МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Таврійський державний агротехнологічний університет
імені Дмитра Моторного
Механіко-технологічний факультет

ЗАТВЕРДЖУЮ
В.О. зав. каф. ІМКП
доц. _____Олена ДЕРЕЗА
« 12 » червня 2026 р.

Пояснювальна записка
до кваліфікаційної роботи здобувача СВО Бакалавр
(ступінь вищої освіти)

на тему: «Система візуалізації структури трикотажу кулірних переплетень з ажурними і рельєфними ефектами»

17 ПМД. 12310401.06.26/000000 ПЗ

Виконала: здобувачка вищої освіти 3С курсу,
групи 31С(ФМБ) ПМ
спеціальності 131 «Прикладна механіка» за ОПП «Комп'ютерне проектування і дизайн»
(шифр і назва спеціальності та ОПП)

_____ Аліна МАРКОВИЧ
(підпис)

Керівник доц. _____ Олександр МАЦУЛЕВИЧ
(підпис)

Консультант доц. _____ Михайло ЗОРЯ
(підпис)

Консультант доц. _____ Лариса БОЛТЯНСЬКА
(підпис)

Нормоконтроль доц. _____ Олександр МАЦУЛЕВИЧ
(підпис)

Рецензент _____ Ольга ЗАЛЕВСЬКА
(підпис)

Запоріжжя - 2026 рік

ТАВРІЙСЬКИЙ ДЕРЖАВНИЙ АГРОТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ДМИТРА МОТОРНОГО

Факультет: МТ Кафедра: ІМКП

Спеціальність 131 «Прикладна механіка»
ОПП «Комп'ютерне проектування і дизайн»

ЗАТВЕРДЖУЮ:
Завідувач кафедри ІМКП
к.т.н, доц. _____Олександр ВЕРШКОВ
«20» березня 2026р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧЦІ
Маркович Аліні Андріївні
(прізвище, ім'я, по батькові)

Тема кваліфікаційної роботи: «Система візуалізації структури трикотажу кулірних переплетень з ажурними і рельєфними ефектами» затверджена наказом по університету від 14.10.2025 за № 548-С.

1. Термін здачі студентом закінченого проекту: 12 червня 2026р.
2. Вихідні дані до роботи: середовище розробки – Eclipse Juno 4.2.1; мова програмування – Java; програмна платформа – Java, операційна система – Windows 10.
3. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) опис загальної мети і задач підсистеми; призначення програмного забезпечення; огляд методів реалізації програмної системи; опис програмної реалізації; структура, архітектура програмної системи; алгоритми обчислювальних методів; методика роботи користувача з програмною системою; економіко-організаційна частина; охорона праці; висновки.
4. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) постановка задачі, архітектура програмної системи, об'єктно-орієнтовна модель системи, блок-схема алгоритму автоматизованої побудови графічного запису, сценарій роботи користувача з програмною системою, область ефективного використання базового і нового варіантів програмного продукту, висновки.

6. Консультанти розділів роботи

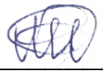
Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економіко-організаційна частина	Болтянська Л.О., к.т.н., доц.		
Охорона праці та безпека в НС	Зоря М.В., к.т.н., доцент		

Керівник  Олександр МАЦУЛЕВИЧ
(підпис)

Завдання прийняла до виконання  Аліна МАРКОВИЧ
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1.	Затвердження теми роботи.	08.05-10.05	Виконано
2.	Аналіз поставленої задачі.	11.05-12.05	Виконано
3.	Розробка архітектури системи.	13.05-15.05	Виконано
4.	Початок програмної реалізації. Розробка окремих підсистем.	18.05-20.05	Виконано
5.	Редагування програмного продукту із керівником практики на підприємстві.	21.05-22.05	Виконано
6.	Оформлення документації.	25.05-26.05	Виконано
7.	Оформлення завершального етапу розробки програмного продукту.	27.05-28.05	Виконано
8.	Захист програмного продукту	29.05-01.06	Виконано
9.	Оформлення пояснювальної записки	02.06-05.06	Виконано
10.	Захист	08.06-12.06	Виконано

Студентка-дипломниця  Аліна МАРКОВИЧ

Керівник кваліфікаційної роботи  Олександр МАЦУЛЕВИЧ

РЕФЕРАТ

Метою роботи є створення програмного продукту для графічної візуалізації структури переплетень трикотажу з ажурними і рельєфними ефектами. Програмна система передбачає побудову графічного запису переплетень із цифрового запису, введеним користувачем або завантаженим із файлу. Графічний запис дозволяє наглядно оцінити характеристики трикотажу та є основою для подальшого, більш детального, моделювання та проектування трикотажу.

Реалізовано графічний інтерфейс для взаємодії з користувачем та графічного відображення результатів моделювання.

Пояснювальна записка складається з 7 розділів, має обсяг 113 сторінок, містить 13 слайдів, 44 рисунків, 17 таблиць, 4 додатки та 30 літературних джерел.

Ключові слова: технічне завдання, програмне забезпечення, система автоматизованого проектування, Unigraphics, числове програмне керування, автоматизована система, технологічний процес, коефіцієнт міцності.

№ рядка	Формат	Позначення	Найменування	Кількість листів	Номер листа	Прим.
1	А4	17 ПМД.12310401.06.26/000000 ПЗ	Розрахунково -			
2			пояснювальна записка	113		
3	А1	17 ПМД. 12310401.06.26/110 000	Актуальність	1	0	
4			кваліфікаційної роботи			
5	А1	17 ПМД. 12310401.06.26/210 000	Графічний та цифровий			
6			запис переплетінь трикотажу	1	1	
7	А1	17 ПМД. 12310401.06.26/220 000	Існуючі програмні			
8			рішення	1	2	
9	А1	17 ПМД. 12310401.06.26/310 000	Архітектура програмної			
10			системи	1	3	
11	А1	17 ПМД. 12310401.06.26/320 000	Сценарій створення цифрового запису	1	4	
12	А1	17 ПМД. 12310401.06.26/330 000	Редагування даних та налаштування візуалізації	1	5	
13						
14						
15						
18						
19						
20						
21						

Зм	Лист	№ докум.	Підпис	Дата	17 ПМД. 12310401.06.26/000000			
Розроб.		Маркович А.А		11.06	Система візуалізації структури трикотажу кулірних переплетень з ажурними і рельєфними ефектами	Літ	Лист	Листів
Перевір.		Мацулевич О.Є.		11.06			1	77
Консул.		Зоря М.В.		11.06		ТДАТУ, 2026		
Консул.		Болтянська Л.О.		11.06				
Н.контр.		Мацулевіч ОЄ.		11.06				
Затв.		Дереза О.О.		11.06				

ЗМІСТ

ВСТУП

Важливою задачею динамічно розвиваючої економіки є необхідність скорочення термінів та вартості інженерної підготовки виробництва при одночасному підвищенні складності і якості розроблюваних інноваційних проектів. Вирішити дану проблему можливо лише на базі автоматизованого функціонального, конструкторського та технологічного проектування, шляхом розвитку, вдосконалення та впровадження у виробництво відповідних автоматизованих систем.

В даний час створені і високо розвинені системи автоматизованого проектування (САПР) в машинобудуванні, електроенергетиці, будівництві, текстильній галузі. Існуючі САПР основані на алгоритмічному підході, мають засоби інформаційної підтримки проектування. Проте переважна більшість САПР інженерної підготовки виробництва не здатні в повному обсязі підтримувати автоматизацію прийняття рішень. Стосовно проектування трикотажу вони є вузькоспеціалізованими. В низ задача інтеграції підсистем художньо-технологічного і структурно-параметричного проектування вирішена для обмеженого кола переплетень в рамках візерункових можливостей конкретної в’язальної машини. Створення універсальної системи проектування та візуалізації трикотажу дозволить максимально повно використовувати весь парк в’язального обладнання, швидко проектувати нові візерунки, а також раціонально використовувати інтелектуальну працю проектувальників.

Різноманітність структур трикотажу та складність процесу в’язання його є причиною того, що задача проектування трикотажу з заданими властивостями на цей час повністю ще не вирішена. САПР трикотажу потребує застосування сучасної методології проектування, яка дозволяє довести методи проектування до готових технологій, та повинна базуватися

на наукових основах, що потребує проведення глибоких теоретичних та експериментальних досліджень процесу в’язання, структури, параметрів та властивостей трикотажу.

Специфіка проектування трикотажу полягає в тому, що значна частина знань – це особистий досвід спеціалістів високого рівня. Крім того, знання даної предметної області слабо структуровані. Для підвищення якості проектних рішень необхідно в системі мати базу знань, використовувати методи штучного інтелекту і надавати можливість носіям технологічного досвіду зберегти його в системі, тобто провести «інтелектуалізацію» підсистем.

В даній роботі зроблена спроба автоматизації проектування мало вивченого і найбільш складного по структурі об’єкта трикотажної промисловості – трикотажу основов’язних переплетень. Інтелектуалізація проектування досягається за рахунок використання онтологічного підходу в процесі розробки моделі представлення знань і передбачає застосування сучасних методологій, що дозволяють довести методи проектування до готових технологій. Розроблені теоретичні розробки, що використовуються для автоматизованого проектування, доведені до практичної реалізації – створення програмного продукту, що дозволяє виконати в комплексі процедури художньо-технологічного та структурно-параметричного проектування переплетень трикотажу.

Крім того, в інтелектуальних САПР зростає зворотна дія ЕОМ на людину за рахунок різкого збільшення об’єму нових знань, що виробляються при функціонуванні систем, тому їх можна застосовувати для навчання і підвищення кваліфікації проектувальників, а також в якості навчальних САПР при підготовці інженерів технологів трикотажного виробництва, а також інженерів-системотехніків – розробників автоматизованих систем.

Значний внесок в розробку методів та засобів автоматизації проектування трикотажу внесли вітчизняні та закордонні вчені. В роботах

Л.А. Кудрявіна, І.І. Шалова, О.Н. Талакіной, З. Миколайчик, О.В. Попоновой, К.Н. Комова, В.А. Заваруєва, С.І. Співкіной, Є.А. Воробйова, Ф.А. Мойсеєнко, М.А. Алєксєєвой, Н.В. Холодковой, Н.М. Ромашевской, А.Н. Лєбєдєвой, Є.М. Зіміной, В.П. Король, В.І. Дзюби, а також в програмних продуктах фірми Texion Software Solution, ALS, J-Magic, KarlMayerзнайшли відображення результати досліджень по створенню алгоритмів проектування трикотажу, розробці комп'ютерної програми функціонування візерункотвірних виконуючих органів в'язальних машин і відтворення трикотажу по розрахункам [1,2]. Разом з тим автоматизоване проектування структури переплетень трикотажу є складною багатоваріантною задачею, розв'язок якої ускладнюється тим, що знання предметної області недостатньо структуровані і формалізовані. Аналіз робіт вище названих вчених та фірм показав, що характерним для більшості із них є встановлення емпіричних залежностей, справедливих лише для певного асортименту трикотажу, виду переплетень, обладнання та сировини, що робить їх непридатними для розробки універсальних алгоритмів. Розроблений програмний продукт розширює можливість набір інструментальних засобів, необхідних для відображення всієї гами можливих візерункових ефектів.

На даний час для візуалізації та моделювання переплетень трикотажу існує ряд програмних інструментів, що пропонуються закордонними інститутами та фірмами Ontolingua, KASTUS, Protégé, Cyc, OntoEdit, WebOnto, які надають наступні можливості: використання механізму логічного виводу, підтримка мов запитів, багатокористувацький режим. Основними недоліками даних інструментальних засобів є: наявність у значної частини програм закритого коду, відсутність методологічної підтримки, відсутність розширюваності, відсутність у деяких програм засобів графічного редагування, висока вартість адаптації. В зв'язку з цим виникає необхідність розробок нових, більш ефективних інструментальних

засобів представлення знань для їх використання в області автоматизованого проектування переплетень трикотажу.

Програмний додаток розробляється мовою програмування Java у програмному середовищі EclipseJuno з використанням бібліотеки Java2D для відображення графічного запису в інтерфейсі користувача.

Програмний додаток має ряд базових можливостей для вирішення поставленої задачі. Робота за програмним додатком поділена на три етапи: введення необхідних даних, побудова графічного запису та збереження результатів. Звіт з переддипломної практики представлений шістьма розділами в яких описується задача візуалізації структури трикотажу, огляд існуючих програмних рішень поставленої задачі, методи реалізації програмної системи, опис програмної системи та роботи користувача з програмою.

1. ЗАДАЧА ВІЗУАЛІЗАЦІЇ СТРУКТУРИ ПЕРЕПЛЕТЕНЬ ТРИКОТАЖУ

Сучасні темпи розвитку обчислювальної техніки, комп'ютерних технологій і програмного забезпечення надали новий імпульс розвитку систем автоматизованого проектування в багатьох галузях виробництва, в тому числі і трикотажного.

Метою роботи є створення системи візуалізації структури трикотажу основов'язних та кулірних переплетень, за допомогою інтерактивної побудови графічного запису переплетень.

Призначенням програмного комплексу є інтерактивне графічне моделювання схеми переплетень трикотажу.

Вхідними даними системи виступає цифровий запис (аналітичний запис) переплетень для кожної гребінки. Аналітичний запис кладок ниток на голки використовується в якості математичного опису основов'язаних полотен, який дозволяє визначити тип та кількість елементів петельної структури трикотажу та повністю відображає петельну структуру трикотажу. Задання вхідних даних передбачає введення наступної інформації:

- кількість гребінок;
- аналітичний запис для кожної гребінки;
- рапорт кладки по висоті гребінки;
- рапорт проборки гребінки;

Вихідна інформацією представляє собою графічне зображення (графічний запис) схеми переплетень трикотажу. Графічний запис основов'язного трикотажу дає достатнє уявлення про характер переплетення, властивості, візерунок і використовують його для складання програми візерунчатого ланцюга.

Користувачами програмної системи можуть бути фахівці легкої промисловості, студенти та викладачі вузів. Систему можна застосовувати

для навчання і підвищення кваліфікації проектувальників, а також в якості навчальних САПР при підготовці інженерів технологів трикотажного виробництва, а також інженерів-системотехніків – розробників автоматизованих систем [3,4].

2. ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ СИСТЕМИ ВІЗУАЛІЗАЦІЇ СТРУКТУРИ ПЕРЕПЛЕТЕНЬ ТРИКОТАЖУ

2.1 САПР трикотажу

Протягом останніх 10 років досягнуті значні успіхи у вирішенні такої складної комплексної задачі, як розробка систем автоматизованого проектування, що стало можливо завдяки розробкам теоретичних основ проектування та програмування, а також потужній обчислювальній техніці.

Сучасні темпи розвитку обчислювальної техніки, комп'ютерних технологій і програмного забезпечення надали новий імпульс розвитку систем автоматизованого проектування в багатьох галузях виробництва, в тому числі і трикотажного.

В сучасній трикотажній промисловості одним із головних напрямків є розвиток інформаційних технологій в області трикотажного виробництва на базі створення систем проектування і контролю технології петле утворення, яка дозволяє створювати нові технології, скорочувати сировинні, трудові і часові витрати на стадії підготовки виробництва і проектувати в'язальне устаткування з більш широкими технологічними можливостями.

Автоматизованим називається проектування, при якому окремі перетворення описів об'єкта та подання їх на різних символічних мовах і алгоритмах функціонування здійснюються шляхом взаємодії людини з ЕОМ.

Системи автоматизованого проектування є складними системами, що обумовлюється складністю: розв'язуваної задачі, керування процесом розробки, опису поведінки окремих підсистем, складністю забезпечення гнучкості кінцевого продукту. Основу САПР складає комплекс засобів автоматизації проектування, який включає в себе засоби технічного,

програмного, інформаційного, методичного і організаційного забезпечення [1].

Розробка систем автоматизованого проектування трикотажу є комплексною задачею, яка включає в себе декілька етапів [2].

I – етап художнього проектування або конструювання трикотажного полотна або виробу. На цьому етапі проводиться формування тариці рисунка, елементами якої є візерунчаті ефекти трикотажу. Для трикотжу такі матриці рисунка називають патроном рисунка. Вони не несуть інформації про структуру, технологічні особливості отримання та властивості трикотажу.

