,9518	40473,7705	4623,8187
16	28088,1088	34249,9294	6161,8206	27703,6661	32723,7531	5020,0870
17	52108,9141	58270,7311	6161,8169	50161,6864	60441,6810	10279,9946
18	42831,0177	48992,8745	6161,8568	43662,7947	50277,7267	6614,9320
19	36885,3084	43047,5266	6162,2182	38439,4688	43398,2478	4958,7790
20	48407,6810	54569,5021	6161,8210	47739,6340	56479,4405	8739,8064
21	35428,5264	41590,4417	6161,9153	36954,1936	41683,4661	4729,2724
22	57998,6465	64160,4610	6161,8145	53681,5425	66505,1574	12823,6149
23	26559,8796	32721,6982	6161,8186	25344,7272	30668,8106	5324,0834
24	30056,5167	36218,3418	6161,8251	30505,5721	35226,8397	4721,2676
25	28413,5937	34575,4149	6161,8212	28184,7382	33147,0843	4962,3461
26	61947,8665	68109,6804	6161,8138	55860,6685	70418,8482	14558,1796
27	45707,7004	51869,5292	6161,8287	45840,7475	53511,3199	7670,5723
28	29200,9569	35362,7797	6161,8228	29318,9974	34154,5206	4835,5232
29	28181,1045	34342,9253	6161,8208	27841,8551	32845,1521	5003,2970
30	41913,4466	48075,3294	6161,8828	42927,7613	49231,7292	6303,9679

Таблиця 3.3 – Зведені результати для інтервалу прогнозування для вихідних та нормалізованих даних

№	Прогнозув. ліва (ненорм)	Прогнозув права (ненорм)	Прогнозув інтервал (ненорм)	Прогнозув. ліва (норм)	Прогнозув права (норм)	Прогнозув інтервал (норм)
1	22205,5543	56530,7353	34325,1810	28746,5524	56203,1039	27456,5516
2	58832,2245	98392,4823	39560,2578	50348,2737	103237,4172	52889,1435
3	25015,5822	59325,2516	34309,6694	30849,3964	60377,9787	29528,5822
4	17946,3170	52414,1550	34467,8380	25271,6660	49486,3434	24214,6774
5	22808,1806	57124,6545	34316,4739	29208,6840	57113,3431	27904,6591
6	11365,1025	46316,9483	34951,8457	18863,8892	37643,8744	18779,9852
7	27007,0129	61345,0378	34338,0249	32266,7268	63240,0718	30973,3450
8	29240,2769	63649,9666	34409,6897	33794,4804	66369,1850	32574,7046
9	12638,9230	47473,4169	34834,4939	20240,1840	40133,9213	19893,7373
10	17048,3003	51563,8515	34515,5511	24484,1967	47994,6146	23510,4180
11	6769,5332	42232,9138	35463,3806	12966,4066	27210,6604	14244,2537
12	26513,0006	60840,8957	34327,8951	31920,2464	62536,7882	30616,5417
13	18796,8928	53225,1034	34428,2106	25998,1612	50872,3384	24874,1772
14	25469,4845	59782,7077	34313,2232	31177,3815	61036,8205	29859,4390
15	20300,7195	54672,3420	34371,6226	27240,7590	53265,1355	26024,3764
16	13800,4445	48537,5937	34737,1491	21428,6862	42306,2768	20877,5906
17	37653,6701	72725,9750	35072,3049	39087,0836	77566,7143	38479,6307
18	28717,4040	63106,4882	34389,0841	33442,2029	65643,5841	32201,3812
19	22808,1806	57124,6545	34316,4739	29208,6840	57113,3431	27904,6591
20	34129,6119	68847,5712	34717,9594	36948,4158	72974,9235	36026,5077
21	21338,0889	55680,8791	34342,7902	28069,6219	54877,0796	26807,4577
22	43154,1128	79004,9946	35850,8818	42244,0988	84511,1987	42267,0999
23	12204,1311	47077,4467	34873,3156	19779,6064	39297,6797	19518,0733
24	15842,9644	50431,8941	34588,9298	23390,9473	45941,4868	22550,5395
25	14139,2440	48849,7645	34710,5205	21764,7709	42924,4993	21159,7283
26	46772,0911	83285,4558	36513,3647	44219,5555	88957,1116	44737,5561
27	31524,8425	66052,3871	34527,5445	35298,0923	69493,8097	34195,7174
28	14957,0825	49606,6541	34649,5716	22558,1127	44390,9610	21832,8483
29	13897,2871	48626,7427	34729,4556	21525,2159	42483,6606	20958,4448
30	27814,8700	62173,9060	34359,0360	32826,4725	64381,2070	31554,7345

Також в роботі для лінійної та нелінійної регресій були пораховані R^2, MMRE та PRED(25) [14, 15, 16] (табл. 3.4).

Таблиця 3.4 – R^2, MMRE та PRED(25) для лінійного та нелінійного рівнянь

Модель	Коефіцієнт детермінації, R^2	Середнє відхилення відносної помилки, MMRE	Рівень передбачення, PRED(25%)
Лінійна	0,7356	0,1151	0,8333
Нелінійна	0,7577	0,1029	0,9333

Як видно із наведеної таблиці 3.4, нелінійне рівняння, побудоване із використанням нормалізуючого перетворення десятичного логарифму має кращі показники PRED (0.25) на 0,1; MMRE на 0,013 та R^2 на 0,022 ніж лінійне, що побудоване у припущенні про нормальність вихідних даних.

Таким чином, кращий прогноз дає нелінійне рівняння регресії [17].

3.2 Постановка задачі на розробку програми для оцінювання розміру програмного забезпечення

Провівши дослідження існуючих методів та моделей для оцінювання розміру програмного забезпечення та побудувавши удосконалене нелінійне регресійне рівняння для оцінювання розміру програмного забезпечення обраного типу, необхідно перейти до розробки програмного забезпечення, яке автоматизує процес оцінювання розміру згідно розробленої моделі.

У розроблюваному програмному забезпеченні повинні бути реалізовані наступні функції:

1) Занесення даних.
2) Редагування даних.
3) Нормалізація даних.
4) Розрахунок верхньої та нижньої меж довірчого інтервалу.
5) Розрахунок верхньої та нижньої меж інтервалу прогнозування.

Розгорнута постановка задачі сформульована в технічному завданні, яке наведене в додатку А.

4 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПЗ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ

При розробці проекту програмного забезпечення були розроблені ескізний, технічний та робочий проекти з використанням структурної методології.

4.1 Ескізний проект програмного забезпечення для оцінювання розміру ПЗ

Проаналізувавши вимоги, що були висунені до програмного забезпечення в технічному завданні, був розроблений ескізний проект даного програмного забезпечення шляхом побудови контекстної діаграми, концептуальної моделі данних, моделі переходів станів та проекту інтерфейсу.

4.1.1 Контекстна діаграма програмного забезпечення

Контекстна діаграма показує обмін інформацією користувача програми та контекстного процесу. Контекстна діаграма ПЗ представлена на рисунку 4.1.

На контекстній діаграмі представлена зовнішня сутність «Користувач», контекстний процес, та потоки даних.

Рисунок 4.1 – Контекстна діаграма ПЗ для оцінювання розміру

Словник даних і потоки даних для контекстної діаграми представлено у таблиці 4.1.

Таблиця 4.1 – Словник даних і потоки даних контекстної діаграми

Найменування потоку	Опис потоку
Інформація про новий проект	Назва проекту, кількість методів
Інформація для корегування проекту	Назва проекту, кількість методів
Запит на оцінювання	Проект
Результат вводу	Назва проекту, кількість методів
Результат корегування	Назва проекту, кількість методів
Результат оцінювання	Кількість строк коду, ліва межа довірчого інтервалу, права межа довірчого інтервалу, ліва межа інтервалу передбачення, права межа інтервалу передбачення

4.1.2 Концептуальна модель даних ПЗ

При проектуванні структури даних звичайно будують три моделі – концептуальну, логічну та фізичну.

З аналізу контекстної діаграми системи можна побудувати концептуальну модель даних системи.

Найбільш розповсюдженим засобом зображення концептуальної моделі даних є діаграма «сутність-зв’язок».

Перший крок побудови моделі – визначення сутностей.

Можна виділити наступні сутності:

– «Проекти»;

– «Характеристики проектів»;

– «Результати оцінювання проектів».

На другому кроці необхідно визначити зв’язки між сутностями.

Розрізняють взаємозв'язки 1:1, 1:M та M:N.

Зв’язки між сутностями наступні:

«Проекти» – «Характеристики проектів». Одному проекту відповідають одні характеристики проектів, та одним характеристикам проектів відповідає один проект.

«Проекти» – «Результати оцінювання проектів». Одному проекту відповідають одні результати оцінювання, та одним результатам оцінювання відповідає один проект.

На третьому кроці необхідно визначити характеристики сутностей. У кожної сутності повинен бути хоча б один ключ. Ключ сутності використовується для визначення конкретної сутності. Ключі бувають прості та складові.

Визначимо атрибути сутностей:

– «Проекти»: назва.

– «Характеристики проектів»: кількість методів.

– «Результати оцінювання проектів»: розмір, інтервал передбачення лівий, інтервал передбачення правий, довірчий інтервал лівий, довірчий інтервал правий.

На рисунку 4.2 представлена концептуальна модель даних ПЗ.

Представлена модель даних задовольняє умовам третьої нормальної форми. Приведення моделі до необхідного рівня нормальної форми є основою побудови реляційної БД. У процесі нормалізації елементи даних групуються в таблиці, що представляють об'єкти і їхні взаємозв'язки[3].

Використання нормалізації при розробці інформаційної моделі забезпечує мінімальний обсяг записаної на будь-якому носії БД та її максимальну швидкодія, що відбивається на якості функціонування програмних систем [18].

Рисунок 4.2 – Концептуальна модель даних програмного забезпечення

4.1.3 Модель переходів станів програмного забезпечення

Для того щоб проаналізувати, як програмне забезпечення реагує на дії, створені користувачем, необхідно побудувати діаграму станів програмного забезпечення.

На діаграмі станів представляються:

– стани;

– послідовність подій.

При аналізі роботи системи визначили наступні її стани:

– Меню;

– Введення даних;

– Корегування даних;

– Оцінювання проекту;

– Нормування;

– Передбачення;

– Довірчі інтервали;

– Інтервали передбачення;

– Вихід.

Модель переходів станів ПЗ представлена на рисунку 4.3.

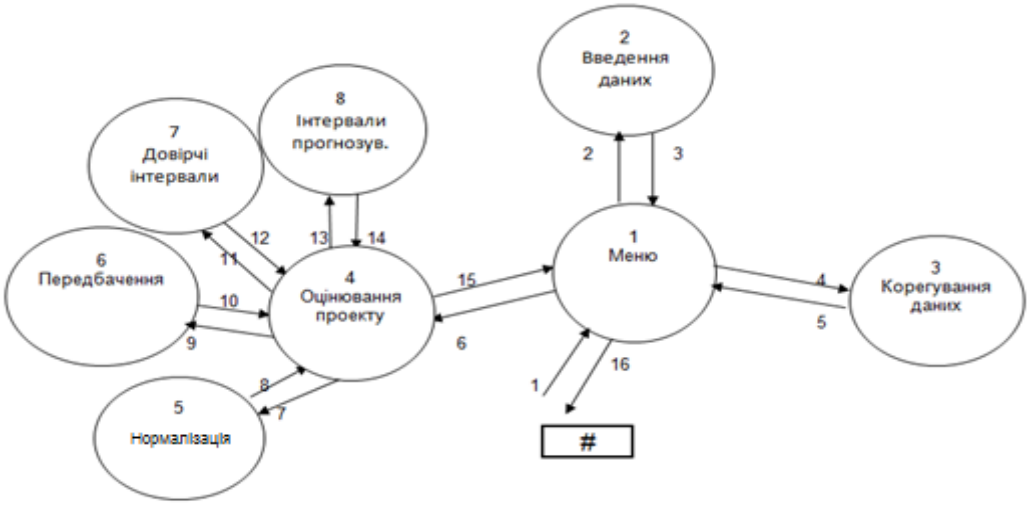

Рисунок 4.3 – Модель переходів станів програмного забезпечення

Можливі події на переходах описані в таблиці 4.2.