II – етап проектування даних для відтворення візерунка трикотажу. Такими даними для основов'язаних машин є схеми набору візерункотвірних ланцюгів або команд зсуву гребінок для машин з електронним механізмом візерункотворення, схеми розстановки вушкових гребінок на лініях зсуву, рапорт снування і проборки гребінок нитками – підсистема CAM (ComputerAidedManufactoring – автоматизоване забезпечення процесу вироблення).

III – етап автоматизованого проектування параметрів структури і властивостей трикотажу – підсистема CAT (ComputerAidedEngineering – автоматизоване інжінірінгове супроводження проекту). На цьому етапі проводиться автоматизоване перетворення інформації про рисунок в інформацію про структуру трикотажу.

IV – етап відтворення трикотажу за розрахунком – вироблення продукції на в'язальному устаткуванні – підсистема CAM.

V – етап підготовки техніко-економічної документації, необхідної для виготовлення стандартної продукції.

Аналіз етапів автоматизованого проектування трикотажу дозволив зробити висновок, що однією з важливіших задач є створення методології, яка дозволить здійснювати взаємозв'язок підсистем САПР, тобто забезпечити можливість переходу від патрона-матриці рисунка до програми

для постійно запам’ятовуючих приз стоїв (ПЗП) в’язальних машин і до матриці структури трикотажу, яка дозволить провести проектування його параметрів та властивостей. Ця задача не може бути вирішена без розробки системи кодування трикотажу. Проф. Л.А. Кудрявін [1] сформував основні вимоги, яким повинна відповідати система кодування трикотажу. Це представлення інформації у вигляді, придатному для використання в ЕОМ і в той же час максимально наближена до патрону рисунка, для забезпечення інформації про зовнішній вигляд трикотажу та універсальність, тобто можливість описувати всю різноманітність структур трикотажу[3].

Різноманітність структур трикотажу та складність процесу його в’язання є причиною того, що задача проектування трикотажу з заданими властивостями на цей час повністю ще не вирішена. САПР трикотажу потребує застосування сучасної методології проектування, яка дозволяє довести методи проектування до готових технологій, та повинна базуватися на наукових основах, що потребує проведення глибоких теоретичних та експериментальних досліджень процесу в’язання, структури, параметрів та властивостей трикотажу [5-7].

2.2 Розвиток систем візуалізації при автоматизованому проектуванні трикотажу

Найважливішою проблемою при автоматизованому проектуванні трикотажу розробка найбільш прийнятної системи представлення трикотажу. Одна з основних вимог до процесу автоматизованого проектування трикотажу полягає в тому, щоб досягти найбільшої наочності зображення його структури.

Для здійснення процесу автоматизованого проектування необхідно мати детальний опис структури трикотажу, методи автоматизованого визначення елементів структури трикотажу з урахуванням інформації про рисунок, алгоритм розрахунку рисунка, а також програми в’язання.

Базою, яка забезпечує САПР трикотажу, служать визначенні системи і методи опису структур трикотажу, алгоритми розрахунку їх основних параметрів, розмірів виробів, методи опису процесів петле утворення.

Вихідними даними для здійснення процесу проектування є база даних, яка може знаходитись як в самій в'язальній машині, так і в персональному комп'ютері. База даних містить в собі величезну кількість даних як про артикули і зразки, які безпосередньо виробляються на машині, так і про різні розробки у вигляді проектів або концептуальних моделей.

Інформація про переплетення, яка знаходиться в базі даних, достатньо обширна і включає в себе різні дані про колір і вид ниток, із яких вироблено рисунок або виріб, так і вихідні дані про структуру як лицьової, так і виворітної сторін.

Проектування здійснюється, у відповідності з проектрими процедурами, або в системі CAE (CjmputerAidedEngineering), або в системі CAM (ComputerAidedManufacturing). В системі CAM здійснюється вироблення продукції на в'язальному устаткуванні, яке повністю або частково оснащено комп'ютерною технікою. Застосування комп'ютерної техніки полегшує роботу й обслуговування устаткування на різних етапах виробництва.

Дані про переплетення або рисунок, що взяті із бази даних або введені за допомогою дизайн програми в комп'ютер, переносяться на устаткування. Перенесення даних може здійснюватися як за допомогою дисків, так і за допомогою інших пристроїв перенесення інформації. Такі джерела інформації називаються ПЗП (постійно запам'ятовуючий пристрій).

Інформація про переплетення переноситься на спеціалізовані персональні комп'ютери, які розташовують безпосередньо на в'язальному устаткуванні. Дані комп'ютери можна називати ОЗП (оперативно запам'ятовуючий пристрій). ОЗП керують різними системами в'язальної машини і вносять зміни у відповідності з отриманою інформацією про вироблене переплетення, його структуру та необхідні операції або

маніпуляції, які здійснюються механічними або електронними пристроями устаткування.

Отримані дані передаються на механізм відбору в'язальних машин, які безпосередньо здійснюють вироблення продукції.

Відмінною особливістю систем автоматизованого проектування переплетень трикотажу від інших програм підрахунку параметрів структури й аналізу є можливість проектування нового асортименту без здійснення процесу вироблення пробного зразка на в'язальній машині.

Дана можливість досягається шляхом використання підпрограм з побудовою зображення реального виду переплетення на екрані комп'ютера.

Застосування систем автоматизованого проектування переплетень трикотажу дозволяє зменшити час і витрати на розробку нового асортименту [8,9,10].

2.3 Розвиток інформаційних форм представлення трикотажу

Як системи аналітичної розробки трикотажу в останній час все частіше використовують підсистеми CAD і CAE, які основані на системах візуалізації. В цьому випадку вихідні дані про передбачуванні переплетення або рисунок вводяться безпосередньо в комп'ютер і за допомогою спеціалізованих програм здійснюється побудова зображень, максимально наближена до реального.

Відмітною особливістю цих програм є можливість побудови декількох варіантів зображення зразка, що проектуються, а саме: символічне зображення, зображення графічного запису трикотажу, що є найважливішою технологічною інформацією для відтворення трикотажу, і зображення структури трикотажу.

Розробка таких систем є пріоритетним завданням в сучасному проектуванні трикотажу. Для вирішення подібних завдань необхідно

здійснити всебічний опис як структури трикотажу, призначеного для проектування, так і присвячених розробці алгоритмів аналізу.

До інформації про зовнішній вид трикотажу відносять:

- кольорове зображення;
- характеристика рапорту у вигляді патрона-матриці рисунка і його розмірів;
- інформація про передбачуванні властивості;
- кількісні характеристики трикотажу, які визначаються рапортом переплетення – $R_h R_h$, $R_b R_b$, величиною А – петельного кроку, В – висоти петельного ряду та інше.

Технологічна інформація, яка необхідна для відтворення трикотажу, може бути представлена у вигляді:

- графіка прокладання ниток;
- аналітичного (цифрового) запису прокладання ниток;
- схем інформації про програми відбору робочих органів візерункоутворення;
- опису технічних особливостей процесу в’язання.

Переплетення трикотажу є найбільш важливою характеристикою якості трикотажу. Переплетення характеризує візуально макроструктуру трикотажу, представляє взаємозв’язок між ділянками зігнутих ниток у вигляді остовів петель, протяжок і накидів. Аналіз цього взаємозв’язку дає можливість визначити, у першому наближенні, передбачуваний зовнішній вид трикотажу, його візерунчасті можливості та інші характеристики.

Побудова зображення переплетень трикотажу є дуже складним і трудоємким процесом, який вимагає високої кваліфікації фахівців. Побудова складних візерунчастих і комбінованих переплетень трикотажу звичайними ручними методами практично неможлива.

2.4 Розробка методів візуалізації структури трикотажу

З розвитком комп’ютерної техніки все більшого застосування стали знаходити системиЮ які основані на графічних принципах побудови зображення і в яких замість буквено-цифрового опису об’єкт подається у вигляді графічних позначень, таких як зірочки, квадратики, риски і т.д.

В даний час комп’ютерна технологія дозволяє досягти реальної достовірності зображення об’єктів і процесів, що проектуються, а також здійснювати імітацію процесів виробництва і реальне їх моделювання.

Існують спеціальні дизайнерські системи, робота яких основана на двовимірній графіці. До таких систем відносяться: «Tex-Design», «Picture-portfolio», «PhotoModeler» та інші.

Графічна інформація займає основну частину потоків в САПР одягу. Широке застосування графічної інформації в останній час обумовлене рядом причин:

- графічна інформація володіє концентруючи ми можливостями;
- впровадження засобів комп’ютерної графіки у виробництві дозволяє застосовувати все більший обсяг інформації;
- підвищена потреба в високій якості об’єкта проектування дозволяє отримувати зображення, яке максимально наближене до реального.

Графічну інформацію, що використовується в САПР, автори книги [4] ділять на два типи: геометричну і колористичну.

До геометричної інформації автори відносять форму розроблюваного виробу, стосовно до проектування виробу. В даному випадку будь-яку деталь або виріб можна описами сукупністю простих геометричних фігур, завдяки яким отримується виріб складної форми.

Колористична інформація дозволяє більш детально описати виріб, оскільки вносить уточнення як в структуру проектованого переплетення, так і в його колір.

В роботі [5] авторами досліджувались можливість отримання найбільш раціональними способами генерованого візуального зображення таких особливостей трикотажу:

- найважливіших характеристик якості переплетення;
- технологічної інформації про графіки прокладання ниток в різних петле твірних системах (циклах петлеутворення);
- зовнішній вид поверхні трикотажу.

Відомо, що з'єднанням елементів петельної структури в певній послідовності утворюється трикотаж, а тип з'єднання, тобто взаємозв'язок цих елементів, характеризує переплетення трикотажу.

В даний час існує декілька принципових підходів до побудови переплетення трикотажу:

- Отримання переплетень трикотажу з використанням комп'ютерних мов програмування і методів 2D і 3D проектування без взаємозв'язку між найважливішими кількісними характеристиками трикотажу.
- Генерування зовнішнього виду елементів структури і автоматичне з'єднання протяжками.
- Автоматизована побудова структури переплетення трикотажу по його аналітичним моделям.
- Автоматизована побудова переплетення з використанням топологічних ниткових регулярних структур текстильних виробів з використанням показників їх зачеплення.

Для процесу напівавтоматизованої візуалізації переплетення трикотажу використовують мову програмування «VISUAL BASIC» для операційної системи «WINDOWS».

Розроблена авторами [6] система дозволяє здійснювати побудову переплетень з урахуванням відносних розмірів остовів петель (їх індекси), відображення товщини ниток і масштабування зображення. Проте система має ряд недоліків: візуалізація структури трикотажу лише з однієї гребінки, відсутність можливості задання кольорів кожної нитки, відсутність можливості збереження результатів в зручному вигляді.

2.5 Методи автоматизованого аналізу цифрового запису трикотажу

Достатнє уявлення про склад і чергування петель трикотажу дає його графічний запис, в якому умовно показують прокладання нитки при утворенні рапорту переплетення. Голки голечниць в'язальних машин позначають точками, якщо трикотаж одинарний і виробляється на однофонтурній машині, точками й хрестиками, - якщо трикотаж подвійний і виробляється на двоконтурній машині. Горизонтальний ряд точок позначає голки однієї голечниці. На машинах з двома голечницями (двофонтуриних) голки можуть розташовуватися в шахматному порядку (ластичне розташування голок) і в порядку, в якому голки задньої голечниці розташовуються навпроти голок передньої голечниці (інтерлочне розташування голок).

При аналізі структури основов'язаного трикотажу цифровий (аналітичний) запис кладок ниток вводиться в пам'ять ЕОМ. Аналіз цифрового запису кладок ниток основов'язаного трикотажу дає можливість визначити тип утворюваних остовів петель (відкриті і закриті), кількість цих петель, а також число протяжок в рапорті переплетення (візерунок).

На рисунку 2.1 приведений графічний запис прокладання ниток гребінки в основов'язаномутрикотажі. Символами $h_j h_j$ та ii позначені

відповідно номера ланок «рисунчатой цепи» та номера петельних рядків, RR - рапорт переплетення по висоті, $R_{гр}R_{гр}$ - рапорт проборки ниток в гребінку.

Прийнявши дані позначення напрямок прокладання нитки на голкивизначається виразом

$$\begin{gathered}(h_j + h_{j+1}) - (h_{j+k} + h_{j+k+1}) = \overleftarrow{HC}, \\ (h_j + h_{j+1}) - (h_{j+k} + h_{j+k+1}) = \overleftarrow{HC},\end{gathered} \quad (2.1)$$

де h_jh_j – номер ланки «рисунчатой цепи»,kk – індекс, що залежить від тактності роботи основов'язальної машини.

При двохтактній роботі $k = 2k = 2$, при трьохтактній $k = 3k = 3$ і т.д. $\overleftarrow{HC}\overleftarrow{HC}$ – вектор позначення кладки нитки на голку. Якщо вираз (2.1) більше нуля ($+\overleftarrow{HC}+\overleftarrow{HC}$), виконується умова отримання закритої петлі; при $-\overleftarrow{HC}-\overleftarrow{HC}$ (менше нуля) виконується умова утворення відкритої петлі. Якщо вираз дорівнює нулю, що відповідає утворенню переплетення ланцюжок, аналіз прокладання ведуть по ланкам «рисунчатой цепи», що виконують прокладання ниток за голками: при $h_{j+1} - h_{j+k} = 0h_{j+1} - h_{j+k} = 0$ пров'язується відкрита петля, при $h_{j+1} - h_{j+k} \neq 0h_{j+1} - h_{j+k} \neq 0$ – закрита петля.

Аналізуючи цифровий запис прокладання ниток по цим признакам, легко визначити число закритих та відкритих петель в рапорті переплетення для однієї нитки:

$$m_{о.п} = \sum_{i=1}^{R}(\pm\overleftarrow{HC}; HC), m_{о.п} = \sum_{i=1}^{R}(\pm\overleftarrow{HC}; HC), \quad (2.2)$$

де $m_{о.п}m_{о.п}$ – число відкритих та закритих петель в рапорті переплетення по висоті для однієї нитки; $\pm\overleftarrow{HC}\pm\overleftarrow{HC}$ – величина і напрямок

кладки нитки на голки; $HCHC$ – величина кладки нитки на голки при $HC = 0HC = 0$.

При відомій проборці гребінки $R_{\text{гр}} = a_{\text{гр}} + \text{б}R_{\text{гр}} = a_{\text{гр}} + \text{б}$ число різних остовів петель для всіх її пробраних ниток в рапорті вузлів визначається як

$$M_{\text{оп}(R)} = (m_{\text{о.п.о}} + m_{\text{о.п.з}})a_{\text{гр}}M_{\text{оп}(R)} = (m_{\text{о.п.о}} + m_{\text{о.п.з}})a_{\text{гр}}, \quad (2.3)$$

де $m_{\text{о.п.о}}$де $m_{\text{о.п.о}}$ – число відкритих петель в рапорті переплетення по висоті для однієї нитки основи; $m_{\text{о.п.з}}m_{\text{о.п.з}}$ – число закритих петель в рапорті переплетення по висоті для однієї нитки основи; $a_{\text{гр}}a_{\text{гр}}$ – число пробраних ниток в рапорті проборки гребінки.

При автоматизованому аналізі графіка прокладання ниток на ЕОМ число різних протяжок петель в рапорті переплетення виражається в одиницях петельного кроку трикотажу.

Проекція протяжки $\Pi_i\Pi_i$ на напрямок петельного ряду може бути виражена як різниця середніх арифметичних номерів ланок «рисунчатой цепи» суміжних петельних рядів (ii та $i + 1i + 1$):

$$(R_{ib} - 1) = \Pi_i = \frac{(h_j + h_{j+1}) - (h_{j+k} + h_{j+k+1})}{2q}$$

$$(R_{ib} - 1) = \Pi_i = \frac{(h_j + h_{j+1}) - (h_{j+k} + h_{j+k+1})}{2q}, \quad (2.4)$$

де $R_{ib}R_{ib}$ – рапорт прокладання нитки в -му петельному ряді; $\Pi_i\Pi_i$ – проекція протяжки в петельних кроках; qq – коефіцієнт, що враховує тип основов'язальної машини.

Проекція протяжки, що з'єднує суміжні петельні ряди, на напрямок петельних стовпців завжди дорівнює CC, де CC – коефіцієнт співвідношення щільності в трикотажі. З урахуванням цього протяжка $m_i m_i$, що виражена в петельних кроках, буде дорівнювати

$$m_i = \sqrt{\Pi_i^2 + C^2} m_i = \sqrt{\Pi_i^2 + C^2}, \quad (2.5)$$

де CC – коефіцієнт співвідношення щільностей.

При нитках $x_j x_j$, пробраних в гребінку, число протяжок для всього рапорта візерункавизначається як

$$M_{\eta(H,b)} = x_j M_{\eta(R_H)} M_{\eta(H,b)} = x_j M_{\eta(R_H)}.$$

В кожному циклі утворення петлі гребінка робить два зсуви і дві прокачки, як показано на схемі (рисунок 2.1, а). В загальному випадку вектор $\overrightarrow{3C3C}$ характеризує напрямок зсуву гребінки за голками. Величина зсуву гребінки $\overrightarrow{3C3C}$ виражається цілими числами кроків голки $n = (R_b - 1) n = (R_b - 1)$, де $R_b R_b$ – рапорт переплетення по ширині, і визначає тип отримуваного трикотажу. Загальна величина зсуву гребінки $\overrightarrow{3C} = nt = (R_b - 1) t \overrightarrow{3C} = nt = (R_b - 1) t$, де tt – голковий крок машини. Зсув гребінки перед голками, що в загальному випадку позначається як $\overrightarrow{HC} \overrightarrow{HC}$, завжди однаковий по величині і для трикотажу основов'язаних переплетень дорівнює одному голковому кроку: $\overrightarrow{HC} = 1 \overrightarrow{HC} = 1$.Пронумерувавши проміжки між голками цілими додатними числами(0, 1, 2 і т. д.), кожне з яких відповідає висоті ланки програмного ланцюга, можна записати зсув гребінки для кожного петельного ряду трикотажу. Наприклад цифровий запис зсувів гребінки (рисунок 2.1, в)

виконаний для основов'язаного трикотажу, рапорт переплетення якого по висоті $R_п R_п$ дорівнює 2, а графічний запис показаний на рисунку 2.1, б.

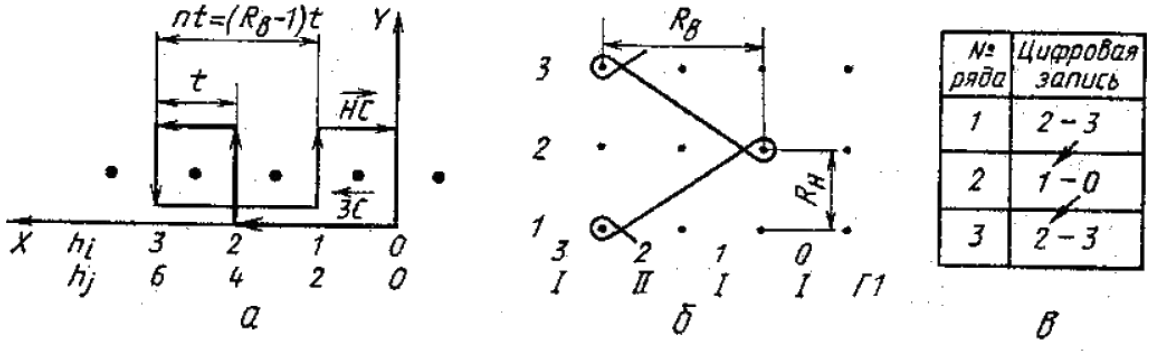


Рисунок 2.1 – Цифровий та графічний записпереплетення трикотажу

При складанні цифрового запису, що визначає програму зсувів гребінки, записуються лише зсуви гребінок перед голками (кладки ниток на голки). Для даного прикладу вони будуть: в першому петельному ряді – 2 – 3 ($\overleftarrow{HC} = 1\overleftarrow{HC} = 1$); для другого петельного ряду – 1 – 0 ($\overrightarrow{HC} = 1\overrightarrow{HC} = 1$).

3. ОГЛЯД МЕТОДІВ ТА ЗАСОБІВ РЕАЛІЗАЦІЇ СИСТЕМИ ВІЗУАЛІЗАЦІЇ СТРУКТУРИ ПЕРЕПЛЕТЕНЬ ТРИКОТАЖУ

Основні складові системи візуалізації структури переплетень трикотажу створені з використанням широких можливостей технології Java 2D. Так як дана технологія є досить надійною, та є гнучкою у своїй реалізації, то, в залежності від обставин, створення нових модулів та удосконалення розробленої системи не потребуватиме багато часу [24,25].

3.1 Java 2D як основний засіб візуалізації графічного запису

На сьогоднішній день графіка має важливе значення для мов програмування. Вона дозволяє візуалізувати будь-які програми, що надає практичність та зручність їх використання. Дозволяє створювати яскраві і цікаві веб-сторінки, зручна при розробці баз даних, написанні мобільних і комп’ютерних ігор.

Програми, тісно взаємодіють з користувачем, що сприймають сигнали від клавіатури і миші, працюють в графічному середовищі. Кожна програма, призначене для роботи в графічному середовищі, має створити хоча б одне вікно, в якому відбуватиметься його робота, і зареєструвати його в графічній оболонці операційної системи, щоб вікно могло взаємодіяти з операційною системою і іншими вікнами: перекриватися, переміщатися, змінювати розміри, згортатися в ярлик.

У технології Java графіка ускладнюється тим, що додатки Java повинні працювати в будь-якому графічному середовищі. Потрібна бібліотека класів, незалежна від конкретної графічної системи. У першій версії JDK завдання вирішили таким чином: були розроблені інтерфейси, що містили методи роботи з графічними об’єктами. Класи бібліотеки AWT

реалізують ці інтерфейси для створення додатків. Програми Java використовують дані методи для розміщення та переміщення графічних об’єктів, зміни їх розмірів, взаємодії об’єктів [24].

Бібліотека класів Java, заснованих на peer-інтерфейсах, отримала назву AWT (AbstractWindowToolkit). При виведенні об’єкта, створеного в додатку Java і заснованого на peer-інтерфейсі, на екрані створюється парний йому (peer-to-peer) об’єкт графічної підсистеми операційної системи, який і відображається на екрані.

У версії JDK 1.1 бібліотека AWT була перероблена. У неї додана можливість створення компонентів, повністю написаних на Java і не залежних від peer-інтерфейсів. Такі компоненти стали називати «легкими» (lightweight) на відміну від компонентів, реалізованих через peer-інтерфейси, названих «важкими» (heavy).

Була створена велика бібліотека «легких» компонентів Java, яка отримала назву Swing. У ній були переписані всі компоненти бібліотеки AWT, так що бібліотека Swing може використовуватися самостійно, незважаючи на те, що всі класи з неї розширюють класи бібліотеки AWT.

Бібліотека класів Swing поставлялася як доповнення до JDK 1.1. До складу Java 2 SDK вона включена як основна графічна бібліотека класів, що реалізує ідею «100% PureJava», поряд з AWT.

У Java 2 бібліотека AWT значно розширена додаванням нових засобів малювання, виведення текстів та зображень, що одержали назву Java 2D, і засобів, що реалізують переміщення тексту методом DnD (DragandDrop).

Крім того, в Java 2 включені нові методи введення/виведення InputMethod Framework і засоби зв’язку з додатковими пристроями введення/ виведення, такими як світлове перо або клавіатура Бройля, названі Accessibility.

Всі ці засоби Java 2 : AWT, Swing, Java 2D, DnD, InputMethod Framework та Accessibility склали бібліотеку графічних засобів Java, названу JFC (JavaFoundationClasses).

При створенні компонента, тобто об’єкта класу Component, автоматично формується його графічний контекст (graphicscontext). У контексті розміщується область малювання та виведення тексту і зображень. Контекст містить поточний і альтернативний колір малювання і колір фону – об’єкти класу Color, поточний шрифт для виведення тексту – об’єкт класу Font.

У контексті визначено систему координат, початок якої з координатами (0, 0) розташований у верхньому лівому кутку області малювання, вісь *Ox* направлена вправо, вісь *Oy* – вниз. Точки координат знаходяться між пікселями.

Управляє контекстом клас Graphics або новий клас Graphics2D, введений в Java 2. Оскільки графічний контекст сильно залежить від конкретної графічної платформи, ці класи зроблені абстрактними. Тому не можна безпосередньо створити екземпляри класу Graphics або Graphics2D.

Однак кожна віртуальна машина Java реалізує методи цих класів, створює їх екземпляри для компонента і надає об’єкт класу Graphics методом getGraphics() класу Component або як аргумент методів paint() і update().

В системі пакетів і класів Java 2D, основа, якої – клас Graphics2D пакета java.awt, є кілька принципово нових положень.

- Крім координатної системи, прийнятої в класі Graphics і названої координатним простором користувача (UserSpace), введена ще система координат пристрою виводу (DeviceSpace): екрану монітора, принтера. Методи класу Graphics2D автоматично переводять (transform) систему координат користувача в систему координат пристрою при виведенні графіки.
- Перетворення координат користувача в координати пристрою можна задати «вручну», причому перетворенням здатне служити будь-яке афінне перетворення площини,

зокрема, поворот на будь-який кут та/або стиснення/розтягування. Воно визначається як об’єкт класу AffineTransform. Його можна встановити як перетворення за замовчуванням методом setTransform(). Можливо виконувати перетворення «на льоту» методами transform і translate і робити композицію перетворень методом concatenate().

· Оскільки афінне перетворення «вещественно», координати задаються дробовими, а не цілими числами.

· Графічні примітиви: прямокутник, овал, дуга та інші, реалізують тепер новий інтерфейс Shape пакету java.awt. Для їх малювання можна використовувати новий, єдиний для всіх фігур, метод draw, аргументом якого здатний служити будь-який об’єкт, який реалізує інтерфейс Shape. Введено метод fill, що заповнює фігури – об’єкти класу, який реалізує інтерфейс Shape.

· Для креслення ліній введено поняття пера (stroke). Властивості пера описує інтерфейс Stroke. Клас Basicstroke реалізує цей інтерфейс. Перо володіє чотирма характеристиками:

· воно має товщину (width) в один (за замовчуванням) або кілька пікселів;

· воно може завершити лінію (endcap) закругленням – статична константа CAP_ROUND, прямим обрізом – CAP_SQUARE (за замовчуванням), або не фіксувати певний спосіб закінчення – CAP_BUTT;

· воно може сполучати лінії (linejoins) закругленням – статична константа JOIN_ROOND, відрізком прямої – JOIN_BEVEL, або просто зістиковувати – JOIN_MITER (за замовчуванням);

· воно може креслити лінію різними пунктирами (dash) і штрих- пунктиром, довжини штрихів і проміжків задаються в масиві,

елементи масиву з парними індексами задають довжину штриха, з непарними індексами – довжину проміжку між штрихами.

· Методи заповнення фігур описані в інтерфейсі Paint. Три класи реалізують цей інтерфейс. Клас Color реалізує його суцільною (solid) заливкою, клас GradientPaint – градієнтним (gradient) заповненням, при якому колір плавно змінюється від однієї заданої точки до іншої заданої точки, клас Texturepaint – заповненням по попередньо заданому зразку (patternfill).

· Букви тексту розуміються як фігури, тобто об’єкти, що реалізують інтерфейс Shape, і можуть малюватися методом draw з використанням всіх можливостей цього методу. При їх малюванні застосовується перо, всі методи заповнення та перетворення.

· Крім імені, стилю і розміру, шрифт отримав багато додаткових атрибутів, наприклад, перетворення координат, підкреслення або перекреслення тексту, виведення тексту справа наліво. Колір тексту і його фону є тепер атрибутами самого тексту, а не графічного контексту. Можна задати різну ширину символів шрифту, надрядкові і підрядкові індекси. Атрибути встановлюються константами класу TextAttribute.

· Процес візуалізації (rendering) регулюється правилами (hints), певними Константами класу RenderingHints.

З такими можливостями Java 2D стала повноцінною системою малювання, виведення тексту і зображень [25].

3.2 Евклідові перетворення як метод побудови основних графічних примітивів графічного запису

Евклідові перетворення, а саме обертання та зсув представляють основні методи, що використовуються при побудові геометричних

примітивів,необхідних для створення графічного запису переплетень структури трикотажу.

Евклідові перетвореннястановлять підгрупу афінних перетворень. Для того щоб афінні перетворення залишалися евклідовими, потрібно, щоб виконувались умови:

– *одиничності*

$$\begin{cases} a^2 + b^2 = 1; \\ d^2 + e^2 = 1, \end{cases}$$

– *ортогональності*

$$ad + be = 0.$$

Основний інваріант евклідових перетворень – відстань (зберігаються відстані між будь-якими двома точками об’єкта). Як наслідок зберігається величина кута. Евклідові перетворення розкладаються на зсув і обертання[8].

В основі методу *однорідних координат* лежить подання про те, що кожна точка в *n*-вимірному просторі може розглядатися як проекція точки з (*n*+1)-вимірного простору.

Точку в тривимірному просторі – в однорідних координатах зображують у вигляді чотиривимірного вектора:

$$\begin{bmatrix} x' & y' & z' & h \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \times \mathbf{T},$$

де $\mathbf{T}$ – матриця перетворення.

Перетворення з однорідних координат у декартові задають формулою

$$\begin{bmatrix} x^* & y^* & z^* & 1 \end{bmatrix} = \begin{bmatrix} \frac{x'}{h} & \frac{y'}{h} & \frac{z'}{h} & 1 \end{bmatrix}.$$

Властивості однорідних координат дозволяють визначати за допомогою єдиної матриці всі перетворення: зсув, повороти, аксонометричні або центральні проекції, а також будь-які поєднання перетворень у вигляді добутку матриць. Використання однорідних координат дозволяє застосовувати єдиний математичний апарат для просторових перетворень (поворотів, масштабування й перенесення) точок, прямих, квадратичних і бікубічних поверхонь і ліній[9].

Нехай *M* – довільна точка на площині з координатами*x, y* відносно заданої прямолінійної координатної системи. Однорідними координатами цієї точки називається будь-яка трійка одночасно нерівних нулю чисел x', y', *w,* пов'язаних із заданими числами *x* й *y*такими співвідношеннями:

$$x = \frac{x'}{w};\ y = \frac{y'}{w}.$$

Точку*T*(5, 7) у двовимірному просторі, наприклад, можна записати в однорідних координатах як *T2*(15, 21, 3) або як *T2*(500, 700, 100) і т. д. Очевидно, що позначення (*x, y*) являє собою звичайний запис для точки (*x, y,*1) у тривимірному просторі.

Евклідові перетворення поділяють на зсув й обертання.

Зсув. У параметричному вигляді це:

$$\begin{cases} x' = x + m; \\ y' = y + n; \\ z' = z + l, \end{cases}$$

або у векторно-параметричному:

$$\mathbf{r}' = \mathbf{r} + \mathbf{t},$$

де**r**(*x*, *y*, *z*) –координати точки об'єкта, **r'**(*x*', *y*', *z*') – координати точки перетвореного об'єкта, **t**(*m*, *n*, *l*)– вектор зсуву.

Матриця зсуву для однорідних координат має вигляд:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ m & n & l & 1 \end{bmatrix}.$$

Обертання. Розглянемо обертання навколо координатних осей без перенесення її початку. Виконаємо обертання навколо осі *Z* (рисунок 3.1). Для цього позначимо: *r* – довжина вектора, а□– кут між вектором і віссю *X*. Вектор положення обертається навколо осі *Z* на кут □ і потрапляє в точку *P*'. Запишемо вектори положення для *P*та *P'*:

$$\mathbf{P} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} = \begin{bmatrix} r\cos\phi & r\sin\phi & 0 & 1 \end{bmatrix};$$

$$\mathbf{P}' = \begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} r\cos(\phi+\theta) & r\sin(\phi+\theta) & 0 & 1 \end{bmatrix}.$$

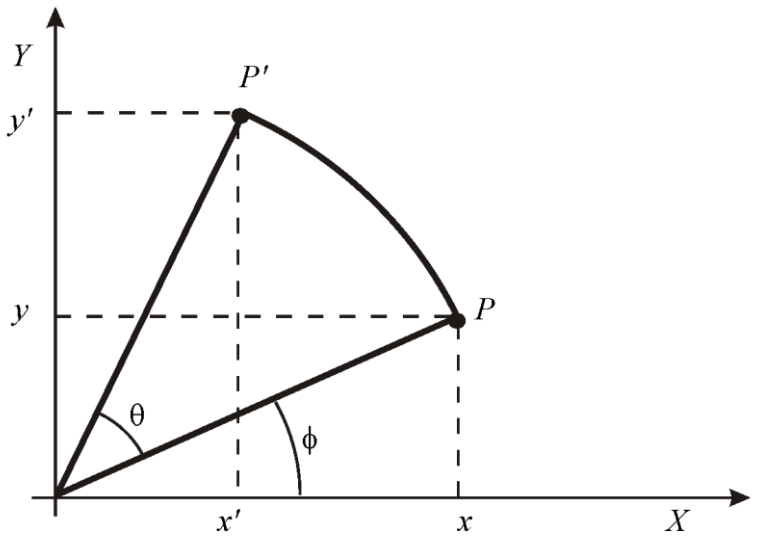


Рисунок 3.1 – Обертання навколо осі *Z*

Використовуючи формулу длякосинусівсуми кутів, перепишемо вираз для точки *P*’ у вигляді:

$$\mathbf{P}' = \begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} r(\cos\phi\cos\theta - \sin\phi\sin\theta) & r(\cos\phi\sin\theta + \sin\phi\cos\theta) & 0 & 1 \end{bmatrix}$$

.