Таблиця 4.2 – Події на переходах

N	Події на переходах
1	початкова подія (завантаження програмного забезпечення)
2	вибір дії – введення даних
4	вибір дії – корегування даних
6	вибір дії – оцінювання проекту
7	вибір дії – нормалізація даних
9	вибір дії – передбачення розміру
11	вибір дії – формування довірчих інтервалів
13	вибір дії – формування інтервалів передбачення
3,5,8,10,12,14,15– повернення до вибору дії	
16	вихід з програми

4.1.4 Проектування інтерфейсу програмного забезпечення

Аналізуючи контекстну діаграму й діаграму станів, а також вимоги, пред'явлені в технічному завданні, можна визначити зовнішній інтерфейс.

З моделі переходів станів програмного забезпечення (рис.4.3) видно, що процес роботи програми починається із входу в програму. Тому головну форму програми буде доцільно створити у вигляді стандартного вікна з меню (рис. 4.4), на якому будуть розташовані елементи керування для доступу до функцій програми.

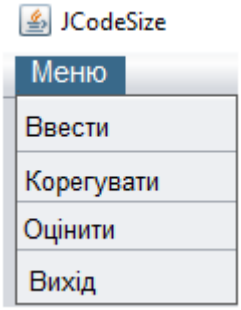

Рисунок 4.4 – Форма «Меню» програмного забезпечення

В меню можемо обрати одну із команд:

– Ввести інформацію про новий проект;

– Ввести інформацію для корегування існуючого проекту;

– Визвати команду оцінювання розміру програмного забезпечення;

– Вийти з програми.

При виборі команди «Ввести інформацію про новий проект» з’явиться форма, проект якої представлений на рисунку4.5:

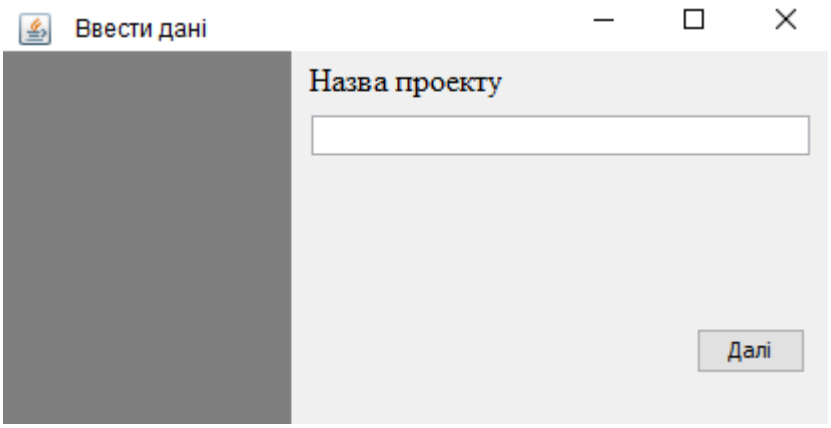

Рисунок 4.5 – Форма для вводу назви проекту

Далі переходимо на наступний крок. Проект вікна для вводу кількості методів представлений на рисунку (рис. 4.6):

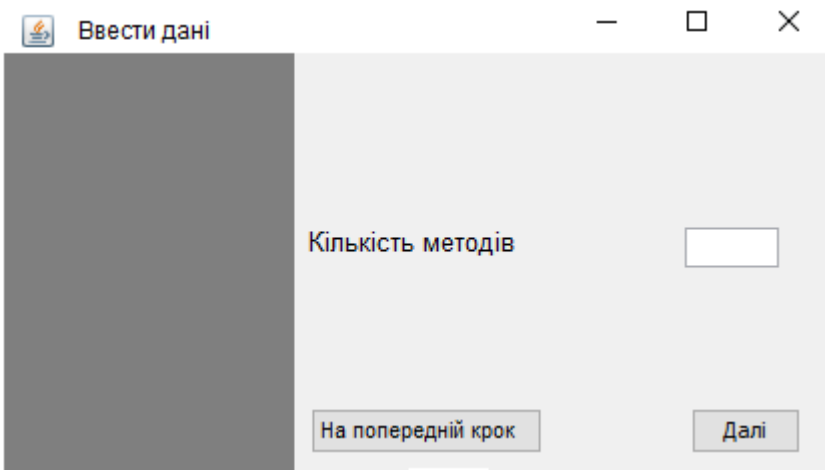

Рисунок 4.6 – Форма для вводу кількості методів

Далі переходимо до наступного вікна для перевірки введених даних: назви проекту та кількості методів (рис. 4.7):

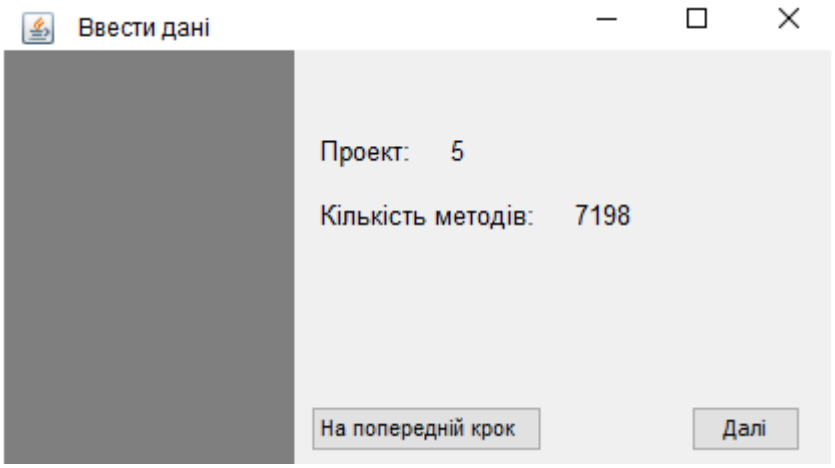

Рисунок 4.7 – Форма для перегляду результатів вводу

Після натискання на кнопку «Далі» отримаємо результат оцінювання (рис. 4.8.)

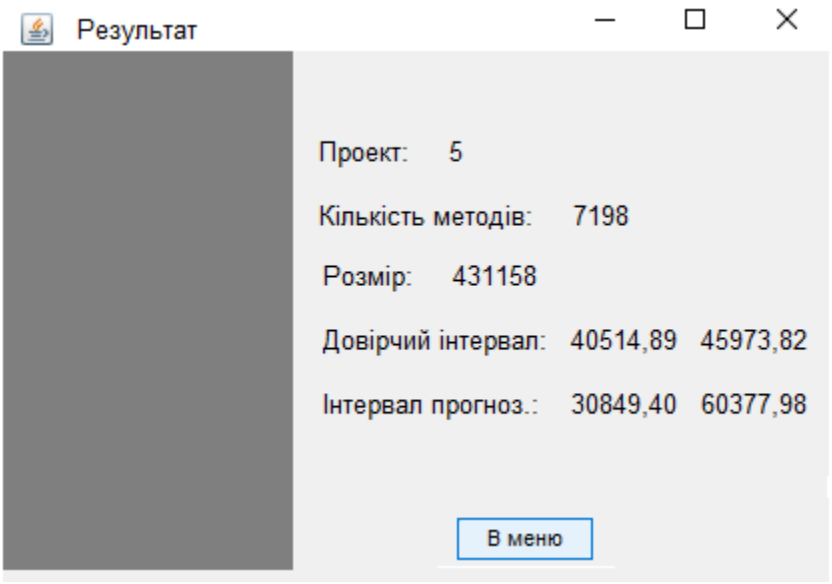

Рисунок 4.8 – Форма для перегляду результатів оцінювання.

4.2 Технічний проект програмного забезпечення для оцінювання розміру ПЗ

Під час технічного проектування була виконана декомпозиція контекстної діаграми до рівня специфікацій програмних модулів, розроблені специфікації програмних модулів та побудована логічна модель даних.

4.2.1 Декомпозиція контекстного процесу програмного забезпечення

Діаграми потоків даних необхідні для відображення всіх процесів, що відбуваються з даними. Для вирішення поставленої задачі, необхідно прослідкувати за потоками даних, що показані на рисунку 4.1, де зображена контекстна модель програми. Відповідно до технічного завдання, процес «Оцінювання розміру ПЗ інтернет-магазинів» може бути деталізований процесами «Ввести новий проект», «Корегувати існуючий проект», «Оцінити проект». Декомпозиція контекстного процесу представлена на рисунку 4.9.

Таблиця відповідності потоків даних контекстної діаграми та DFD першого рівня представлена в таблиці 4.3.

Таблиця 4.3 – Відповідність потоків даних контекстної діаграми та DFD першого рівня

Найменування потоків контекстної діаграми	Найменування потоків на DFD першого рівня
Інформація про новий проект	Інформація про новий проект
Інформація для корегування проекту	Інформація для корегування проекту
Запит на оцінювання	Запит на оцінювання
Результат вводу	Результат вводу
Результат корегування	Результат корегування
Результат оцінювання	Результат оцінювання

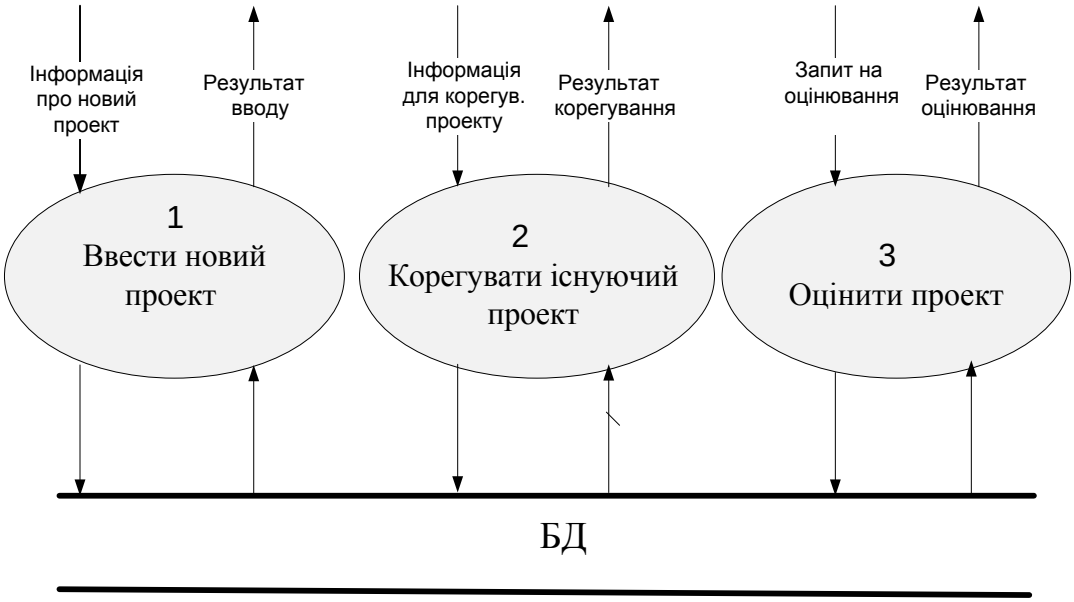

Рисунок 4.9 – Діаграма потоків даних першого рівня програмного забезпечення

Процес 1 «Ввести новий проект» призначений для введення інформації про новий проект, а саме: назви проекту та кількості методів.

Процес 2 «Корегувати існуючий проект» призначений для корегування інформації про існуючий проект, а саме: назви проекту та кількості методів.

Процес 3 «Оцінити проект» портебує деталізації, яка буде виконана на DFD другого рівня.