Використовуючи певні співвідношення для *x, y, z* точки *P*, можна переписати вираз для*P*’ так:

$$\mathbf{P}' = \begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x\cos\theta - y\sin\theta & x\sin\theta + y\cos\theta & 0 & 1 \end{bmatrix}.$$

Отже, перетворена точка має координати:

$$\begin{cases} x' = x\cos\theta - y\sin\theta, \\ y' = x\sin\theta + y\cos\theta, \\ z' = z. \end{cases}$$

Матриця перетворення для обертання навколо осі *Z* має вигляд:

$$\mathbf{T} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (3.2)$$

Обертання вважається додатним за правилом правої руки, тобто у напрямі за годинниковою стрілкою, якщо дивитися з початку координат у додатному напрямі осі обертання.

Аналогічно матрицю для обертання навколо осі *X* на кут □записуємо у вигляді:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\psi & \sin\psi & 0 \\ 0 & -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.3)$$

У разі обертання навколо осі *Y*на кут матриця перетворення має вигляд:

$$\mathbf{T} = \begin{bmatrix} \cos\phi & 0 & -\sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ \sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.4)$$

Знаки синусів у матриці (4) протилежні знакам цих членів у матрицях (2), (3). Це потрібно, щоб виконувалась умова про додатний напрям за правилом правої руки. Детермінант кожної з матриць перетворень дорівнює +1, що й треба для повного обертання.

Тривимірні обертання обчислюють перемноженням матриць, тому вони не комутативні, тобто порядок множення впливає на кінцевий результат. У загальному випадку оберненою для будь-якої матриці перетворення повного повороту, тобто з визначником, що дорівнює +1, є її транспонована матриця (такі матриці називають ортогональними) [26].

3.3 XML як засіб повторного використання результатів моделювання та візуалізації

XML (ExtensibleMarkupLanguage) – це нова SGML-похідна мова розмітки документів, що дозволяє структурувати інформацію різного типу, використовуючи для цього довільний набір інструкцій.

XML призначений для зберігання структурованих даних (замість існуючих файлів баз даних), для обміну інформацією між програмами, а також для створення на його основі більш спеціалізованих мов розмітки (наприклад, XHTML), які інколи називають словниками. XML є спрощеною підмножиною мови SGML.

XML – це мова розмітки, що описує цілий клас об'єктів даних, що називаються XML-документами. Ця мова використовується в якості засобу для опису граматики інших мов і контролю за правильністю складання документів.

Специфікація XML була запропонована консорціумом W3C (організацією по стандартизації нових веб-технологій) в якості рекомендації, затверджена в 1998 році.

Сьогодні XML може використовуватися в будь-яких додатках, яким потрібна структурована інформація – від складних геоінформаційних систем, з гігантськими обсягами переданої інформації до звичайних настільних програм, що використовують цю мову для опису службової інформації. Можна виділити безліч завдань, пов'язаних із створенням та обробкою структурованої інформації, для вирішення яких може використовуватися XML:

- В першу чергу, ця технологія може виявитися корисною для розробників складних інформаційних систем, з великою кількістю програм, пов'язаних потоками інформації самої різної структури. У цьому випадку XML-документи виконують роль універсального формату для обміну інформацією між окремими компонентами великої програми.
- XML є базовим стандартом для нової мови опису ресурсів, RDF, що дозволяє спростити багато проблеми в Web, пов'язані з пошуком потрібної інформації, забезпеченням контролю за вмістом мережевих ресурсів, створення електронних бібліотек і т.д.

- Мова XML дозволяє описувати дані довільного типу і використовується для представлення спеціалізованої інформації, наприклад хімічних, математичних, фізичних формул, медичних рецептів, нотних записів, і т.д.
- XML-документи можуть використовуватися в якості проміжного формату даних в триланкових системах. Зазвичай схема взаємодії між серверами додатків і баз даних залежить від конкретної СУБД і діалекту SQL, використовуваного для доступу до даних. Якщо ж результати запиту будуть представлені в деякому універсальному текстовому форматі, то ланка СУБД стане «прозорою» для програми. Крім того, сьогодні на розгляд W3C запропонована специфікація нової мови запитів до баз даних XQL, який в майбутньому може стати альтернативою SQL.
- Інформація, що міститься в XML-документах, може змінюватися, передаватися на машину клієнта і оновлюватися по частинах. Розробляються специфікації XLink і XPointer, що дозволять посилатися на окремі елементи документа з урахуванням їх вкладеності і значень атрибутів.
- XML може використовуватися в звичайних додатках для зберігання і обробки структурованих даних в єдиному форматі.

XML-документ представляє собою звичайний текстовий файл, в якому за допомогою спеціальних маркерів створюються елементи даних, послідовність і вкладеність яких визначає структуру документа і його зміст. Основною перевагою XML документів є те, що при відносно простому способі створення та обробки (звичайний текст може редагуватися будь-яким тестовим процесором і оброблятися стандартними XML-аналізаторами), вони дозволяють створювати структуровану інформацію, яку добре «розуміють» комп’ютери.

XML дозволяє описувати і передавати такі структуровані дані, як:

- окремі документи;

- метадані, що описують вміст якого-небудь вузла Інтернет;
- об’єкти, що містять дані і методи роботи з ними (наприклад, елементи управління ActiveX або об’єкти Java) ;
- окремі записи (наприклад, результати виконання запитів до баз даних);
- різноманітні веб-посилання на інформаційні та людські ресурси в Інтернеті.

Всі описані вище переваги XMLформату зумовили вибір його як засобу збереження та повторного використання результатів моделювання та візуалізації (графічного запису) структури переплетень трикотажу.

3.4 Прикладні і системні засоби програмування та середовища проектування програмного забезпечення

Даний програмний продукт розроблявся мовою програмування Java у програмному середовищі EclipseIndigo 3.7.

EclipseIndigo 3.7 – це великий набір засобів управління життєвим циклом застосунку для забезпечення якості результатів, від етапу проектування до етапу розгортання. При створенні нових рішень або вдосконаленні існуючих застосунківEclipseIndigo 3.7 дозволяє втілити ідеї в життя завдяки підтримці різних платформ та технологій, включаючи хмарні та паралельні обчислення.

Мова програмування Java займає деяку проміжну позицію: із стандарту мови прибрані найбільш неприємні і неоднозначні особливості C++, але в той же час мова зберегла потужній можливості, властиві для таких мов, як C++, C# або VB.

У нього входить багато корисних особливостей — простота, об’єктна орієнтованість, типова захищеність, «збірка сміття», підтримка сумісності

версій і багато іншого. Дані можливості дозволяють швидко і легко розробляти програми.

В Java, як в безсумнівно сучасній мові, також існують характерні особливості для обходу можливих помилок. Наприклад, крім згаданого вище механізму «збірки сміття», там всі змінні автоматично ініціалізувалися середовищем і володіють типовою захищеністю, що дозволяє уникнути невизначених ситуацій у випадку, якщо програміст забуде ініціалізувати змінну в об’єкті або спробує провести неприпустиме перетворення типів.

Також в Java були зроблені заходи для виключення помилок при оновленні програмного забезпечення. Зміна коду, в такій ситуації, може змінити суть самої програми. Це дозволяє обійти помилки в коді і забезпечити гнучку сумісність версій. Також новою особливістю є пряма підтримка інтерфейсів і наслідування інтерфейсів. Дані можливості дозволяють розробляти складні системи і розвивати їх з часом.

В Java була уніфікована система типів, тепер ви можете розглядати кожен тип як об’єкт. Незважаючи на те, використовуєте ви клас, структуру, масив або вбудований тип, ви можете звертатися до нього як до об’єкта. Об’єкти зібрані в пакети, які дозволяють програмно звертатися до чого-небудь. Це означає що замість списку включаються файлів заголовків у своїй програмі ви повинні написати які пакети, необхідні для доступу до об’єктів і класів усередині них, ви хочете використовувати.

За умовчанням, Java забороняє пряме управління пам’яттю, надаючи взамін багату систему типів і збірку сміття. Як наслідок, в Java активно використовується всього один оператор доступу «.».

Перетворення типів в Java значно обмеженіший, ніж в C++, зокрема, більшість перетворень можуть бути здійснене тільки явним чином. Крім того, всі приведення повинні бути безпечними (тобто заборонені неявні перетворення з переповнюванням, використання цілих змінних як покажчиків і тому подібне). Природно, це помітно спрощує аналіз типів при компіляції

У Java немає множинного наслідування, замість нього пропонується використовувати реалізацію декількох інтерфейсів.

У наш час, коли посилюється зв’язок між світом комерції і світом розробки програмного забезпечення, і корпорації витрачають багато зусиль на планування бізнесу, відчувається необхідність у відповідності абстрактних бізнес процесів їх програмним реалізаціям.

На жаль, більшість мов реально не мають прямого шляху для зв’язку бізнес логіки та коду. Наприклад, сьогодні багато програмістів коментують свої програми для пояснення того, які класи реалізують якийсь абстрактний бізнес об’єкт.

Java дозволяє використовувати типізовані, розширювані метадані, які можуть бути прикріплені до об’єкта. Архітектурою проекту можуть визначатися локальні атрибути, які будуть пов’язані з будь-якими елементами мови — класами, інтерфейсами і т.д.

Розробник може програмно перевірити атрибути будь-якого елементу. Це істотно спрощує роботу, наприклад, замість того, щоб писати автоматизований інструмент, який буде перевіряти кожен клас або інтерфейс, на те, чи є він дійсно частиною абстрактного бізнес об’єкта, можна просто скористатися повідомленнями заснованими на визначених в об’єкті локальних атрибутах.

Всі ці преваги зумовили вибір платформи Java для розробки програмного забезпечення дипломної роботи.

4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇСИСТЕМИ ВІЗУАЛІЗАЦІЇ СТРУКТУРИ ПЕРЕПЛЕТЕНЬ ТРИКОТАЖУ

Проект розробляється на програмній платформі Java, а тому необхідно встановлювати JVM (JavaVirtualMachine) на операційну системуна якій буде використовуватись програмний продукт.

4.1 Архітектура програмної системи

Розроблена програмна система складається з наступних основних модулів (рисунок 4.1):

- модуль введення/виведення даних – відповідає за введення цифрового запису (введення даних можливе або з графічного інтерфейсу користувача або з XMLфайлу, що був збережений раніше) та збереження графічного запису (в XMLфайлі). Для реалізації роботи з XMLфайлами використовувалась стандартна бібліотекаJAXB, що є частиною стандартного набору класів мови Java (JSDK);
- модель даних – представляє собою набір класів, що реалізують структуру цифрового запису переплетень та містять додаткову інформацію (рапорт, пробору для кожної гребінки, колір ниток та ін.) в програмі;
- розрахунковий модуль – відповідає за побудову основних графічних примітивів, необхідних для створення графічного запису переплетень. Розрахунковий модуль використовує модуль математичних розрахунків для знаходження параметрів (координат дотичних) необхідних для побудови;

· модуль візуалізації – відповідає за відображення (візуалізацію) розрахунків (графічного запису переплетень) на екрані монітору (в графічному інтерфейсі користувача), використовує можливості бібліотеки Java2D.

4.2 Об’єктно-орієнтовна модель програмної системи

Розроблена програмна система включає наступні основні пакети: com.makalex.diplom.gui (включає класи, що відповідають за графічний інтерфейс користувача), com.makalex.diplom.calc (включає класи, що відповідають за проведення розрахунків та побудову графічного запису переплетень структури трикотажу), com.makalex.diplom.model (включає класи, що відповідають представлення даних всередині програмної системи) та com.makalex.diplom.util (включає додаткові класи). Пакети дозволяють об’єднувати класи за їх функціональним та структурним призначенням в одну область імен. Схема взаємодії між пакетами представлена на рисунку 4.2.

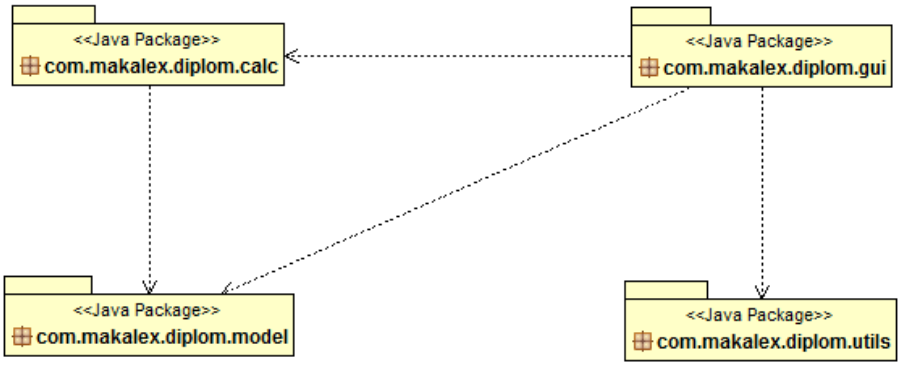


Рисунок 4.2 – Діаграма взаємодії пакетів

Розглянемо кожен з пакетів більш детально. На рисунку 4.3 представлена діаграма основних класів пакету com.makalex.diplom.gui та їх взаємодія між собою.

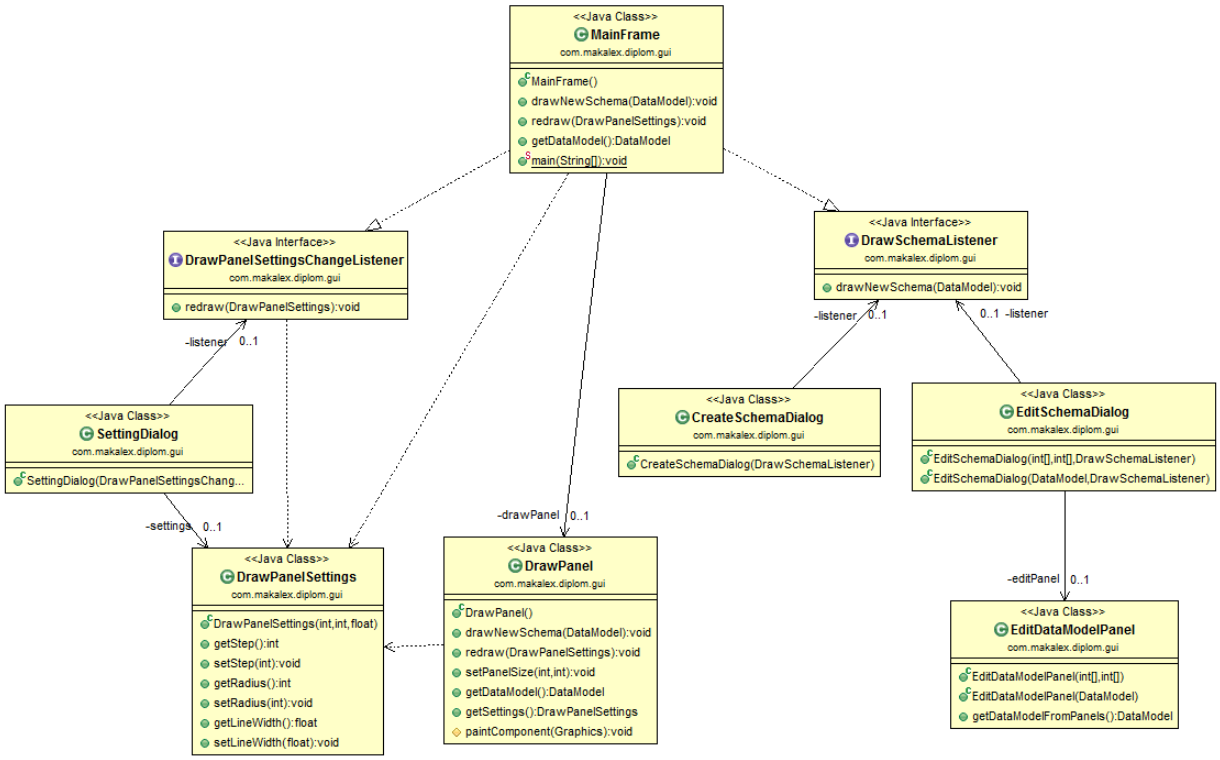


Рисунок 4.3 – Діаграма класів пакету com.makalex.diplom.gui

Головним класом пакету є клас MainFrame, що представляє собою головне вікно програми, яке містить всі необхідні компоненти для інтерактивної взаємодії з користувачем. Клас MainFrame містить в собі клас DrawPanel – графічна панель, на якій відбувається безпосереднє малювання графічного запису. CreateSchemaDIalog, EditSchemaDIalog, SettingDialog– класи, що відповідають за створення діалогових вікон для введення нового цифрового запису, редагування цифрового запису та налаштувань графічної панелі відповідно.

На рисунку 4.4 представлена діаграма класів пакету com.makalex.diplom.calc та їх взаємодія між собою.

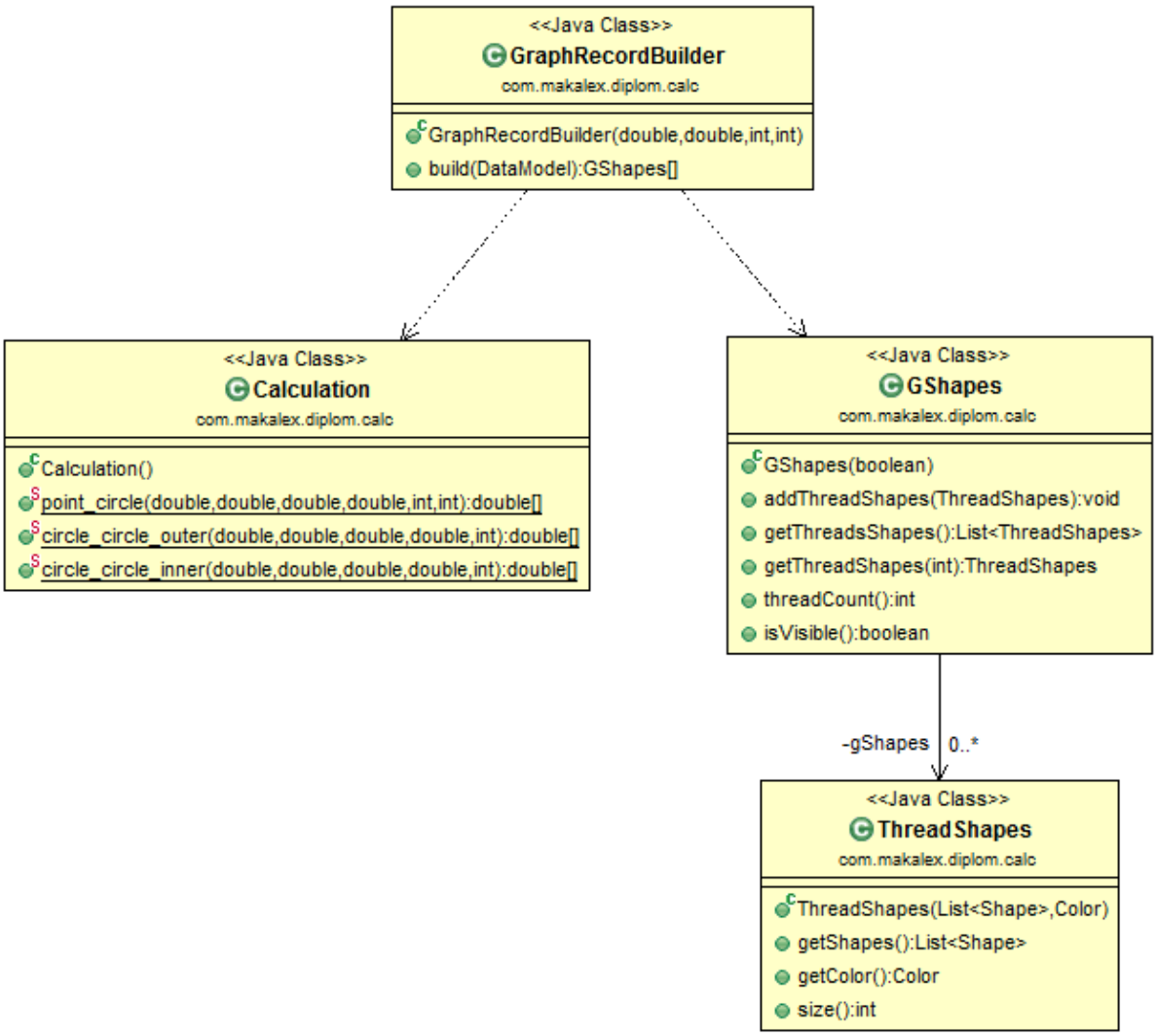


Рисунок 4.4 – Діаграма класів пакету com.makalex.diplom.calc

Головним класом пакету є клас GraphRecordBuilder, що відповідає за побудову графічного запису структури переплетень трикотажу. Клас Calculationмістить в собі методи, що необхідні для знаходження координат дотичних різних типів, які використовуються при побудові основних примітивів графічного запису. Клас GShapesмість в собі дані про графічні примітиви кожної гребінки. Клас ThreadShapes містить дані про окрему нитку в гребінці.

На рисунку 4.5 представлена діаграма класів пакету com.makalex.diplom.model та їх взаємодія між собою.

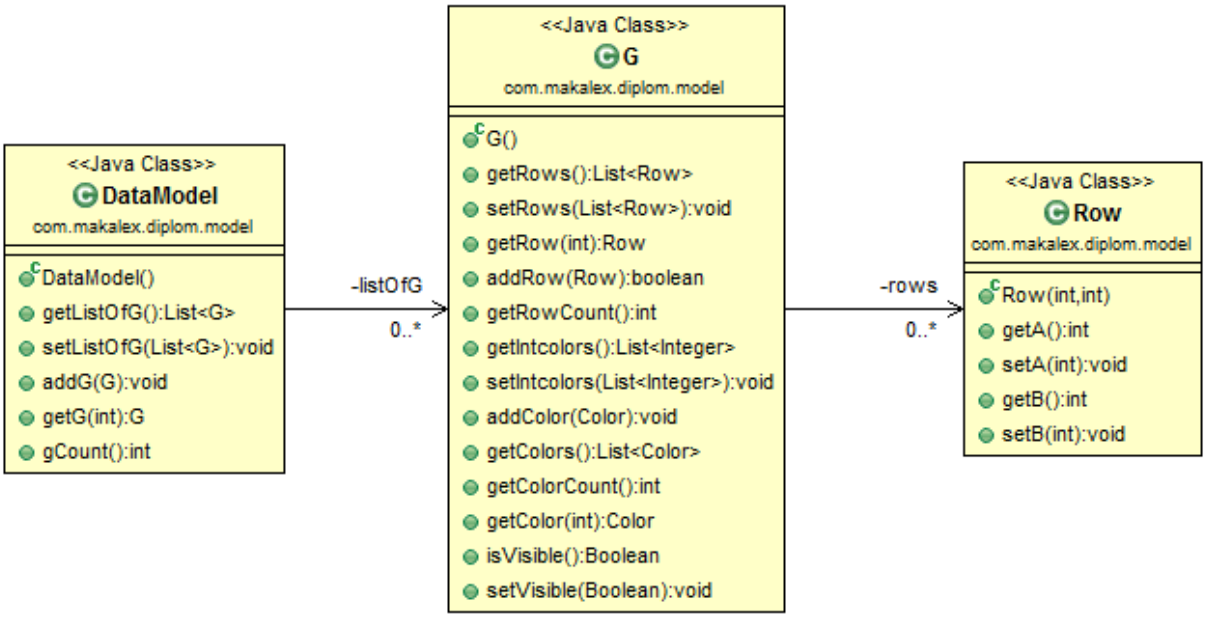


Рисунок 4.5 – Діаграма класів пакету com.makalex.diplom.model

Головним класом пакету є клас DataModel, що інкапсулює в собі всю необхідну інформацію для побудови графічного запису. Зокрема містить дані про кожну гребінку (за реалізацію відповідає клас G), що включає цифровий запис та проборку. Клас Row представляє собою реалізацію одного рядка цифрового запису [24,25].

4.3 Збереження результатів у XML файл

По завершенню створення та редагування цифрового та графічного записів переплетень трикотажу дані необхідно зберегти для повторного їх використання. Для цього використовуються мова розмітки XML, що надає зручну та декларативну форму представлення інформації. На рисунку 4.6 представлений приклад вихідного XML файлу, що містить дані про цифровий запис та проборку для кожної гребінки.