4.2.2 Побудова діаграми потоків даних другого рівня програмного забезпечення

На DFD першого рівня процес 3 «Оцінити проект» являється складним. Він деталізується до DFD другого рівня процесами: «Виконати нормалізацію», «Виконати передбачення розміру», «Порахувати інтервал передбачення», «Порахувати довірчий інтервал» (рис. 4.10):

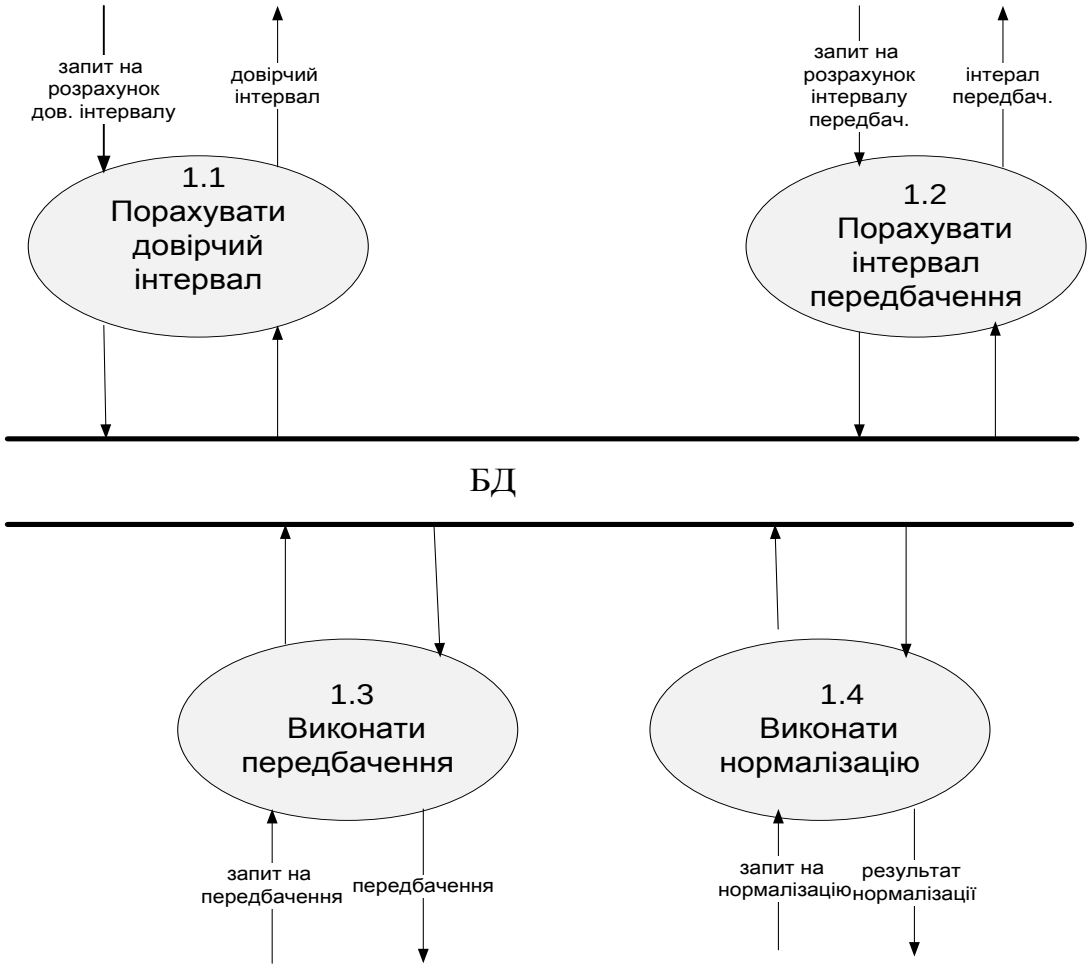

Рисунок 4.10 – Деталізація процесу «Оцінити проект» до DFD другого рівня

Процес «Виконати нормалізацію» призначений для нормалізації даних про проект.

Процес «Виконати передбачення розміру» призначений для обчислення розміру проекту згідно удосконаленого рівняння регресії.

Процес «Порахувати інтервал передбачення» призначений для обчислення лівої та правої границь інтервалу передбачення.

Процес «Порахувати довірчий інтервал» призначений для обчислення лівої та правої границь довірчого інтервалу.

Складемо таблицю відповідності потоків на DFD першого та другого рівнів (табл.4.4)

Таблиця 4.4 – Відповідність потоків на DFD першого та другого рівнів

Потоки даних DFD першого рівня	Потоки даних DFD другого рівня
запит на оцінювання	запит на розрахунок довірчого інтервалу
	запит на розрахунок інтервалу передбачення
	запит на передбачення
	запит на нормалізацію
результат оцінювання	довірчий інтервал
	інтервал передбачення
	передбачення
	результат нормалізації

.

4.2.3 Логічна модель даних програмного забезпечення

Базою для побудови логічної моделі даних є концептуальна модель. Логічна модель є більш детальним представленням даних. В ній сутності трансформуються в таблиці даних (часто одна сутність представляється декількома таблицями), атрибути сутностей стають полями таблиць, уточнюються відношення між таблицями, виділяються первинні ключі та типи даних [18].

На рисунку 4.11 зображена логічна модель даних програмного забезпечення.

Рисунок 4.11 – Логічна модель даних ПЗ

На рисунку 4.11 сутності представлені таблицями, для кожного атрибуту сутності вказані назва, тип даних та ключові поля.

Модель організації даних в логічній моделі буде реалізована в фізичній моделі відповідно до обраної СКБД.

4.3 Робочий проект програмного забезпечення для оцінювання розміру ПЗ

Під час робочого проектування був здійснений вибір засобів розробки, побудована фізична модель даних, виконана реалізація програмного продукту, його тестування та випробування.

4.3.1 Мова програмування та СКБД для розробки програмного забезпечення

Для реалізації проекту обрали мову Java, яка проста у використанні, поширена, об'єктно-орієнтована.Також Java не залежить від архітектури. Компілятор Java генерує об‘єктний байт-код, який не залежить від операційної платформи середовища виконання програми.

Мова Java без пербільшення являється самою затребуваною мовою програмування в наш час. Вона характеризується безпечністю та безвідмовністю. Мова Java гарно зарекомендувала себе в інтернеті.

Тому мова Java була обрана в якості мови програмування для даного проекту.

В якості СКБД обрали MySQL. MySQL широко використовується при роботі з базами даних, безкоштовна, підтримується багатьма мовами програмування, використовується в інтернеті, має високу швидкість обробки даних, активно розвивається.

4.3.2 Розробка фізичної моделі даних для ПЗ програмного забезпечення для оцінювання розміру програмних застосунків

На основі опису вхідної інформації й вимог, сформульованих до програмного забезпечення, концептуальної моделі даних та логічної моделі

можна створити фізичну модель бази даних. Фізична модель представлена таблицями 4.5 – 4.7.

Таблиця 4.5 – Структура таблиці «Характеристики проектів»(pdanni)

№	Назва	Тип	Опис	Ключ
1	id_pr_danni	INT	Код проекту	PK
2	class_danni	REAL	Кількість методів	

Таблиця 4.6 – Структура таблиці «Результати оцінювання» (resultat)

№	Назва	Тип	Опис	Ключ
1	id_pr_resultat	INT	Код проекту	PK
2	size_resultat	INT	Розмір	
3	n_dov_resultat	REAL	Інтервал передбачення лівий	
4	v_dov_resultat	REAL	Інтервал передбачення правий	
5	n_prog_resultat	REAL	Інтервал прогнозування лівий	
6	v_prog_resultat	REAL	Інтервал прогнозування правий	

Таблиця 4.7 – Структура таблиці «Проекти»(project)

№	Назва	Тип	Опис	Ключ
1	id_project	INT	Код проекту	PK
2	title_project	CHAR(36)	Назва проекту	

4.3.3 Кодування та тестування програмного забезпечення

Кодування програмного забезпечення виконується на основі результатів технічного проекту. Першим етапом кодування є встановлення програмного зв’язку з базою даних.

Програма написана на мові Java, яка реалізована для багатьох платформ. В якості СКБД обрано MySQL. Текст програми наведено в додатку Д. Інструкцію користувача наведено в додатку Г, Опис програми в додатку Б.

Тестування програмного забезпечення

Самими поширеними стратегіями тестування являються функціональне тестування (чорного ящика) та структурне тестування (білого ящика). До методів функціонального тестування відносяться: метод розбиття на класи еквівалентності, метод аналізу граничних значень, метод функціональних діаграм та інші. До методів структурного тестування відносяться: покриття операторів, покриття умов, покриття рішень, комбіноване покриття умов та інші. В першому випадку тестування проводиться згідно специфікації, у другому випадку – згідно структури програми. Програма була протестована методом функціонального тестування, а саме – розбиття на класи еквівалентності.

Розглянемо тестування функції «Корегувати існуючий проект». В табл. 4.8 наведені класи еквівалентності вхідних даних.

Таблиця 4.8 – Класи еквівалентності вхідних даних для функції «Корегувати існуючий проект».

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Найменування проекту	1 Строка тексту	2 Пуста строка
Кількість методів	3 Число більше 0	4 Число менше 0
		5 Число яке дорівнює 0
		6 Не число

У табл. 4.9 наведені тести з правильних класів еквівалентності. При тестуванні правильних класів еквівалентності необхідно обирати мінімальну кількість тестів. Може бути один тест на всі правильні класи еквівалентності.

Таблиця 4.9 – Тести з правильних класів еквівалентності

№ тесту	№ класу еквівалентності	Вхідні дані	Вихідні дані
1	1	В форму введена строка тексту	Стає можливим зберегти інформацію під найменуванням даного проекту
	3	В форму введено число більше 0	

У таблиці 4.10 наведені тести з неправильних класів еквівалентності.

За результатами тестування неправильних класів еквівалентності – програмне забезпечення працює вірно.

Тестування зводилося до послідовного вводу тестових наборів даних й аналізу отриманих результатів. На кожний неправильний клас еквівалентності потрібний свій тест.

Таблиця 4.10– Тести з неправильних класів еквівалентності

№ тесту	№ класу еквівалентності	Вхідні дані	Вихідні дані
1	2	В форму введена строка із пробілів	Повідомлення про невірне найменування проекту
2	4	В форму введено число менше 0	Повідомлення про невірну кількість класів
3	5	В форму введено 0	Повідомлення про невірну кількість класів
4	6	В форму ведена строка тексту	Повідомлення про невірний формат параметра

4.3.4 Випробування програмного забезпечення

Випробування ПЗ включає в себе випробування кожної функції (режиму) та випробування взаємодії функцій, тобто роботи ПЗ у цілому.

Отже, випробуємо кожний режим ПЗ, та перевіримо придатність ПЗ до використання по призначенню, його відповідність заданим вимогам.

Випробування програмного забезпечення виконаємо згідно з програмою і методикою випробувань, яка знаходиться у додатку В.

При проведенні випробувань розглядали функції:

1) запуск програми:

– запустили програмне забезпечення.

Внаслідок вказаних дій відкрилося головне вікно програми. Результат відповідає очікуваному.

2) функція введення даних:

– у головному меню вибрали пункт «Ввести».

Внаслідок вказаних дій відкрилося вікно для вводу даних про новий проект. Ввели назву проекту та натиснули кнопку «Далі».

Відкрилося вікно для вводу кількості методів. Ввели кількістьі методів та натиснули кнопку «Далі». Відкрилося вікно для перевірки введених даних. Перевірили введені дані та натиснули кнопку «Далі». Внаслідок вказаних дій інформація про новий проект записалася до бази даних і програма перейшла до функції «Оцінити». Результат роботи функції оцінювання з’явився в наступному вікні. Після натискання кнопки «В меню» запис з інформацією про оцінювання розміру програмного забезпечення, довірчий інтервал та інтервал прогнозування були записані до бази даних і відбувся перехід в меню.

Після виконання перелічених дій можна зробити висновок про те, що програмне забезпечення працює коректно, всі дії виконуються вірно у відповідності до вимог.

5 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Вступ

Дана робота присвячена підвищенню достовірності оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури та розробці програми для її реалізації. Програму можна використовувати в галузі розробки ПЗ, а також управління програмними проектами.

На сьогоднішній день оцінювання розміру розробки програмного забезпечення виконується за допомогою лінійних моделей та експертних оцінок, які не завжди дають можливість отримати достовірну оцінку. У даній роботі оцінювання розміру виконується з допомогою нелінійного рівняння регресії та нормалізуючого перетворення на основі десяткового логарифму. Цей підхід дозволяє використовувати для розрахунків негаусівські дані з нелінійною залежністю між факторами, які впливають на розмір програмного забезпечення. Крім того, отримання достовірної оцінки розміру ПЗ дозволить більш точно оцінити час та ресурси, які необхідні для розробки програмного продукту.

Ефективність програмних систем визначається ступенем їх позитивного впливу на підприємство, що управляється, або процес в результаті використання засобів обчислювальної техніки, програмного забезпечення, зміни інформаційних потоків та організаційної структури управління.

Створення програмного забезпечення вимагає одноразових витрат на розробку, придбання необхідних технічних засобів і поточних витрат на функціонування продукту. Економія від функціонування ПЗ визначається з

урахуванням витрат на його експлуатацію. Відношення цієї економії до витрат на створення програмного забезпечення характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються за діючими на момент розрахунку оптовими цінами, тарифами і ставками заробітної плати.