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Data>
    <G>
        <Colors>
            <c>-16777216</c>
        </Colors>
        <AnalyticRecord>
            <row>
                <a>1</a>
                <b>2</b>
            </row>
            <row>
                <a>2</a>
                <b>1</b>
            </row>
            <row>
                <a>1</a>
                <b>0</b>
            </row>
        </AnalyticRecord>
        <visible>true</visible>
    </G>
    <G>
       ...
    </G>
    ...
</Data>
```

Рисунок 4.6 – Збереження результатів у XMLфайл

XML-документ представляє собою звичайний текстовий файл, в якому за допомогою спеціальних маркерів створюються елементи даних, послідовність і вкладеність яких визначає структуру документа і його зміст. Основною перевагою XML документів є те, що при відносно простому способі створення та обробки (звичайний текст може редагуватися будь-яким тестовим процесором і оброблятися стандартними XML-аналізаторами), вони дозволяють створювати структуровану інформацію, яку добре «розуміють» комп'ютери [24,25].

4.4 Алгоритм побудови основних примітивів графічного запису

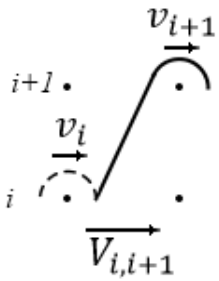
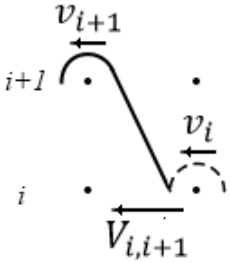
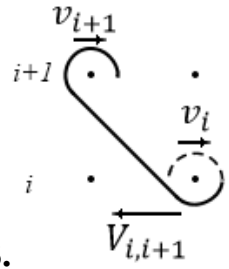
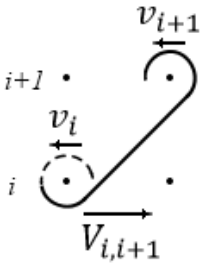
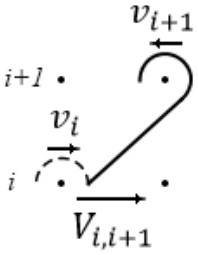
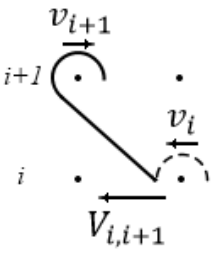
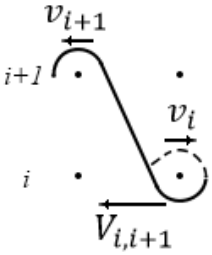
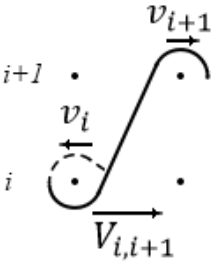
Розглянемо процес побудови графічного запису більш детально. Позначимо перше число для кожного ряду цифрового запису як $a_i a_i$, друге

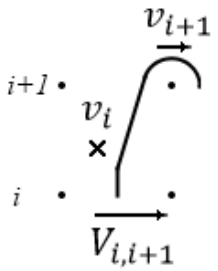
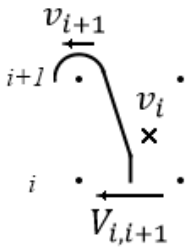
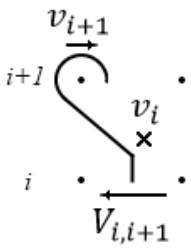
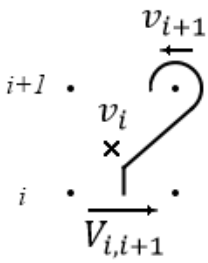
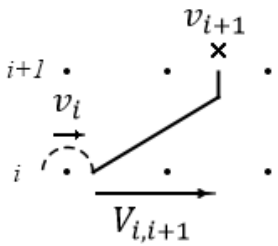
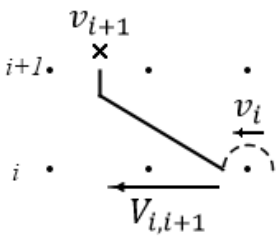
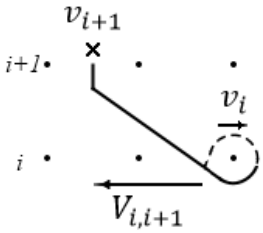
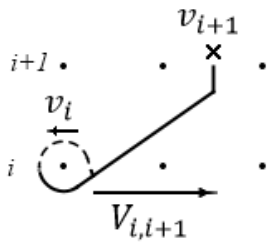
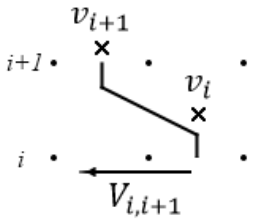
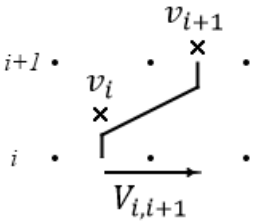
– як $b_i b_i$, де ii – номер ряду. На кожні ітерації побудови графічного запису достатньо використовувати значення двох суміжних рядів ($a_i a_i$, $b_i b_i$, $a_{i+1} a_{i+1}$, $b_{i+1} b_{i+1}$). Необхідно визначити напрямок зміщення гребінки для ii-го та $i+1i+1$ рядів та напрямок зміщення між рядами:

$$v_i = a_i - b_i$$
$$v_{i+1} = a_{i+1} - b_{i+1}$$
$$V_{i,i+1} = \max(a_i, b_i) - \max(a_{i+1}, b_{i+1})$$

Кожне з наведених вище величин ($v_i v_i$, $v_{i+1} v_{i+1}$, $V_{i,i+1} V_{i,i+1}$) може приймати одне з трьох значення – більше нуля, менше нуля або рівне нулю, що відповідає зміщенню гребінки вправо, вліво та зміщення гребінки між рядами відповідно. Проаналізувавши всі можливі варіанти, отримаємо основні примітиви графічного запису переплетень, що представленні в таблиці 3.1.

Таблиця 3.1 – Основні складові (примітиви) графічного запису

1. 	**2.** 	**3.** 	**4.** 
5. 	**6.** 	**7.** 	**8.** 

9. 	**10.** 	**11.** 	**12.** 
13. 	**14.** 	**15.** 	**16.** 
17. 	**18.** 		

Послідовно проходячи від ряду до ряду ділянки з’єднуються, таким чином утворюючи візерунок (рисунок 3.2).

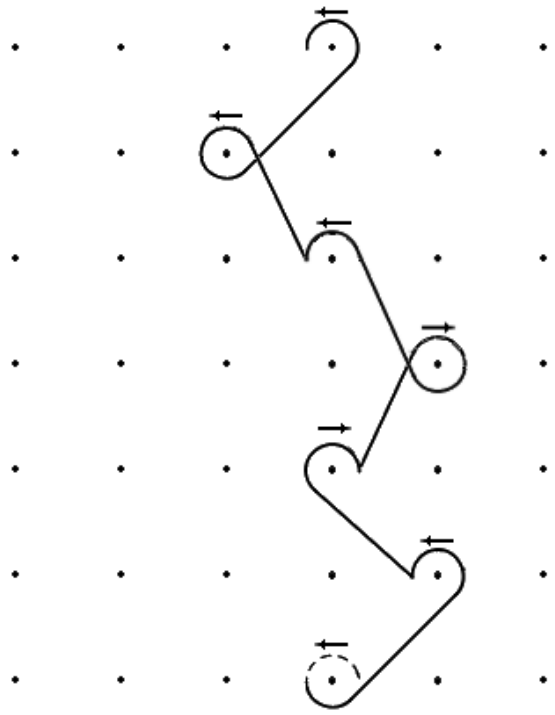


Рисунок 3.2 – Приклад графічного запису

При розрахунках використовувались наступні типи геометричної побудови:

1. дотична до кола (ділянки типу 1, 2, 5, 6, 9, 10, 11, 12, 15, 16);
2. зовнішня дотична до двох кіл (ділянки типу 3, 4);
3. внутрішня дотична до двох кіл (ділянки типу 7, 8).

Розглянемо алгоритм знаходження прямої, дотичної до кола. Так як можна провести дві дотичні з точки до кола – розглянемо ці випадки.

Знайдемо першу дотичну на прикладі ділянки типу 1. Позначимо радіус кола та міжголковий крок як rr та hhвідповідно (рисунок 3.3). Координати точок B, C, D, EB, C, D, E – відомі, також відомі значення rr та hh.

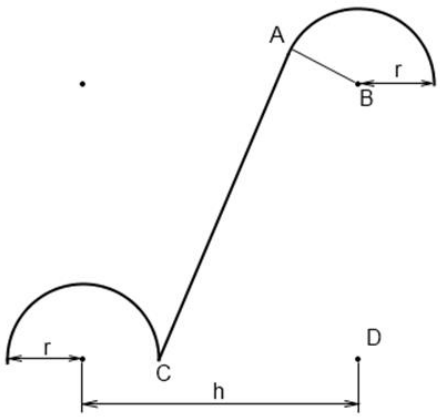


Рисунок 3.3 – Знаходження першої дотичної до кола на прикладі ділянки типу 1

Необхідно знайти координати точки AA (точка дотику). Для цього можемо виконати поворот точки EE на кут tt ($t = \alpha + \beta t = \alpha + \beta$) (рисунок 3.4).

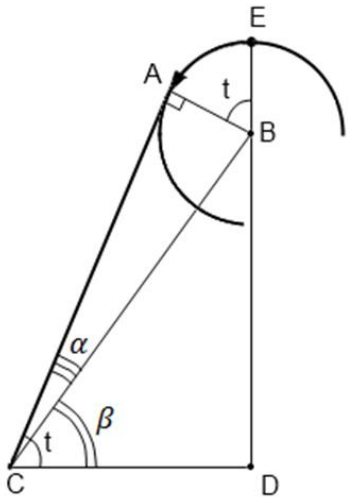


Рисунок 3.4 – Знаходження першої дотичної до кола

Так як кути AA та DD – прямі, то:

$$\alpha = \arcsin\frac{AB}{BC};$$

$$\beta = \arctan\frac{BD}{CD};$$

Використавши формулу для повороту точки у двохвимірному просторі навколо початку координат, отримаємо:

$$x1 = x\cos t + y\sin t$$
$$y1 = x\sin t - y\cos t$$

та прийнявши точку BB за початок координат, отримаємо:

$$A_x = E_x\cos t + E_y\sin t$$
$$A_y = E_x\sin t - E_y\cos t$$

Дана дотична використовується при побудові ділянок типу 1, 6, 9, 11, 16.

Знайдемо другу дотичну на прикладі ділянки типу 2 (рисунок 3.5).

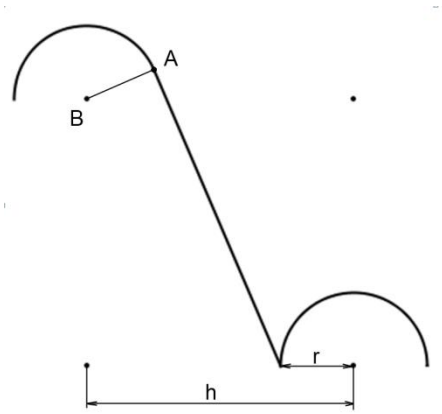


Рисунок 3.5 – Знаходження другої дотичної до кола на прикладі ділянки типу 2

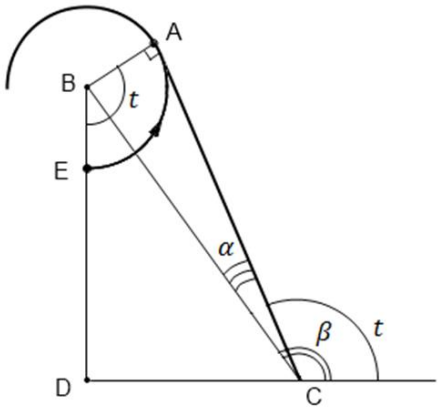


Рисунок 3.6 – Знаходження другої дотичної до кола

Координати точки AA знаходимо аналогічно як і для першої дотичної за виключеннямрозрахунку кутаtt, який визначається як $t = \beta - \alpha t = \beta - \alpha$ (рисунок 3.6).

Дана дотична використовується при побудові ділянок типу 2, 5, 10, 12, 15.

Розглянемо алгоритм знаходження внутрішньої дотичної до двох кіл на прикладі ділянки типу 7 (рисунок 3.7).

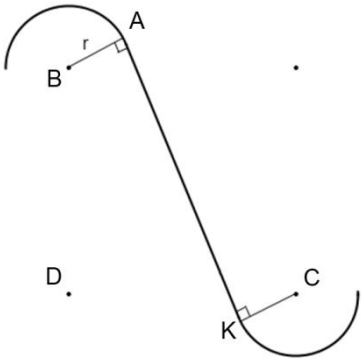


Рисунок 3.7 – Знаходження внутрішньої дотичної до двох кіл на прикладі ділянки типу 7

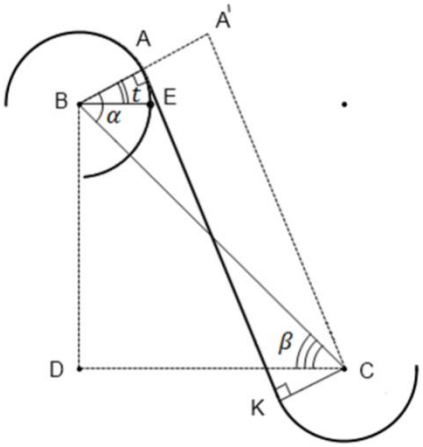


Рисунок 3.8 – Знаходження внутрішньої дотичної до двох кіл

Як і у випадку з дотичною до кола, для знаходження координат точки AA необхідно виконати поворот точки EE на кут tt, який визначається як $t = \alpha - \beta t = \alpha - \beta$ (рисунок 3.8). Внутрішня дотична використовується при побудові ділянок типу 7, 8.

Розглянемо алгоритм знаходження зовнішньої дотичної до двох кіл на прикладі ділянки типу 3 (рисунок 3.9).

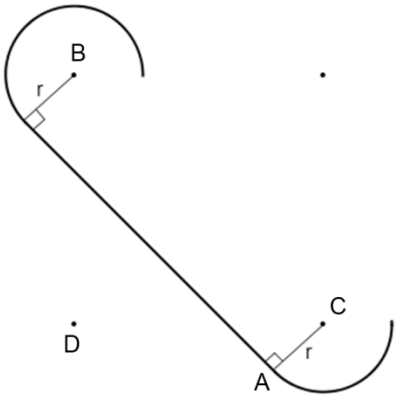


Рисунок 3.9 – Знаходження зовнішньої дотичної до двох кіл на прикладі ділянки типу 3

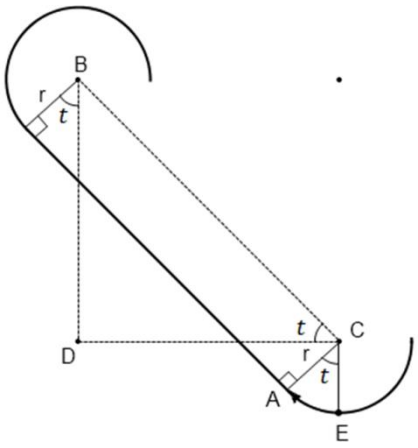


Рисунок 3.10 – Знаходження зовнішньої дотичної до двох кіл

Для знаходження координат точки AA необхідно виконати поворот точки EE на кут tt, який визначається як $t = \arcsin\frac{BD}{BC}t = \arcsin\frac{BD}{BC}$ (рисунок 3.10). Зовнішня дотична використовується при побудові ділянок типу 3, 4.

Загальнийалгоритм побудови графічного запису переплетень представлений на рисунку 3.1.

Рисунок 3.1 – Алгоритм побудови графічного запису

Згідно даної схеми спочатку вводиться графічний запис переплетень (користувачем в графічному інтерфейсі, або дані беруться із файлу) та інформація про рапорт (найменша повторювана ділянка малюнку). Далі визначається кількість рядків аналітичного запису та визначаються зсув гребінки в ii та $i + 1i + 1$ рядках, що дає представлення про тип петлі (відкрита, закрита або протяжка). Відповідно до цих значень будуємо ділянку певного типу (таблиця 3.1).

В кінці циклу виводимо отриманий графічний запис на екран та оцінюємо результати з очікуваним значенням. Якщо результат задовільний завершуємо виконання, в іншому разі редагуємо вхідні дані (цифровий запис) та перебудовуємо графічний запис [8-12].

5. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З ПРОГРАМНОЮ СИСТЕМОЮ

Для встановлення програми необхідно скопіювати папку gmna локальний комп’ютер та запустити виконуючий файл start.exe. Програма містить в собі віртуальну машинну Javata всі необхідні бібліотеки (Java2D, JAXB), тому додатково встановлювати нічого не потрібно.

При успішному запуску з’явиться вікно (рисунок 5.1).

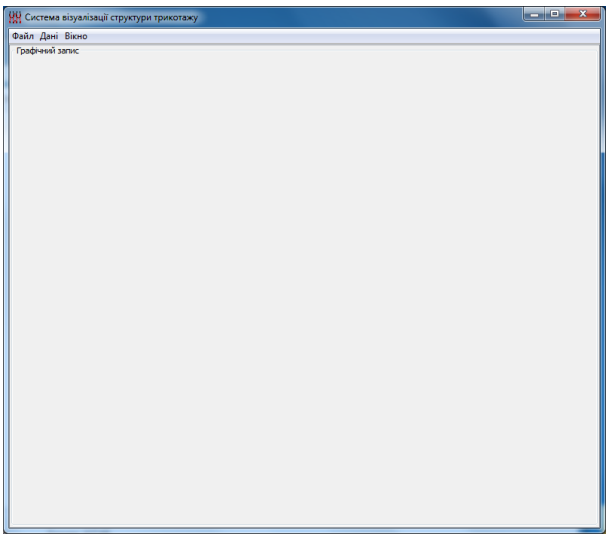


Рисунок 5.1 – Головне вікно програми

Головне меню складається з трьох пунктів: «Файл» – містить команди для створення або відкриття існуючого цифрового запису, «Дані» – для редагування цифрових записів, «Вікно» – для налаштування графічної панелі.

Для створення нового графічного запису необхідно натиснути на меню «Файл» та вибрати пункт «Новий» (рисунок 5.2).

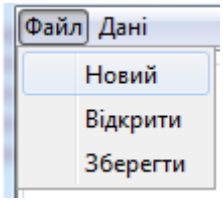


Рисунок 5.2 – Створення нового графічного запису

Після цього з’явиться вікно в якому необхідно задати кількість гребінок (рисунок 5.3).

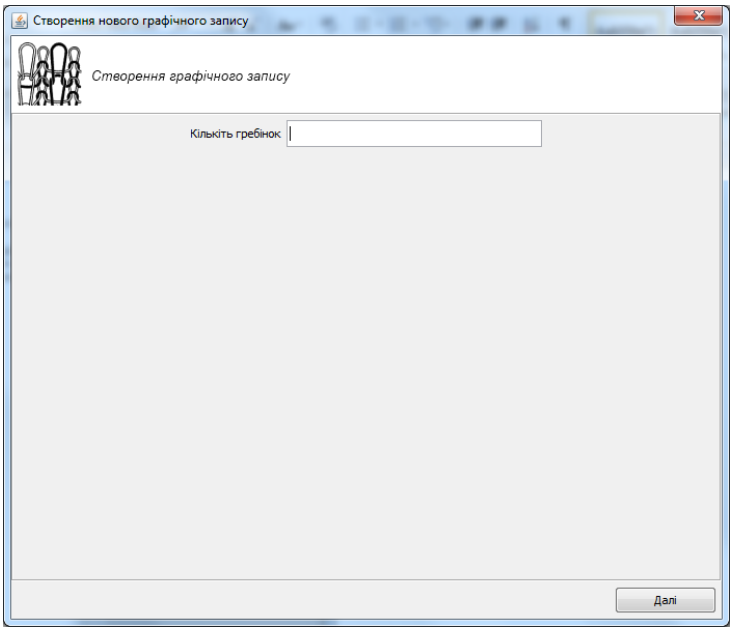


Рисунок 5.3 – Задання кількості гребінок

Далі вказуємо рапорт кладки по висоті та рапорт проборки для кожної гребінки (рисунок 5.4).

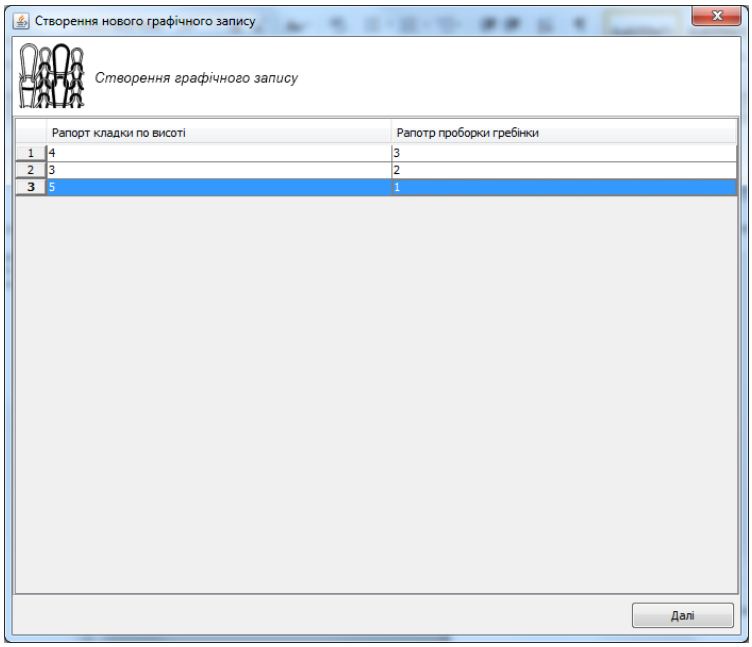


Рисунок 5.4 – Задання рапорту кладки та рапорту проборки

Після успішного введення рапортів переходимо дозадання цифрового запису (рисунок 5.5).

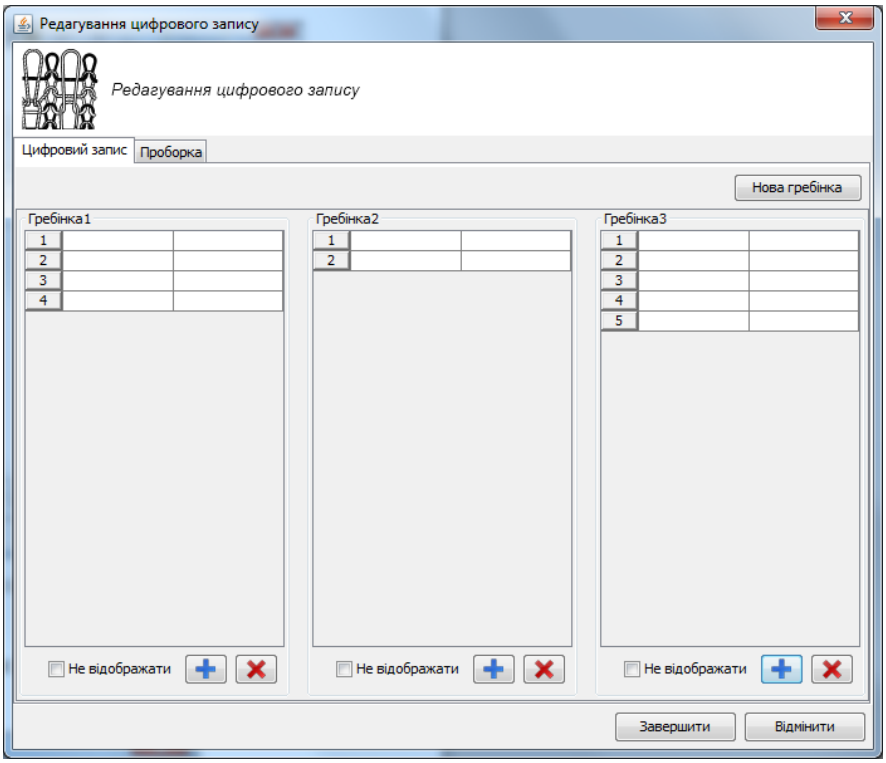


Рисунок 5 – Введення цифрового запису

У разі введення некоректного значення з’явиться вікно з інформацією про його розташування (рисунок 5.6).

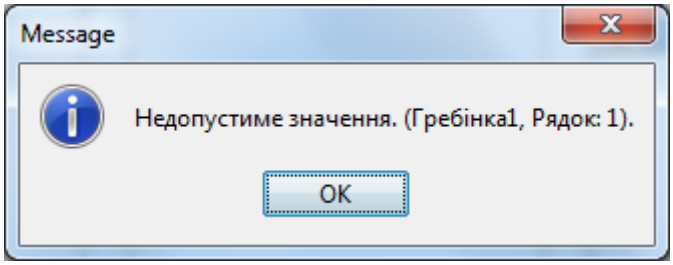


Рисунок 5.6 – Повідомлення у разі введення недопустимого значення