5.2 Розрахунок витрат на створення й експлуатацію програмного забезпечення

Витрати на розробку системи складаються з витрат:

- на заробітну платню розробника;
- на амортизацію ЕОМ, на якій виконується розробка;
- на експлуатацію цієї ЕОМ;
- на засоби розробки;
- на матеріали і комплектуючі.

Розробка програмного забезпечення виконується програмістом, місячна заробітна плата якого складає 8000 грн.

Вартість сучасного ПК становить 15000 грн.

При вартості кіловат-години електроенергії 1.29 грн, розраховується вартість розробки програми.

Витрати на допоміжні матеріали наведені в табл. 5.1.

Таблиця 5.1 – Витрати на допоміжні матеріали

№	Пункти витрат	Сума, грн
1	Флеш-накопичувач даних	350
2	Папір для принтера	110
3	Картридж та тонер для принтера	250
	Разом	710

Вартість ПЗ знаходиться за формулою:

$$С_{пр} = (З_{зп} + З_{сз} + З_{зг} + З_{е}) * Т + З_{м} \quad (5.1)$$

де Т – тривалість розробки, міс.

$З_{зп}$ – основна і додаткова заробітна плата обслуговуючого персоналу, грн.

$З_{сз}$ – відрахування на соціальні заходи (15% від основної і додаткової заробітної плати), грн.

$З_{зг}$ – загальногосподарські витрати (10% від основної заробітної плати), грн.

$З_{е}$ – витрати на електроенергію, грн.

$З_{м}$ – витрати на основні і допоміжні матеріали, грн.

$З_{е}$ при споживанні потужності 0.7 кВт, тривалості роботи на місяць, рівної 22*8 = 176 годин, вартості кіловат-години електроенергії 1.67 грн становить:

$$З_{е} = 176 * 1.67 * 0.7 = 158.92 \text{ грн}$$

Витрати на розробку ПЗ наведені в табл. 5.2.

Таблиця 5.2 – Витрати на розробку програмного забезпечення

№	Найменування витрат	Одиниця	Кількість
1	Тривалість розробки (Т)	Міс.	2
2	Основна і додаткова заробітна плата ($З_{зп}$)	Грн.	9000.00
3	Відрахування на соціальні заходи ($З_{сз}$)	Грн.	1400.00
4	Загальногосподарські витрати ($З_{зг}$)	Грн.	700.00
5	Витрати на допоміжні матеріали ($З_{м}$)	Грн.	710.00
6	Витрати на електроенергію ($З_{е}$)	Грн.	158.92

За формулою 5.1 розрахуємо вартість програми:

$$С_{пр} = (9000 + 1400 + 700 + 158,92) * 2 + 710 = 23227.84 \text{ грн}$$

Амортизаційні відрахування складають 25% балансової вартості на рік:

$$А_{об} = 10000 * 0.25 = 2500 \text{ грн}$$

В масштабах підприємства річні витрати на основні і допоміжні матеріали визначаються в розмірі 5% вартості основного устаткування:

$$В_м = 10000 * 0.05 = 500 \text{ грн}$$

Річний обсяг робіт ПК у годинах визначається в такий спосіб:

$$Ф_м = 264.5 * ТЗ \quad (5.2)$$

Тз – це середнє місячне навантаження устаткування (близько 8 годин), 264.5 – середня кількість робочих днів на рік.

Таким чином, річний обсяг роботи ПК складає:

$$Ф_м = 264.5 * 6 = 2116 \text{ годин}$$

Витрати на електроенергію $З_е$ при 2116 годинах роботи устаткування на рік становлять:

$$З_е = 2116 * 1.67 * 0.5 = 1767 \text{ грн}$$

Експлуатаційні витрати на ПК на рік складуть:

$$З_{зр} = 2500 + 500 + 1767 = 4767 \text{ грн}$$

Отже, у перший рік витрати на створення й експлуатацію ПЗ складуть:

$$З_р = 23227.84 + 4767 = 27994.84 \text{грн}$$

5.3 Економічна ефективність розробки і впровадження програмного забезпечення для оцінювання розміру програмного забезпечення

Основним показником економічної ефективності програмного забезпечення є зменшення трудових витрат та часу на оцінювання розміру програмного забезпечення, а також ефективне розподілення трудових ресурсів на основі отриманої оцінки розміру. Ефективний розподіл ресурсів дає змогу зменшити час на розробку програмного, і, відповідно, грошові витрати на роботу спеціалістів відділу забезпечення якості ПЗ. Також ця

оцінка допоможе знизити вірогідність провалу або додаткових робіт, що також зменшує можливі економічні втрати.

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження програмного продукту вивільняє 1 працівника (за експертною оцінкою фахівців підприємства).

Заробітна плата 1 працівника в рік складає:

$$З = 9000 * 12 * 1 = 108000 \text{ грн}$$

Річний економічний ефект розраховується за формулою:

$$E_{рік} = \Delta C_n – E_n * k \quad (5.3)$$

де ΔC_n – вивільнені кошти після впровадження продукту (108000 грн) мінус експлуатаційні витрати (27994.84 грн) = 80005.16 грн;

E_n – коефіцієнт ефективності (дорівнює коефіцієнту амортизації 0.25);

k – одноразові витрати на впровадження продукту (23227.84 грн).

$$E_{рік} = 80005.16 – 0.25 * 23227.84 = 82193.04 \text{ грн}$$

Термін окупності ПЗ розраховується за формулою:

$$T = k / \Delta C_n = 23227.84 / 80005.16 = 0.29 \text{ року}$$

Отже, термін окупності ПЗ становить приблизно 4 місяці.

5.4 Висновки

У цьому розділі були здійснені розрахунки витрат, економічної ефективності та терміну окупності ПЗ для оцінювання розміру програмного забезпечення. Виходячи з розрахунків, впровадження даного програмного забезпечення окупить себе протягом 4 місяців. Таким чином, його розробка є економічно обґрунтованою.

6 ОХОРОНА ПРАЦІ НА РОБОЧОМУ МІСЦІ ПРОГРАМІСТА

6.1 Система управління охороною праці

Наказом Мінпраці від 22.10. 2001 р. № 432 затверджена Концепція управління охороною праці, яка визначає, що управління охороною праці — це підготовка, прийняття та реалізація правових, організаційних, науково-технічних, санітарно-гігієнічних, соціально-економічних та лікувально-профілактичних заходів, спрямованих на збереження життя, здоров'я та працездатності людини в процесі трудової діяльності.

Відповідно до ст. 13 Закону "Про охорону праці" роботодавець повинен забезпечити функціонування системи управління охороною праці (СУОП). Він очолює роботу з управління охороною праці та несе безпосередню відповідальність за її функціонування в цілому на підприємстві.

СУОП, як підсистема загальної системи управління виробництвом, повинна передбачати такі функції:

– організацію і координацію робіт (обов'язки, відповідальність, повноваження керівників різного рівня, осіб, які виконують та перевіряють виконання роботи);

– облік, аналіз та оцінка ризиків;

– планування показників стану умов та безпеки праці;

– контроль планових показників та аудит всієї системи;

– коригування, запобігання та можливість адаптації до обставин, які змінюються;

– заохочення працівників за активну участь та ініціативу щодо здійснення заходів з підвищення рівня безпеки та поліпшення умов праці.

Завдяки цій системі повинні забезпечуватися вирішення таких основних завдань:

– професійний добір працівників, які виконують роботи підвищеної небезпеки з урахуванням стану їхнього здоров'я та психофізіологічних показників;

– навчання та пропаганда з охорони праці;

– безпека обладнання;

– безпека виробничих процесів;

– безпека будівель та споруд;

– забезпечення нормативних санітарно-гігієнічних умов праці;

– наявність засобів індивідуального захисту (ЗІЗ);

– оптимальні режими праці та відпочинку;

– лікувально-профілактичне обслуговування працюючих;

– санітарно-побутове обслуговування.

Щоб ця система діяла, необхідно запровадити відповідний нормативно-правовий акт, який би регулював усі питання, пов'язані з підготовкою, прийняттям та реалізацією управлінських рішень. При цьому треба пам'ятати, що СУОП є складовою загальної системи управління виробництвом (якістю продукції, що виробляється) і спрямована не тільки на створення оптимальних умов праці, але й на використання резервів виробництва, підвищення продуктивності праці та значне покращання якості продукції.

Враховуючи те, що Закон поширюється на всіх юридичних та фізичних осіб (далі – підприємства), які використовують найману працю, незалежно від того, працює на підприємстві 5 чи 1000 осіб, роботодавець зобов'язаний забезпечити виконання всіх завдань, передбачених цією статтею.

При створенні СУОП необхідно:

- визначити перелік законодавчих та інших нормативно-правових актів, що містять вимоги щодо охорони праці для даного виду економічної діяльності;
- виявити небезпечні та шкідливі виробничі фактори, види робіт, об'єкти, машини, механізми, устаткування підвищеної небезпеки, щоб визначити, які з них найсуттєвіше впливають на умови та безпеку праці;
- визначити основні завдання в галузі охорони праці та встановити пріоритетні напрями;
- розробити організаційну схему для виконання визначених завдань.

У зв'язку з тим, що роботодавець несе безпосередню відповідальність за стан умов та безпеки праці, для офіційного розподілу обов'язків, прав та відповідальності в галузі охорони праці між усіма учасниками виробничого процесу необхідно призначити посадових осіб, які повинні забезпечити вирішення конкретних питань охорони праці. Якщо у підпорядкуванні роботодавця є інженерно-технічні працівники, то у їхніх посадових інструкціях повинні бути відображені обов'язки, права і відповідальність за виконання покладених на них функцій з охорони праці.

Крім того, необхідно визначити порядок взаємодії всіх осіб, які беруть участь в управлінні виробництвом, а також порядок підготовки, прийняття та реалізації управлінських рішень (накази, розпорядження, приписи тощо).

При цьому треба врахувати, що очолює роботу з управління охороною праці та несе безпосередню відповідальність за її функціонування в цілому на підприємстві роботодавець (керівник підприємства), а у цехах, службах, на дільницях — керівники відповідних підрозділів і служб, відповідальні за стан умов та безпеку праці у підпорядкованих їм підрозділах.

Організаційно-методичне керівництво діяльністю структурних підрозділів та функціональних служб з охорони праці, підготовку

управлінських рішень та контроль за їх реалізацією повинна здійснювати служба охорони праці.

Порядок взаємодії осіб, які беруть участь в управлінні охороною праці, повинен забезпечувати виконання таких основних завдань:

1) Професійний добір для працівників, які виконують роботи підвищеної небезпеки, з урахуванням стану здоров'я та психофізіологічних показників (відповідно до Переліку робіт, де є потреба у професійному доборі).

Відповідальність за проведення цих заходів повинна покладатись на службу кадрів (або особу, яка виконує ці функції). Служба спільно із установою охорони здоров'я, службою охорони праці організує проведення медичних оглядів та психофізіологічної експертизи.

2) Навчання та пропаганду з питань охорони праці (відповідно до Типового положення про навчання з питань охорони праці).

Центром цих навчань та пропаганди на підприємстві повинен стати кабінет охорони праці. При чисельності працюючих менше ніж 400 осіб кабінет охорони праці поєднується з приміщенням для навчальних занять.

Відповідальність за своєчасність проведення навчання з питань охорони праці повинна покладатись на керівників структурних підрозділів, а організація навчання — на службу технічного навчання (або особу, яка виконує ці функції). Участь в організації та проведенні навчання беруть такі служби (або окремі фахівці): служба охорони праці та служби, на які покладено функції щодо правильної організації і своєчасного проведення ремонту та випробувань обладнання, систем енергопостачання, розробки та впровадження технологічної документації.

3) Безпека обладнання.

Безпека обладнання, що експлуатується, забезпечується шляхом приведення його у відповідність до вимог системи стандартів безпеки праці (ССБП) та вимог правил охорони праці, а також своєчасного

проведення планово-попереджувальних ремонтів, випробувань, удосконалення систем огороджувальних та запобіжних засобів.

Відповідальність за організацію цієї роботи повинна покладатись на службу (або фахівця), яка виконує функції щодо правильної організації і своєчасного проведення ремонту та випробувань обладнання. Участь в організації цієї роботи беруть служба (фахівці), на яку покладено функції щодо правильної організації і своєчасного проведення ремонту та випробувань систем енергопостачання, та служба охорони праці. Відповідальність за безпечну експлуатацію цього обладнання покладається на керівників структурних підрозділів.

6.2 Аналіз небезпечних і шкідливих виробничих факторів, характерних для приміщення програміста

Використання програмного продукту буде проводитися у приміщенні (рис. 6.1, табл.6.1).

Кількість програмістів в поміщенні – 1. В приміщенні знаходиться 1 стіл, 1 стілець, 1 комп’ютер.

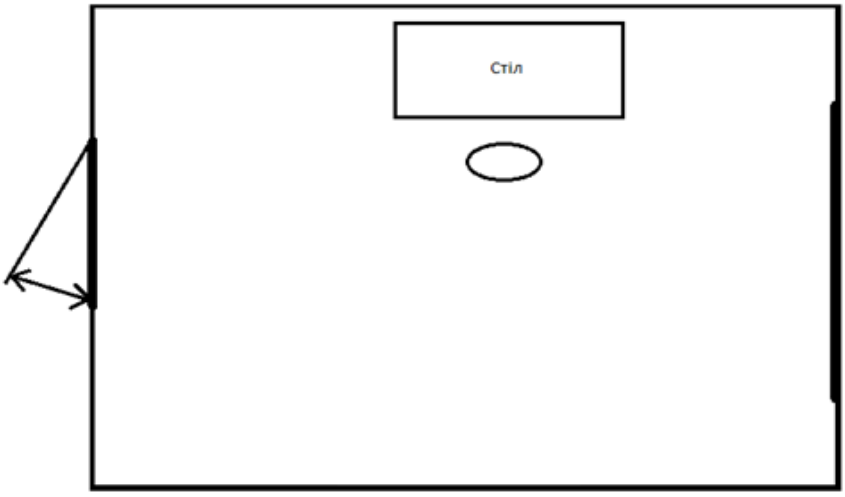

Рисунок 6.1 – План робочого приміщення

Таблиця 6.1 – Розміри приміщення

Наименование параметра	Значение
Длина, м	6.0
Ширина, м	4.0
Высота, м	3.0
Площадь, $м^2$	24.0
Объем, $м^3$	72.0

Виходячи з того, що в приміщенні працює одна людина, отримаємо наступні дані (табл. 6.2):

Таблиця 6.2 – Фактичні і нормативні значення площі і обсягу приміщення

Параметр приміщення	Нормативний	Фактичний
Площа, $м^2$	6 і більше	24.0
Об’єм, $м^3$	20 і більше	72.0

Отже, можна зробити висновок, що параметри приміщення задовольняють вимогою.

До небезпечних фізичних факторів відносяться: машини і механізми, що рухаються; різні підйомно-транспортні пристрої і переміщувані вантажі; незахищені рухливі елементи виробничого устаткування (приводні і передавальні механізми, різальні інструменти, пристосування, що обертаються і переміщуються й ін.); відлітаючі частки оброблюваного матеріалу та інструменту, електричний струм, підвищена температура поверхонь устаткування й оброблюваних матеріалів і т.д.

Шкідливими для здоров'я фізичними факторами є: підвищена чи знижена температура повітря робочої зони; висока вологість і швидкість руху повітря; підвищені рівні шуму, вібрації, ультразвуку і різних випромінювань

– теплових, іонізуючих, електромагнітних, інфрачервоних і ін. До шкідливих фізичних факторів відносяться також запиленість і загазованість повітря робочої зони; недостатня освітленість робочих місць, проходів і проїздів; підвищена яскравість світла і пульсація світлового потоку.

Хімічні небезпечні і шкідливі виробничі фактори за характером дії на організм людини підрозділяються на наступні підгрупи: загальнотоксичні, подразнюючі, сенсибілізуючі (ті, що викликають алергійні захворювання), канцерогенні (ті, що викликають розвиток пухлин), мутогені (ті, що діють на статеві клітини організму). У цю групу входять численні пари і гази: пари бензолу і толуолу, окис вуглецю, сірчистий ангідрид, окисли азоту, аерозолі свинцю та ін., токсичні пили, що утворюються, наприклад, при обробці різанням берилію, свинцюватих бронз і латуней і деяких пластмас зі шкідливими наповнювачами. До цієї групи відносяться агресивні рідини (кислоти, луги), що можуть заподіяти хімічні опіки шкіряного покриву при зіткненні з ними.

Між шкідливими і небезпечними виробничими факторами спостерігається визначений взаємозв'язок. У багатьох випадках наявність шкідливих факторів сприяє прояву травмонебезбечних факторів. Наприклад, надмірна вологість у виробничому приміщенні і наявність струмопровідного пилу (шкідливі фактори) підвищують небезпеку ураження людини електричним струмом (небезпечний фактор).

Рівні впливу на працюючих шкідливих виробничих факторів нормовані гранично-допустимими рівнями, значення яких зазначені у відповідних стандартах системи стандартів безпеки праці і санітарно-гігієнічних правил.

Гранично припустиме значення шкідливого виробничого фактора (за ДСТ 12.0.002-80) – це граничне значення величини шкідливого виробничого фактора, вплив якого при щоденній регламентованій тривалості протягом усього виробничого стажу не приводить до зниження працездатності і захворювання як у період трудової діяльності, так і до захворювання в

наступний період життя, а також несприятливо не впливає на здоров'я потомства.

Науково-технічний прогрес вніс серйозні зміни в умови виробничої діяльності працівників розумової праці. Їхня праця стала більш інтенсивною, напруженою, потребуючою значних витрат розумової, емоційної і фізичної енергії. Це зажадало комплексного рішення проблем ергономіки, гігієни й організації праці, регламентації режимів праці і відпочинку.

На даний час комп'ютерна техніка широко застосовується у всіх галузях діяльності людини. При роботі з комп'ютером людина піддається впливу ряда небезпечних і шкідливих виробничих факторів: електромагнітних полів (діапазон радіочастот: ВЧ, УВЧ і СВЧ), інфрачервоного та іонізуючого випромінювань, шуму і вібрації, статичної електрики та ін.

Робота з комп'ютером характеризується значною розумовою напругою і нервово-емоційним навантаженням операторів, високою напруженістю зорової роботи і досить великим навантаженням на м'язи рук при роботі з клавіатурою ЕОМ. Велике значення має раціональна конструкція і розташування елементів робочого місця, що важливо для підтримки оптимальної робочої пози людини-оператора.

У процесі роботи з комп'ютером необхідно дотримуватися правильного режиму праці і відпочинку. В противному випадку в персонала відзначається значна напруга зорового апарату з появою скарг на незадоволеність роботою, головні болі, дратівливість, порушення сну, утому і хворобливі відчуття в очах, у попереку, в області шиї і руках.

Розглянемо шкідливі виробничі фактори, які спостерігаються в приміщеннях, де здійснюється робота з електронно-обчислювальною технікою.

Освітлення

Правильно спроектоване і виконане виробниче освітлення поліпшує умови зорової роботи, знижує стомлюваність, сприяє підвищенню

продуктивності праці, благотворно впливає на виробниче середовище, створюючи позитивний психологічний вплив на працюючого, підвищує безпеку праці і знижує травматизм.

Недостатність освітлення приводить до напруги зору, послабляє увагу, приводить до передчасної стомленості. Надмірно яскраве освітлення викликає осліплення, роздратування і різь в очах. Неправильний напрямок світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Усі ці причини можуть привести до нещасливого випадку або профзахворювання, тому настільки важливий правильний розрахунок освітленості.

Існує три види освітлення – природне, штучне і сполучене (природне і штучне разом).

Природне освітлення – освітлення приміщень денним світлом, яке проникає через світлові прорізи в зовнішніх конструкціях приміщень. Природне освітлення характеризується тим, що міняється в широких межах в залежності від часу дня, пори року, характеру області і ряду інших факторів.

Штучне освітлення застосовується при роботі в темний час доби і вдень, коли не вдається забезпечити нормовані значення коефіцієнта природного освітлення (похмура погода, короткий світловий день). Освітлення, при якому недостатнє по нормах природне освітлення доповнюється штучним, називається сполученим освітленням.

Штучне освітлення підрозділяється на робоче, аварійне, евакуаційне, охоронне. Робоче освітлення, в свою чергу, може бути загальним чи комбінованим. Загальне – освітлення, при якому світильники розміщуються у верхній зоні приміщення чи рівномірно стосовно до розташування устаткування. Комбіноване – освітлення, при якому до загального додається місцеве освітлення.

Згідно до Сніп II-4-79 у приміщенях обчислювальних центрів необхідно застосувати систему комбінованого освітлення.

При виконанні робіт категорії високої зорової точності (найменший розмір об'єкта розрізнення 0,3...0,5 мм) величина коефіцієнта природного освітлення (КЕО) повинна бути не нижче 1,5%, а при зоровій роботі середньої точності (найменший розмір об'єкта розрізнення 0,5...1,0 мм) КЕО повинний бути не нижче 1,0%. В якості джерела штучного освітлення зазвичай використовуються люмінесцентні лампи типу ЛБ або ДРЛ, які попарно поєднуються у світильники, та повинні розташовуватися над робочими поверхнями рівномірно.

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери, наступні: при виконанні зорових робіт високої точності загальна освітленість повинна складати 300 лк, а комбінована – 750 лк; аналогічні вимоги при виконанні робіт середньої точності – 200 і 300 лк відповідно.

Крім того усе поле зору повинно бути освітлене досить рівномірно – це основна гігієнічна вимога. Іншими словами, ступінь освітлення приміщення і яскравість екрана комп'ютера повинні бути приблизно однаковими, тому що яскраве світло в районі периферійного зору значно збільшує напруженість очей і, як наслідок, приводить до їх швидкої стомлюваності.

Параметри мікроклімату

Параметри мікроклімату можуть мінятися в широких межах, в той час як необхідною умовою життєдіяльності людини є підтримка сталості температури тіла завдяки терморегуляції, тобто здатності організму регулювати віддачу тепла в навколишнє середовище. Принцип нормування мікроклімату – створення оптимальних умов для теплообміну тіла людини з навколишнім середовищем.

Обчислювальна техніка є джерелом істотного тепловиділення, яке може привести до підвищення температури і зниження відносної вологості в приміщенні.

У приміщеннях, де встановлені комп'ютери, повинні дотримуватися визначені параметри мікроклімату. У санітарних нормах СН-245-71

встановлені величини параметрів мікроклімату, що створюють комфортні умови. Ці норми встановлюються в залежності від пори року, характеру трудового процесу і характеру виробничого приміщення (табл. 6.3).

Таблиця 6.3 – Параметри мікроклімату для приміщень, де встановлені комп'ютери

Період року	Параметр мікроклімату	Величина
Холодний	Температура повітря в приміщенні	22…24 °С
	Відносна вологість	40…60 %
	Швидкість руху повітря	до 0,1 м/с
Теплий	Температура повітря в приміщенні	23…25 °С
	Відносна вологість	40…60 %
	Швидкість руху повітря	0,1…0,2 м/с

Шум і вібрація

Шумом називають усякий несприятливо діючий на людину звук. З фізичної точки зору звук являє собою механічні коливання пружного середовища.

Слуховий орган людини сприймає у вигляді чутного звуку коливання пружного середовища, які мають частоту приблизно від 20 до 20000 Гц, але найбільш важливий для слухового сприйняття інтервал від 45 до 10000 Гц.

Сприйняття людиною звуку залежить не тільки від його частоти, але і від інтенсивності і звукового тиску.

Несприятлива дія шуму на людину залежить не тільки від рівня звукового тиску, але і від частотного діапазону шуму, а також від рівномірності впливу протягом робочого часу.

У результаті несприятливого впливу шуму на працюючу людину відбувається зниження продуктивності праці, збільшується брак у роботі, створюються передумови до виникнення нещасливих випадків. Усе це

обумовлює велике оздоровче й економічне значення заходів щодо боротьби із шумом.

У таблиці 6.4 зазначені граничні рівні звуку в залежності від категорії ваги і напруженості праці, які є безпечними у відношенні збереження здоров'я і працездатності.

Таблиця 6.4 – Граничні рівні звуку, дБ, на робочих місцях.

Категорія напруженості праці	Категорія важкості праці			
	I Легка	II Середня	III Важка	IVДуже важка
I Мало напружена	80	80	75	75
II Помірковано напружена	70	70	65	65
III Напружена	60	60	–	–
IV Дуже напружена	50	50	–	–

Електромагнітне й іонізуюче випромінювання

Більшість учених вважає, що як короткочасний, так і тривалий вплив усіх видів випромінювання від екрана монітора не небезпечний для здоров'я персоналу, що обслуговує комп'ютери. Однак вичерпних даних щодо небезпеки впливу випромінювання від моніторів на працюючих з комп'ютерами не існує і дослідження в цьому напрямку продовжуються.

Максимальний рівень рентгенівського випромінювання на робочому місці оператора комп'ютера зазвичай не перевищує 10 мкбер/год, а інтенсивність ультрафіолетового й інфрачервоного випромінювань від екрана монітора знаходиться в межах 10…100 мВт/м^2.

6.3 Розробка заходів щодо забезпечення сприятливих умов праці при роботі з персональним комп'ютером на робочому місці програміста

Перед початком роботи на ПК користувач повинен: переконатися в цілості корпусів і блоків (обладнання) ПК; перевірити наявність заземлення, справність і цілісність кабелів живлення, місця їх підключення. Забороняється включати ПК і починати роботу при виявлені несправності.

Забороняється: замінювати знімні елементи або вузли та проводити перемонтаж при включеному ПК; з'єднувати і роз'єднувати вилки та розетки первинних мереж електроживлення, які перебувають під напругою; знімати кришки, що закривають доступ до струмопровідних частин мережі первинного електроживлення при включеному обладнанні; замінювати запобіжники під напругою; залишати ПК у ввімкненому стані без нагляду.

Після закінчення роботи необхідно знеструмити засоби обчислювальної техніки і периферійне устаткування. У разі безперервного виробничого процесу необхідно залишити включеними тільки необхідне обладнання.

Дисплей. Положення тіла звичайно відповідає напрямку погляду; дисплеї, розташовані занадто низько чи під неправильним кутом, є основними причинами сутулості. Відстань від дисплея до очей може варіюватися в залежності від характеру виконуваної роботи - 40-70 см. При роботі з текстом відстань від екрана до очей повинна лише небагато перевищувати відстань між книгою й очима і складати 40- 45 див. Необхідно давати відпочинок очам і час від часу їх просто закривати. Не можна допускати, щоб очі увесь час були сфокусовані на одній відстані, працюючи з текстом, рекомендується установлювати великий шрифт. Дуже важливо, щоб екран монітора не мерехтів. Частота регенерації зображення повинна бути не менш 72 Гц, тому що при більш низькій частоті помітне мерехтіння, хоча люди з особливо чуттєвим зором можуть помітити його і при більш

високій частоті. Їм доведеться підібрати графічну плату і монітор з частотою регенерації 85 Гц і вище.

Рекомендується встановлювати на екран монітора спеціальні скляні поляризаційні фільтри з заземленням. Необхідно уникати того, щоб термінал був звернений екраном убік вікна, оскільки інтенсивна освітленість області зору може затопити потоками світла очі і розмити зображення на сітківці. Якщо приходиться сидіти поруч з вікном, то треба розташуватися під прямим кутом до нього, причому екран дисплея повинний бути перпендикулярний шибці – цим виключаються відблиски на екрані.

Для зниження впливу низькочастотного електромагнітного випромінювання монітора на електронно-променеві пристрої рекомендується вибирати монітор, що задовольняє стандарту MPR II, чи TCO 99. У противному випадку необхідно установити на екран захисний фільтр, найбільш сучасні моделі, затримують до 99 % електромагнітного випромінювання.

Форма спинки крісла повинна повторювати форму спини працюючого. Крісло необхідно установити на такій висоті, щоб не почувався тиск на куприк (крісло розташоване занадто низько) на стегна (крісло розташоване занадто високо). Фахівці з ергономіки, вважали, що кут між стегнами і хребтом повинний складати 90°, однак недавно проведені дослідження показали, що більшість людей переважно сидять трохи відкинувшись.

Клавіатуру необхідно установити так, щоб не треба було до неї тягтися і нахилятися вперед. При зміні положення тіла (наприклад, з вертикального на похиле) обов'язково треба перемінити положення клавіатури і дисплея. Може виявитися корисною регульована підставка клавіатури, завдяки якій можна без усякої напруги працювати з маніпулятором "миша". Можна поставити клавіатуру і на коліна. Руки при роботі повинні бути прямими в зап'ястях і зігнуті в ліктях приблизно під прямим кутом. Пальці також повинні бути злегка зігнуті. Удари по клавішах не повинні бути занадто сильними.

Зручна висота столу особливо важлива в тому випадку, коли на ньому розташовується клавіатура. Якщо в клавіатури немає підставки, а висоту столу не можна змінити (і він занадто високий), то треба вище підняти сидіння крісла, а під ноги підставити лавочку: чи що-небудь інше. Якщо стіл занадто низький, необхідно підкласти що-небудь під його ніжки.

Якщо при роботі часто приходиться дивитися на документи, треба установити підставку з оригіналом документа в одній площині і на одній, висоті з екраном. Якщо треба частіше дивитися на оригінал, чим на екран, необхідно повернути крісло й екран таким чином, щоб оригінал розміщався прямо перед
очима, а комп'ютер ледве збоку.

Щогодини необхідно робити перерву в роботі. У цей час рекомендується робити вправи для зап'ясть. Нижче описаний можливий комплекс вправ.

– Покласти руку на край столу долонею вниз. Взявшись, за пальці, іншою рукою відвести кисть назад і утримувати протягом 5 секунд. Повторити для іншої руки.

– Злегка упертися рукою в стіл і на 5 секунд напружити пальці в зап'ястя. Повторити іншою рукою.

– Сильно стиснути пальці , а потім розпрямити їх. Виконання приведених рекомендацій дозволяє скоротити вплив шкідливих, виробничих: факторів на організм людини.

Вимоги до освітлення. У приміщеннях, де експлуатуються комп'ютери, штучне освітлення повинне бути загальним і рівномірним. Однак якщо співробітники переважно працюють з документами, то допускається застосувати комбіноване освітлення: крім загального установлюються світильники місцевого освітлення, які не повинні створювати відблисків на поверхні екрана і збільшувати його освітленість більш 300 лк. Освітленість поверхні столу в зоні розміщення робочого документа повинна становити 300-500 лк.

Джерела освітлення варто встановлювати таким чином, щоб вони не осліплювали, при цьому яскравість світних поверхонь (вікна, світильники та ін.), які розташовуються в полі зору, повинна бути не більш 200 кд/м^2.

В якості джерела світла при штучному освітленні повинні застосовуватися переважно люмінесцентні лампи типу ЛБ. При пристрої відбитого освітлення допускається застосування металогалогених ламп потужністю до 250 Вт, а у світильниках місцевого освітлення – ламп накалювання.

Для внутрішньої обробки приміщень повинні використовуватися дифузно-відбиваючі матеріали, з коефіцієнтом відбиття від стелі – 0,7-0,8; для стін – 0,5-0,6; для підлоги – 0,3-0,5. Полімерні матеріали для внутрішньої обробки повинні бути дозволені для застосування органами й установами Держсанепіднагляду.

Поверхня підлоги в приміщеннях повинна бути рівною, без вибоїн, неслизькою, зручною для очистки і вологого прибирання, мати антистатичні властивості.

Вимоги до кондиціонування. У виробничих приміщеннях, у яких установлені комп'ютери, мікроклімат повинен відповідати наступним санітарним нормам:

– температура повітря в теплий період року – не більш 23^0-25^0 С, у холодний – 22^0-24^0 С;

– відносна вологість повітря – 40-60%;

– швидкість руху повітря – 0,1м/с.

Для підвищення вологості повітря в приміщеннях варто застосовувати зволожувачі повітря, щодня заправляти їх дистильованою або кип'яченою водою.

Вимоги до розміщення, оснащення й організації робочих місць. Відстань між робочими столами з моніторами (у напрямку тилу поверхні одного монітора та екрана іншого) повинне бути не менш 2 м, а між бічними поверхнями моніторів – не менш 1,2 м.

Віконні отвори повинні бути обладнані регульованими жалюзями, завісами, зовнішніми навісами та ін.

Бажано, щоб висоту робочої поверхні столу можна було регулювати в межах 680-800 мм, а при відсутності такої можливості вона повинна дорівнювати 725 мм. Модульними розмірами робочої поверхні комп'ютерного столу, на базі яких розраховують конструктивні розміри, варто вважати: ширину 800, 1000, 1200 і 1400 мм; глибину 800 і 1000 мм.

Екран монітора повинен знаходитись від очей користувача на оптимальній відстані 600-700 мм, але не ближче 500 мм з урахуванням розмірів алфавітно-цифрових знаків і символів.

Вимоги до безпеки під час експлуатації й обслуговування ЕОМ. У приміщенні з комп'ютерами повинно проводитися щоденне вологе прибирання. Приміщення повинні оснащатися аптечкою першої допомоги та вуглекислотними вогнегасниками.

Режими праці і відпочинку. Режими праці і відпочинку при роботі на комп'ютерах залежать від виду і категорії трудової діяльності.

При восьмигодинній робочій зміні і роботі на комп'ютері регламентовані перерви варто встановлювати:

– для I категорії робіт – через 2 години від початку робочої зміни і через 2 години після обідньої перерви тривалістю по 15 хв.;

– для II категорії робіт – через 2 години від початку робочої зміни і через 1,5 - 2 години після обідньої перерви тривалістю по 15 хв. або через кожну годину роботи тривалістю по 10 хв.

Під час регламентованих перерв із метою зниження нервово-емоційної напруги, зменшення стомлення очей, усунення гіподинаміки і гіпокінезії доцільно виконувати комплекси вправ, викладених у санітарних нормах.

Після проведення дослідження, щодо відповідності всіх норм та правил, можемо зробити висновок, що робота в приміщенні проходить в умовах, які відповідають усім санітарним нормам та техниці безпеки.

6.4 Висновки

В даному розділі проведено аналіз основних аспектів охорони праці. Було показано, що проводяться всі необхідні заходи для забезпечення гідних умов праці, відповідних нормативно-правових актів охорони праці: періодично провітрюється приміщення, використовуються природне і комбіноване штучне освітлення, робляться перерви для відпочинку очей, ПК і периферійних пристроїв мають захист від струму короткого замикання, використовуються справні штепсельні з'єднання та розетки; для гасіння пожежі передбачена система стійок з пожежними кранами, підключена до водопровідної мережі, і два вогнегасники. Параметри приміщення відповідають нормативним вимогам. Рівень шуму не перевищує нормативних обмежень. Пожежна безпека, електробезпека і мікроклімат відповідають нормам безпеки.

7 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

7.1 Вступ

Під навколишнім середовищем розуміють цілісну систему взаємопов'язаних природних і антропогенних об'єктів і явищ, під впливом і при безпосередньому використанні яких відбувається праця, побутова діяльність, відпочинок людей. Поняття «навколишнє середовище» включає соціальні, природні і штучно створені фізичні, хімічні та біологічні фактори, тобто все те, що впливає на життя і діяльність людини. Складовою частиною навколишнього середовища є природне середовище. Перед сучасним суспільством стоїть завдання не тільки зберегти природу, а й запобігти негативним наслідкам господарської діяльності людини в майбутньому.

Економічна діяльність у всіх її проявах здійснює забруднення навколишнього середовища. У процесі цієї діяльності забруднюються і стають дефіцитними ресурси повітря, води, територій, що здавалися нескінченними. Нині рівень забруднення досяг загрозливих розмірів, набувши по суті кризового характеру.

Цей процес посилюється розвитком виробничих сил і збільшенням маси речовин, що залучаються в господарський обіг. Через це в навколишнє середовище надходить все більше й більше різноманітних речовин, які йому чужі, а часом токсичні. Значна частина з них не включається в природний кругообіг, накопичується в біосфері і зумовлює небажані екологічні наслідки. Відомо, що екологія – це наука взаємовідносин між живими організмами і сферою їх перебування, тому наслідки промислової і господарської діяльності людства можуть завдати непоправні збитки біосфері і велику шкоду людині.

Забруднюючі речовини, що потрапляють в природне середовище здатні переміщуватись на досить великі відстані, а закономірність цих процесів вивчена ще недостатньо. Ці речовини мігрують у великих кількостях в контурах окремих складових біосфери. Так в атмосфері вони переносяться повітряними течіями. Ступінь їх розсіювання формується швидкістю і напрямом переміщення повітряних мас і залежить від метеорологічних умов. потрапивши у воду, забруднюючі речовини окиснюються мікроорганізмами або адсорбуються частинками речовин, які є у воді.

Охорона навколишнього середовища являє собою форму відносин між суспільством і природою. Вона здійснюється різними засобами: економічними, правовими, науково-технічними, санітарно-гігієнічними, біологічними та іншими.

В загальному випадку проблема охорони навколишнього середовища зводиться до вирішення двох завдань:

1) організації раціонального природокористування;

2) забезпечення чистоти природних (екологічних) систем.

При здійсненні різних видів економічної діяльності суб'єкти господарювання використовують різноманітні природні ресурси: землю, воду, корисні копалини тощо. Проте ресурси ці обмежені. Обмеженість природних ресурсів була і залишається головною і дуже жорсткою умовою, що накладається на розвиток економіки і відповідно зростання суспільного добробуту.

Наслідком обмеженості природних ресурсів є конкуренція за їх застосування, тобто суперництво між альтернативними цілями використання ресурсів. Адже майже всі ресурси можуть використовуватися для задоволення найрізноманітніших потреб.

Раціональне природокористування означає розробку та здійснення концепції і конкретних заходів щодо раціонального використання і

відтворення природних ресурсів, гармонічну взаємодію суспільства і природи, людини і навколишнього природного середовища.

Завдання організації раціонального природокористування вирішується шляхом:

1) оптимального розподілу ресурсів між різними господарськими цілями;

2) використання технологій, що зберігають ресурси;

3) проведення заходів щодо поповнення природних ресурсів.

Іншим, не менш важливим, завданням охорони навколишнього середовища є забезпечення чистоти природних екологічних систем, тобто водного середовища, повітряного басейну, ґрунтових покровів тощо, з тим, щоб забезпечити населення екологічно чистими продуктами харчування, водою, повітрям і, в остаточному підсумку, зберегти високий рівень здоров'я населення та його активного довголіття.

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

Кваліфікаційну роботу присвячено удосконаленню регресійного рівняння для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням Java та мікросервісів, та розробці програми для його реалізації. Під час виконання магістерської роботи були поставленні задачі, які було виконано у повному обсязі:

– виконано аналіз існуючих методів та моделей оцінювання розміру програмного забезпечення та обгрунтувано необхідності досліджень;

– удосконалено нелінійне регресійне рівняння для оцінювання розміру програмного забезпечення;

– розроблено програму для оцінювання розміру програмного забезпечення з використанням удосконаленого рівняння.

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

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1.Латанська, Л.О. Оцінювання розміру інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури / Л.О. Латанська, В.І. Ковальчук // VI Міжнародна науково-технічна конференція «Комп'ютерне моделювання та оптимізація складних систем», Дніпро, ДВНЗ УДХТУ, 4-6 листопада 2020 р. – Дніпро, 2020. – С. 43-44.

2. Lewis, James. Microservices [Електронний ресурс]. – Режим доступу: https://martinfowler.com/articles/microservices.html – Назва з екрану

3. Newman, Sam. Principles of Microservices, 2015. [Електронний ресурс]. – Режим доступу: https://vimeo.com/131632250 – Назва з екрану

4. NGINX. The Future of Application Development and Delivery Is Now. [Електронний ресурс]. – Режим доступу: https://www.nginx.com/resources/library/app-dev-survey/ – Назва з екрану

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

6. Boehm, B.W. Software engineering economics [Текст] / B.W. Boehm. – Prentice-Hall, 1981. – 320 p.

7. Boehm, B.W. The COCOMO 2.0 Software Cost Estimation Model [Текст] / B.W. Boehm. – American Programmer, 2000. – 586 p.

8.Модели, методы и средства оценки стоимости програмного обеспечения [Текст]: / Д.В. Баценко, Н.А. Сидоров, Ю.Н. Василенко, Ю.В. Щебетин. [Електронний ресурс]. – Режим доступу: http:// masters.donntu.edu.ua/2010/fknt/zhukova/library/article01.htm. – Назва з екрану

9.Нащеева, А.А. Трудоемкость проекту по созданию программного продукта [Текст]: /А.А. Нащеева, Р.Д. Гутгарц// ВЕСТНИК ИрГТУ. – 2011. – №11 (58). – С. 249- 252.

10. 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

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

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

13. 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.

14. 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.

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

16. 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.

17. What accuracy statistics really measure [Електронний ресурс]. – Режим доступу: https://core.ac.uk/download/pdf/334518.pdf – Назва з екрану

18. Карпова Т.С. Базы данных: модели, разработка, реализация [Текст] / Т.С. Карпова. – СПб.: Питер, 2001. – 304с.

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ ПЗ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ

Вступ

Назва програми – ПЗ оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

Програмне забезпечення являється програмою, яка буде надавати змогу користувачу провести оцінювання розміру.

1. Підстави для розробки

Документом, на підставі якого ведеться розробка програмного продукту, є завдання на кваліфікаційну роботу «Удосконалення регресійного рівняння для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури, та розробка програми для його реалізації»

2. Призначення розробки

2.1. Функціональне призначення

Даний програмний продукт дозволить автоматизувати та скоротити час відповідних розрахунків.

2.2. Експлуатаційне призначення

Програмний продукт буде використовуватись для оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів, та оптимізації коду і побудови інтервала довіри та інтервала прогнозування на основі десяткового логарифму.

3. Вимоги до програми

3.1. Вимоги до функціональних характеристик

3.1.1. Вимоги до складу функцій, що виконуються

Програма повинна забезпечувати можливість виконання наведених нижче функцій:

1) Занесення даних.
2) Редагування даних.
3) Нормалізація даних.
4) Розрахунок верхньої та нижньої меж довірчого інтервалу.
5) Розрахунок верхньої та нижньої меж інтервалу прогнозування.

3.1.2 Вимоги до організації вхідних та вихідних даних

Вхідні дані зберігаються у таблицях бази даних. Внесення вхідних даних виконується шляхом їх введення у спеціальних формах.

Вихідні дані (прогнозне значення, довірчі інтервали та інтервали передбачення) повинні виводитися в відповідних місцях на створеній програмою формі.

3.2 Вимоги до надійності

3.2.1 Вимоги до забезпечення надійності функціонування програми

Надійне (стійке) функціонування програми має бути забезпечено надійним (стійким) функціонуванням операційної системи.

3.2.2 Відмова виконання програмного продукту через некоректні дії користувача

Відмова програми можлива внаслідок некоректних дій користувача при взаємодії з операційною системою. Щоб уникнути виникнення відмов програми за вказаною вище причини необхідно забезпечити стабільне функціонування операційної системи.

3.4 Умови експлуатації

Кліматичні умови експлуатації програмного продукту відповідають умовам експлуатації апаратної частини персонального комп'ютера.

3.5 Вимоги до інформаційної та програмної сумісності

Системні програмні засоби (не вільно поширювані), використовувані програмою, повинні бути представлені ліцензійною версією ОС Windows.

3.6 Вимоги до захисту інформації та програм

Вимоги до захисту інформації та програм відсутні.

3.7 Вимоги до маркування та упаковки

3.7.1 Вимоги до маркування

Вимоги до маркування відсутні.

3.7.2 Вимоги до упаковки

Вимоги до упаковки відсутні.

3.8 Вимоги до транспортування та зберігання

Вимоги до транспортування програми відсутні.

4 Вимоги до програмної документації

Склад програмної документації:

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

5 Техніко-економічні показники

Економічна ефективність впровадження програмного забезпечення розрахована у розділі 5. Згідно з отриманими результатами впровадження даного ПЗ є доцільним, а термін його окупності становить приблизно 4 місяці.

6 Стадії та етапи розробки програмного забезпечення

Стадії та етапи розробки ПЗ оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених мовою Java на базі мікросервісної архітектури, зазначені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Стадії розробки	Етапи робіт	Вміст робіт	Контрольні точки
1	2	3	4
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. Визначення вимог до ПЗ. Обговорення 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 22.10.2020до 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 16.11.2020до 18.11.2020

Продовження таблиці А.1

1	2	3	4
Робочий проект	Розробка ПЗ	Програмування та налагодження програми	З 19.11.2020до 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 11.12.2020до 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

ОПИС ПРОГРАМИ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ JAVA ТА МІКРОСЕРВІСІВ

1 Загальні відомості

1.1 Позначення і найменування програми

Найменування: ПЗ оцінювання розміру програмного забезпечення Інтернет-магазинів, розроблених з використанням мови Java та мікросервісів.

Позначення: «Online_store»

1.2 Мови програмування

ПЗ написане на мові програмування Java. Модулі, реалізовані на інших мовах програмування, не використовуються.

2 Функціональне призначення

2.1 Класи вирішуваних завдань і призначення програми

ПЗ «Online_store» повинно забезпечувати виконання наступних функцій:

1) Занесення даних.
2) Редагування даних.
3) Нормалізація даних.
4) Розрахунок верхньої та нижньої меж довірчого інтервалу.
5) Розрахунок верхньої та нижньої меж інтервалу прогнозування.

2.2 Функціональні обмеження

Файли повинні бути формату EXE.

3 Використовувані технічні засоби

– 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.

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

4 Виклик і завантаження

Запуск ПЗ «Online_store» в графічному режимі здійснюється двома способами:

1) за допомогою файлу запуску: Online_store.exe.

2) з використанням командного рядку: Online_store.exe

5 Вхідні і вихідні дані

Вхідні дані програми (назва проекту, кількість методів) повинні знаходитись в окремому текстовому файлі. Вихідні дані програми мають відображатися на екрані користувача.

ДОДАТОК В

ПРОГРАМА І МЕТОДИКА ВИПРОБУВАНЬ ПЗ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ

1 Об’єкт випробувань

Об’єктом випробувань є ПЗ для оцінювання розміру програмного забезпечення інтернет- магазинів.

2 Мета випробувань

Метою випробувань є перевірка програмного забезпечення на наявність помилок і відповідність реалізованих функцій зазначеним у технічному завданні, яке наведене у додатку А. У випадку прийнятного рівня відповідності програмного забезпечення вимогам, відсутності виявлених помилок і грубих недоліків, 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.

Для проведення випробувань розглядали всі функції.

4 Вимоги до програмної документації

Для проведення випробувань надаються наступні документи:

– опис програми;

– текст програми;

– інструкція користувача;

– програма та методика випробувань.

5 Засоби і порядок випробувань

Випробування проводилось в Windows 7.

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.

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

Порядок проведення випробувань.

Випробування програмного забезпечення повинні виконуватися в наступному порядку:

- 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:

2) запуск програми:

– запустили програмне забезпечення.

Внаслідок вказаних дій відкрилося головне вікно програми. Результат відповідає очікуваному.

2) функція введення даних:

– у головному меню вибрати пункт «Ввести».

Внаслідок вказаних дій повинно відкритися вікно для вводу даних про новий проект. Необхідно ввести назву проекту та натиснути кнопку «Далі». Має відкритися вікно для вводу кількості методів. Після введення кількості методів необхідно натиснути кнопку «Далі». Має відкритися вікно для перевірки введених даних. Після перевірки необхідно натиснути кнопку «Далі». Внаслідок вказаних дій інформація про новий проект повинна бути записана до бази даних і програма має перейти до функції «Оцінити». Результат роботи функції оцінювання повинен зявитися в наступному вікні. Після натискання кнопки «В меню» програма має перейти в меню програми.

ДОДАТОК Г

ІНСТРУКЦІЯ КОРИСТУВАЧА ПЗ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ

Для запуску програми необхідно запустити файл Online_store.exe. Після запуску програми відкриється вікно з меню (рис. Г.1), на якому будуть розташовані елементи керування для доступу до функцій програми.

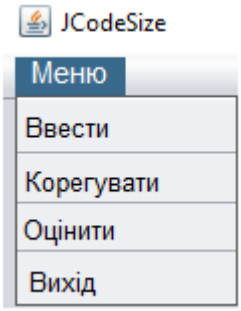

Рисунок Г.1– Меню програмного забезпечення

В меню можемо обрати одну із команд:

– Ввести інформацію про новий проект;

– Ввести інформацію для корегування існуючого проекту;

– Визвати команду оцінювання розміру програмного забезпечення;

– Вийти з програми.

При виборі команди «Ввести інформацію про новий проект» з’явиться форма для вводу назви проекту, яка представлена на рисунку Г.2:

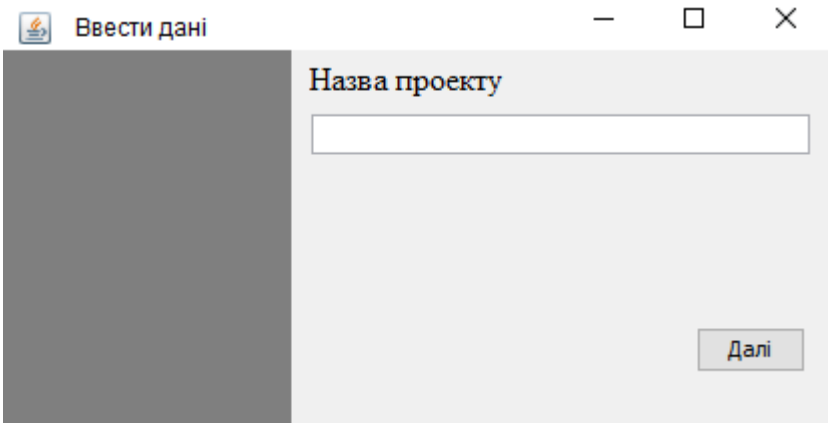

Рисунок Г.2– Форма для вводу назви проекту

Далі переходимо на наступний крок. Форма для вводу кількості методів представлена на рисунку Г.3:

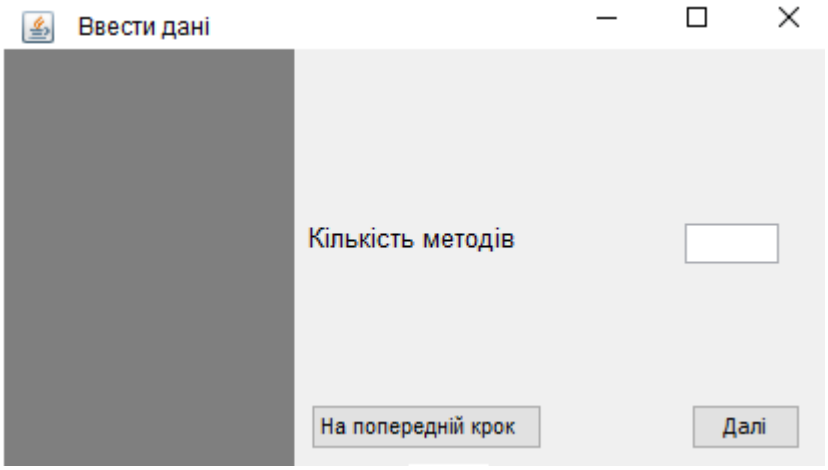

Рисунок Г.3– Форма для вводу кількості методів

Далі переходимо до наступного вікна для перевірки введених даних: назви проекту та кількості методів (рис. Г.4):

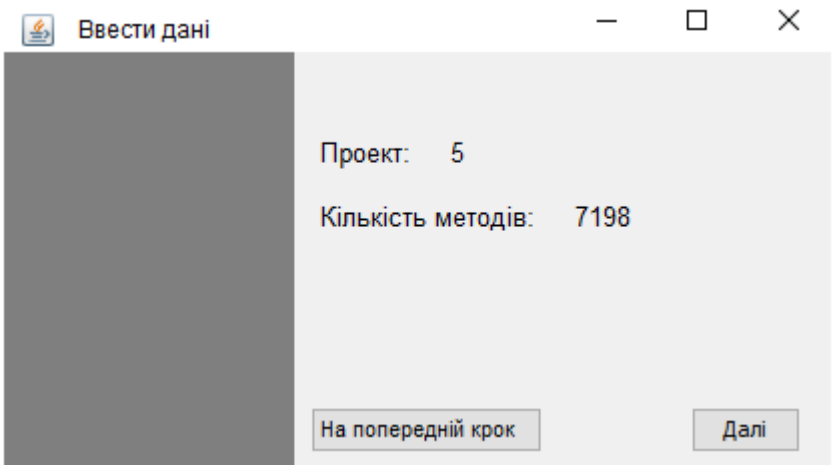

Рисунок Г.4 – Форма для перегляду результатів вводу

Після натискання на кнопку «Далі» отримаємо результат оцінювання (рис. Г.5).

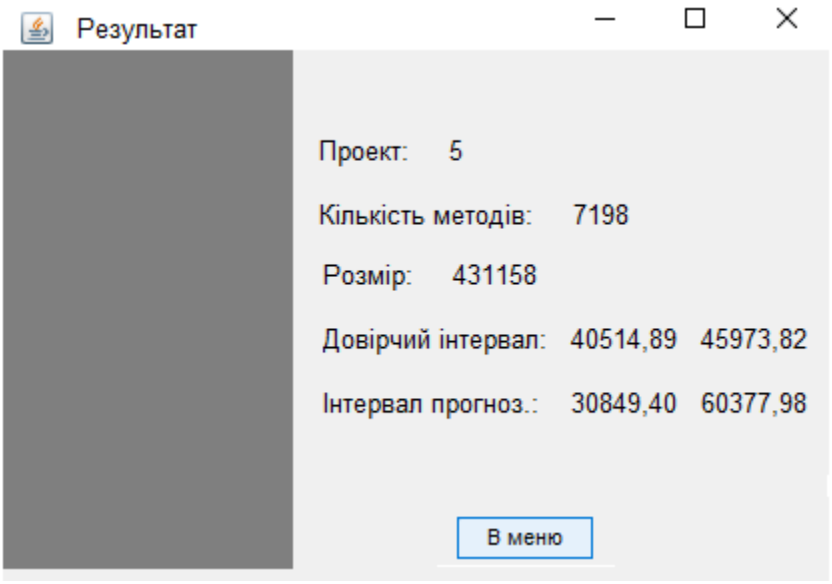

Рисунок Г.5– Форма для перегляду результатів оцінювання.

ДОДАТОК Д

ТЕКСТ ПРОГРАМИ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ІНТЕРНЕТ-МАГАЗИНІВ, РОЗРОБЛЕНИХ З ВИКОРИСТАННЯМ МОВИ JAVA ТА МІКРОСЕРВІСІВ

```
CREATE TABLE project
{
	id_project		INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
	title_project		CHAR(36) NOT NULL
} TYPE=INNODB;

CREATE TABLE pdanni
{
	id_pr_danni		INT NOT NULL,
	class_danni		REAL NOT NULL,
	INDEX		id_pr_index (id_pr_danni),
	FOREIGEN KEY (id_pr_danni) REFERENCES project
} TYPE=INNODB;

CREATE TABLE resultat
{
	id_pr_resultat		INT NOT NULL,
	size_resultat		INT NOT NULL,
	n_dov_resultat			REAL NOT NULL,
	v_dov_resultat		REAL NOT NULL,
	n_prog_resultat		REAL NOT NULL,
	v_prog_resultat		REAL NOT NULL,
	INDEX			id_pr_index (id_pr_resultat),
```

```
        FOREIGEN KEY (id_pr_resultat) REFERENCES project
} TYPE=INNODB;
package com.jcs;

import com.jcs.exceptions.*;
import com.jcs.mat.DljaRaschotov;

public class Project {

   /**
    * Возвращает наименование проекта.
    * @return наименование проекта.
    */
   public String getTitle() {
      return title;
   }

   /**
    * Устанавливает новое наименование проекта.
    * @param title новое значение наименования проекта.
    * @throws EmptyLineException   если аргумент представлен пустой строкой.
    * @throws NullPointerException если аргумент пустая ссылка.
    */
   public void setTitle(String title) throws EmptyLineException, NullPointerException{
      if(title == null)
         throw new NullPointerException();

      title = title.trim();
      if(title.length()==0)
         throw new EmptyLineException();

      this.title = title;
   }
   /**
    * Хранит наименование проекта
    */
```

```
    private String  title;

    /**
     * Хранит начальные значения факторов уровнения
     */
    private XData   sourceData;

    /**
     * Проверяет установлены ли начальные данные для расчета длины кода.
     * @return true если начальные данные установлены.
     */
    public boolean  isSourceData(){
        return sourceData != null;
    }

    /**
     * Устанавливает исходные данные.
     * @param sourceData   исходные данные для оценки длины кода.
     * @throws LessThanZeroException
     * @throws EqualsZeroException
     * @throws IllegalArgumentException
     * @throws NullPointerException
     */
    public void setSourceData(XData sourceData) throws LessThanZeroException,
EqualsZeroException, IllegalArgumentException,NullPointerException{
        if(sourceData == null)
            throw new NullPointerException();

        this.sourceData = sourceData;

    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
```

```
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel16;
    private javax.swing.JLabel jLabel17;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator2;
}
```

Step1.java

```
package com.jcs.gui;

public class Step1 extends javax.swing.JPanel {

    public Step1() {
        initComponents();
    }

    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        jSeparator1 = new javax.swing.JSeparator();
        jButton1 = new javax.swing.JButton();
        jPanel1 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jSeparator2 = new javax.swing.JSeparator();
        jPanel2 = new javax.swing.JPanel();
```

```
        jRadioButton1 = new javax.swing.JRadioButton();
        jRadioButton2 = new javax.swing.JRadioButton();

        jButton1.setText("Далі");
        jButton1.setToolTipText("На попередній крок");

        jPanel1.setBackground(new java.awt.Color(153, 153, 255));


        javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(43, 43, 43)
                .addComponent(jLabel2)
                .addContainerGap(53, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel2)
                .addContainerGap(231, Short.MAX_VALUE))
        );

        jLabel1.setFont(new java.awt.Font("Tahoma", 3, 12)); // NOI18N
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("Назва проекту");

        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addGap(41, 41, 41)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addComponent(jRadioButton2)
                .addComponent(jRadioButton1))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );
    jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jRadioButton1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jRadioButton2)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
    this.setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(jSeparator1)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
            .addComponent(jButton1)
            .addContainerGap())
        .addGroup(layout.createSequentialGroup()
            .addComponent(jPanel1,          javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                .addComponent(jLabel1,  javax.swing.GroupLayout.DEFAULT_SIZE,  249,
Short.MAX_VALUE)
                .addComponent(jTextField1)
                .addComponent(jSeparator2)
                .addComponent(jPanel2,          javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addGap(0, 11, Short.MAX_VALUE))
      );
      layout.setVerticalGroup(
         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel1,       javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGroup(layout.createSequentialGroup()
                   .addContainerGap()
                   .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                   .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                   .addGap(26, 26, 26)
                   .addComponent(jSeparator2,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    10,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
                .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jSeparator1, javax.swing.GroupLayout.PREFERRED_SIZE,
10, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jButton1)
            .addContainerGap())
        );
    }

    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JRadioButton jRadioButton1;
    private javax.swing.JRadioButton jRadioButton2;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator2;
    private javax.swing.JTextField jTextField1;
}
jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder("Оцінка розміру
коду"));

        jLabel8.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        jLabel8.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
        jLabel8.setText("Розмір:");

        jLabel9.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

        jLabel10.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        jLabel10.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel10.setText("Ніжня межа");

        jLabel11.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
```

```
        jLabel11.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel11.setText("Верхня межа");

        jLabel12.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
        jLabel12.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jLabel12.setText("Довірчий інтервал:");

        jLabel13.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
        jLabel13.setText("Інтервал прог-ня:");

        jLabel14.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

        jLabel15.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

        jLabel16.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

        jLabel17.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jSeparator2)
                .addContainerGap())
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(jLabel10)
```

```
                .addGap(36, 36, 36))
            .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel2Layout.createSequentialGroup()
                    .addGap(121, 121, 121)
                    .addComponent(jLabel8)
                    .addGap(18, 18, 18)
                    .addComponent(jLabel9))
                .addGroup(jPanel2Layout.createSequentialGroup()
                    .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                        .addComponent(jLabel12)
                        .addComponent(jLabel13))
                    .addGap(32, 32, 32)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                        .addComponent(jLabel16)
                        .addComponent(jLabel14))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jLabel15)
            .addComponent(jLabel11)
            .addComponent(jLabel17))
        .addGap(38, 38, 38))
    );
    jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                .addComponent(jLabel8)
                .addComponent(jLabel9))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jSeparator2,     javax.swing.GroupLayout.PREFERRED_SIZE,
10, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                .addComponent(jLabel10)
                .addComponent(jLabel11))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                .addComponent(jLabel12)
                .addComponent(jLabel14)
                .addComponent(jLabel15))
            .addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                .addComponent(jLabel13)
                .addComponent(jLabel16)
                .addComponent(jLabel17))
            .addContainerGap(19, Short.MAX_VALUE))
        );
```