Для введення (редагування) проборки гребінки необхідно натиснути на вкладку «Проборка» (рисунок 5.7).

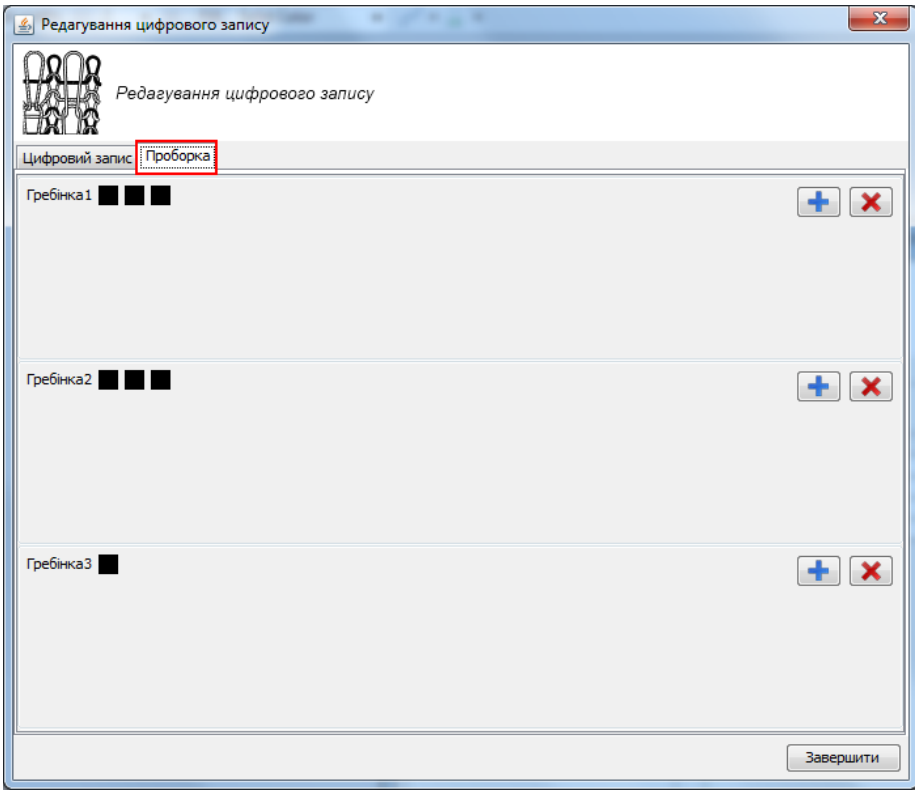


Рисунок 5.7 – Редагування проборки гребінки

Для вибору кольору натискаємо на відповідний квадратик, після чого з’являється вікно (рисунок 5.8).

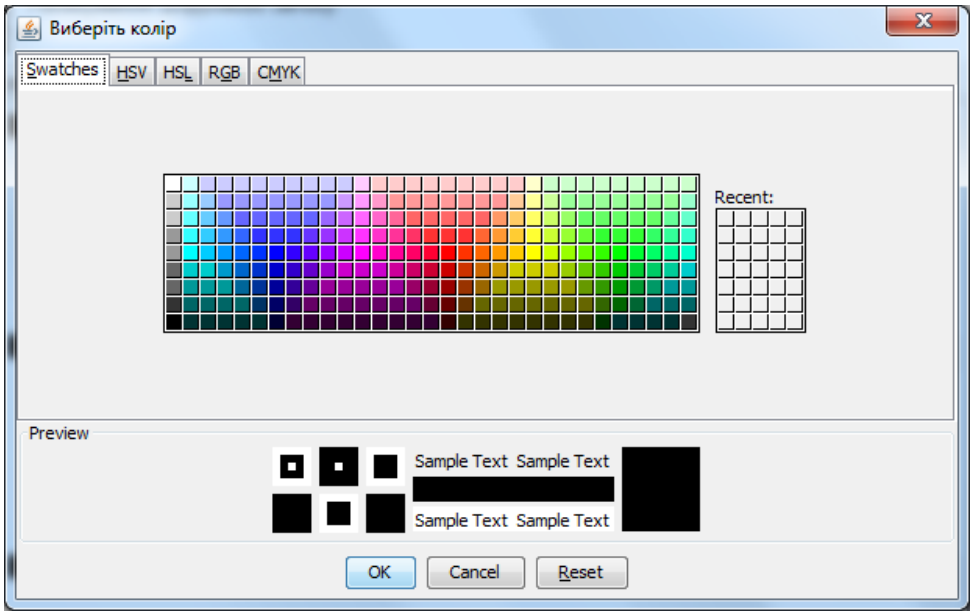


Рисунок 5.8 – Вікно вибору кольору

Вибравши потрібний колір натискаємо кнопку «OK».

Якщо для певної позиції задавати колір не потрібно, то виставляємо прозорий колір. Для цього відповідно до кольорової схеми задаємо максимальну/мінімальну прозорість. Розглянемо на прикладі RGB. Вибираємо вкладку «RGB» та виставляємо значення параметра «alpha» рівним нулю (рисунок 5.9).

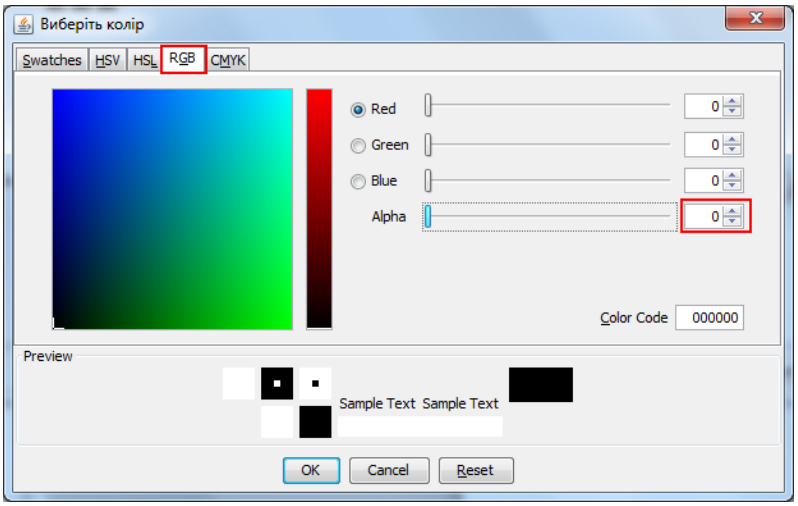


Рисунок 5.9 – Вибір прозорого кольору

Якщо все виконано правильно, то у вибраному квадратику з’явиться хрестик (рисунок 5.10).

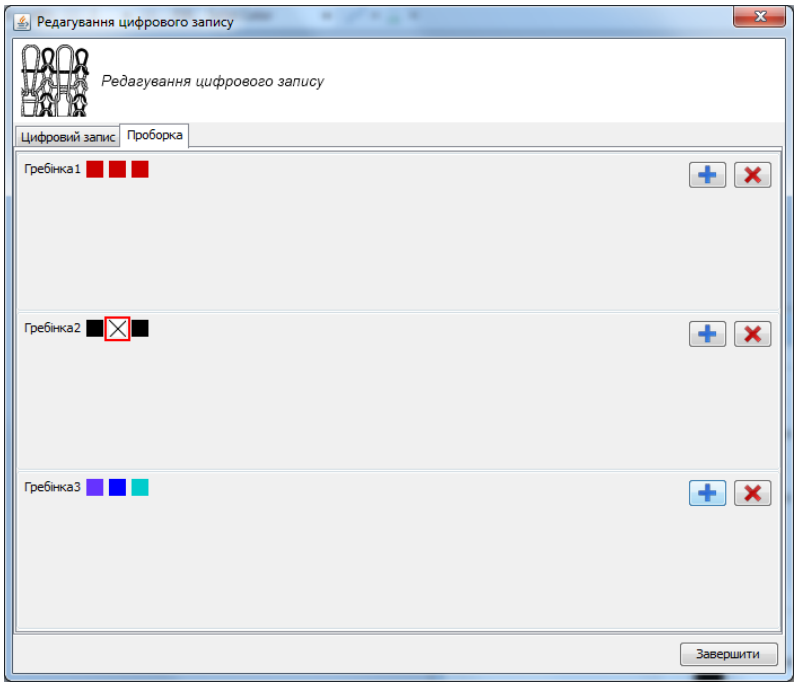


Рисунок 5.10 – Відсутність кольору для певної позиції

Задавши цифровий запис та вибравши потрібні кольори будуємо графічний запис. Для цього натискаємо кнопку «Завершити» в нижній частині вікна вкладки «Цифровий запис» (рисунок 5.11).

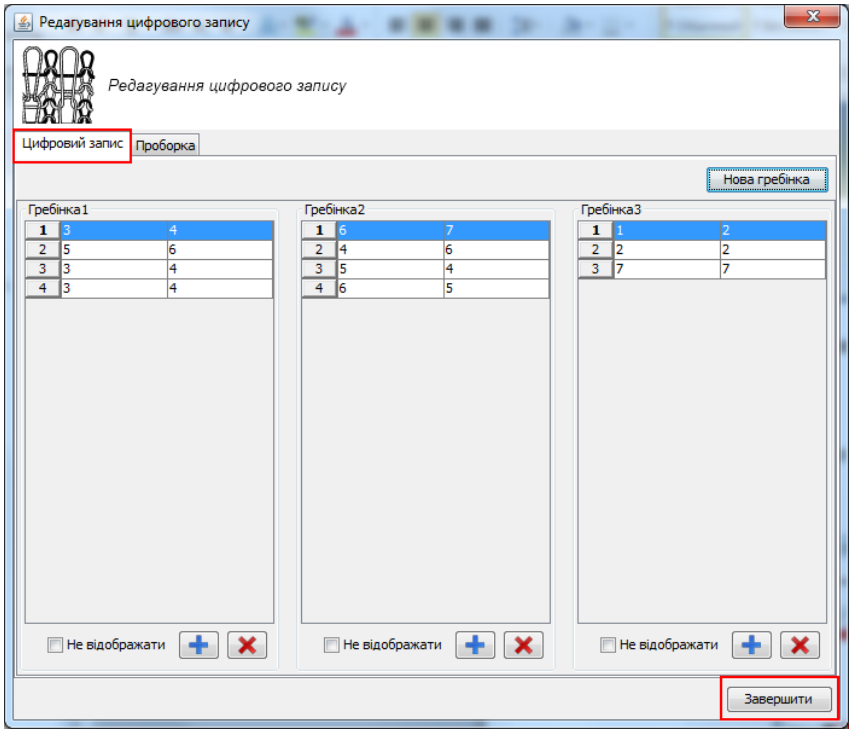


Рисунок 5.11 – Завершення введення даних

Коректно ввівши всі необхідні дані отримаємо графічний запис (рисунок 5.12).

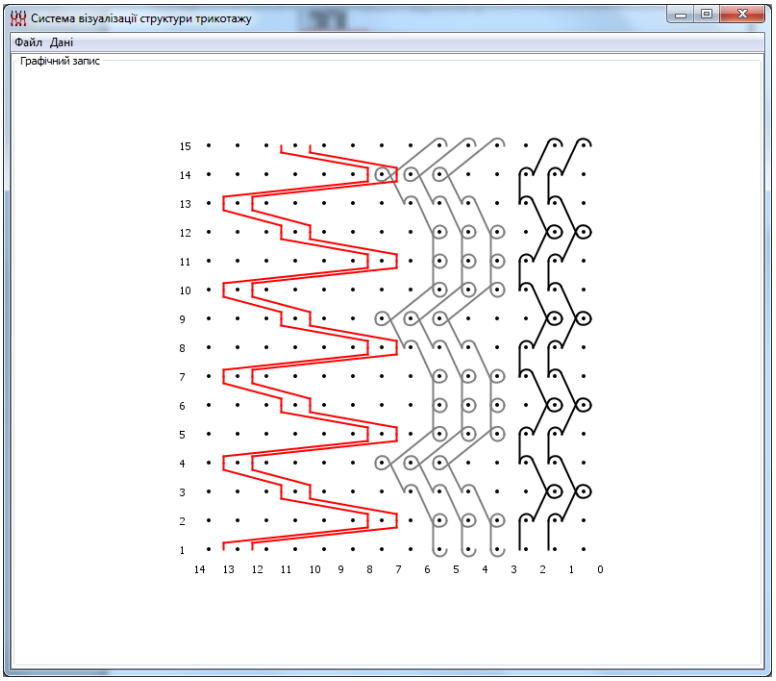


Рисунок 5.12 – Приклад графічного запису

Для редагування введених даних необхідно натиснути на меню «Дані» та вибрати пункт «Редагувати» (рисунок 5.13).

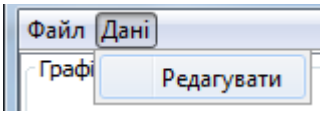


Рисунок 5.13 – Меню для редагуванняцифрового запису

Отримаємо вікно аналогічне вікну при створенні нового графічного запису (рисунок 5.5). Різниця полягає лише в тому, що таблиці для введення цифрового запису уже заповнені (рисунок 5.14).

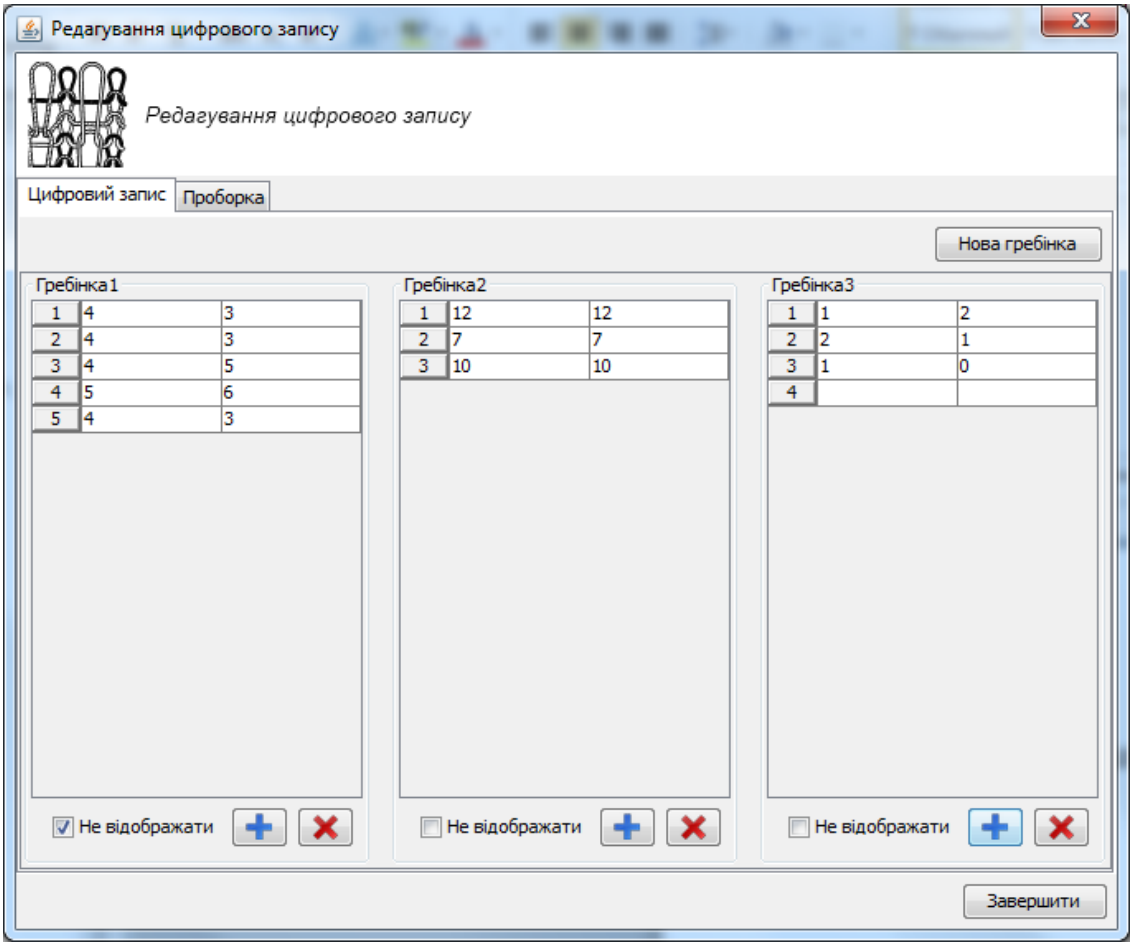


Рисунок 5.14 – Редагування даних

Для додавання рядка таблиці необхідно натиснути кнопку із зображенням плюса (рисунок 5.15).

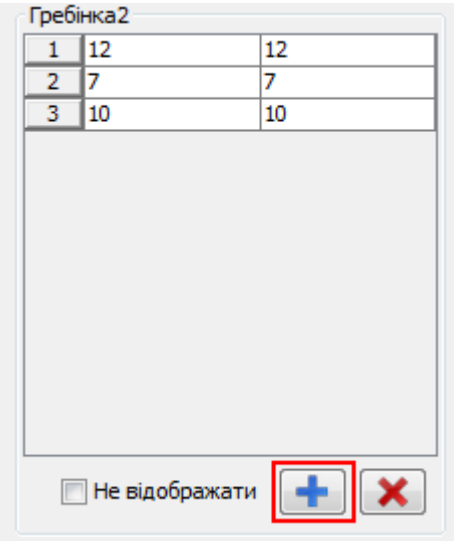


Рисунок 5.15 – Додання рядка цифрового запису

Для видалення рядка (рядків) цифрового запису потрібно вибрати необхідні записи та натиснути кнопку із зображенням хрестика (рисунок 5.16).

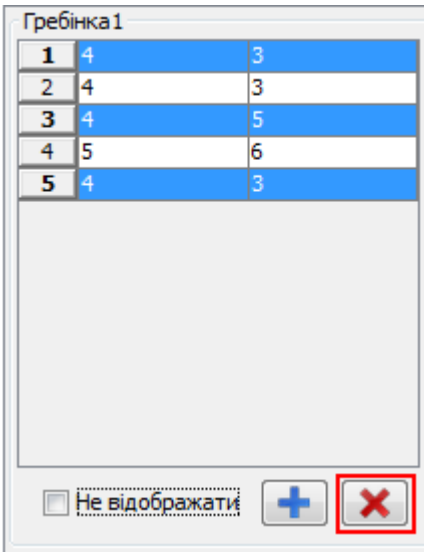


Рисунок 5.16 – Видалення рядків цифрового запису

Щоб не показувати запис певної гребінки ставимо «галочку» на полі «Не відображати» (рисунок 5.17).

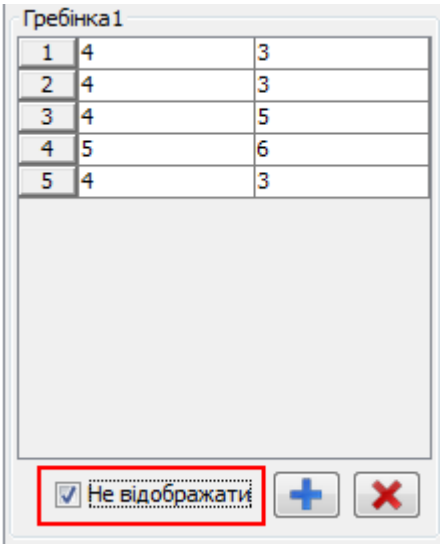


Рисунок 5.17 – Вибір відображення запису певної гребінки

Щоб додати цифровий запис для нової гребінки потрібно натиснути кнопку «Нова гребінка» (рисунок 5.18).

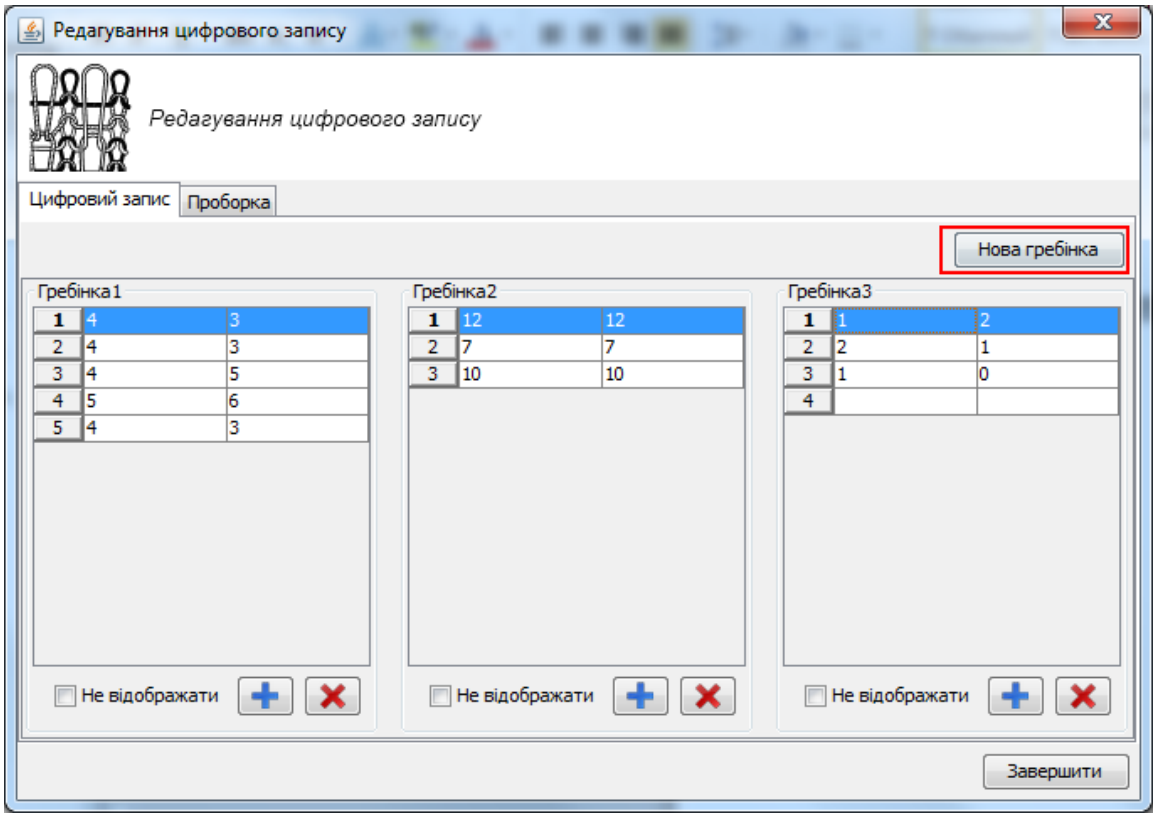


Рисунок 5.18 – Створення цифрового запису для нової гребінки

Для видалення гребінки потрібно видалити всі рядки цифрового запису для даної гребінки.

Для завершення редагування даних натискаємо кнопку «Завершити».

Програмна система передбачає можливість збереження результатів у файл. Для цього натискаємо меню «Файл» та вибираємо пункт «Зберегти» (рисунок 5.19).

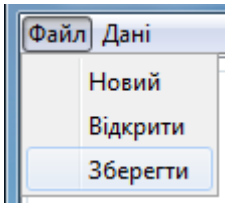


Рисунок 5.19 – Збереження результатів у файл

Після чого з’явиться вікно вибору директорії для збереження (рисунок 20), в якому вказуємо ім’я файлу.

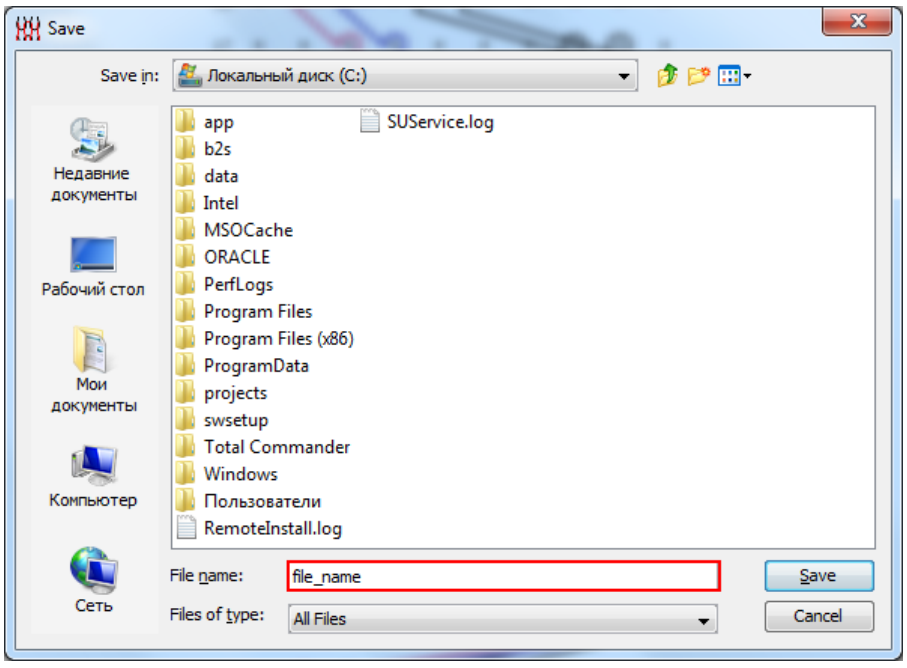


Рисунок 5.20 – Вікно вибору директорії для збереження

Щоб відкрити раніше збережений файл натискаємо меню «Файл» та вибираємо пункт «Відкрити» (рисунок 5.21).

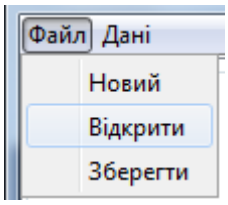


Рисунок 5.21 – Відкриття збереженого файлу

У вікні, що відкрилось вибираємо необхідну директорію і файл та натискаємо кнопку «Open»(рисунок 5.22).

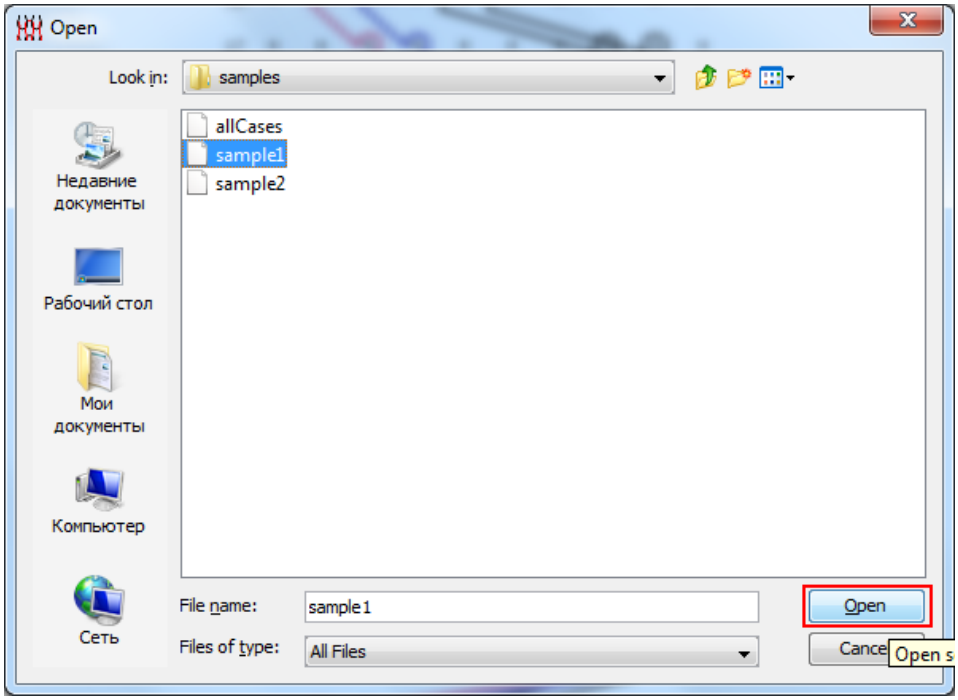


Рисунок 5.22 – Вікно вибору файлу для відкриття

Для зміни налаштувань графічної панелі заходимо в меню «Вікно» та вибираємо пункт «Налаштування» (рисунок 5.23).

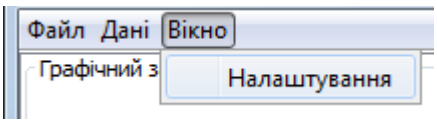


Рисунок 5.23 – Налаштування графічної панелі

У вікні, що відкрилося можна задавати міжголковий інтервал, радіус петлі та товщину ліній. (рисунок 5.24) Для цього пересуваємо відповідний повзунок та натискаємо кнопку «Завершити» для підтвердження змін.

Для виходу з програми натискаємо кнопку закриття вікна. Якщо в цей момент буде відкритий графічний запис, то відкриється діалогове вікно для збереження. Якщо зберігати не потрібно вибираємо «No».

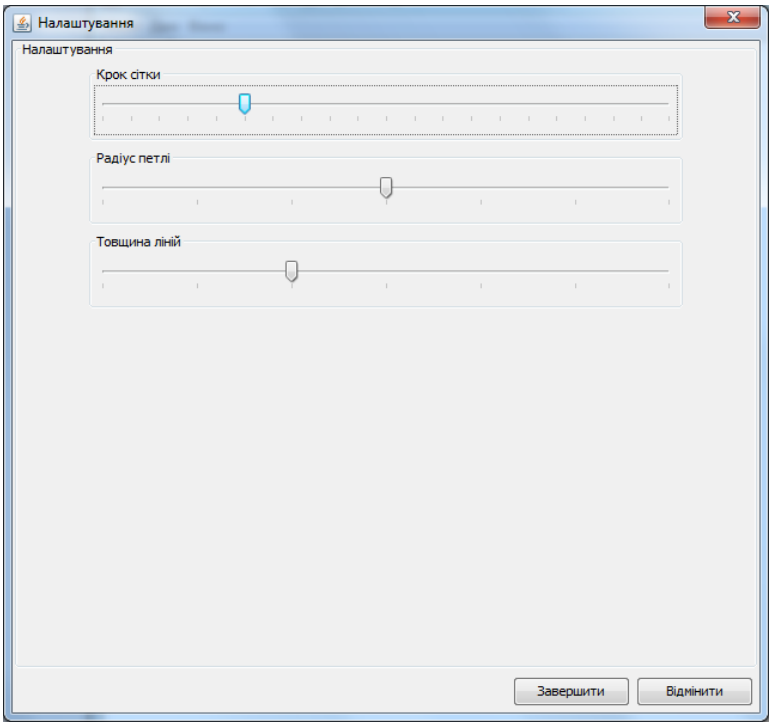


Рисунок 5.24 – Вікно налаштувань

Для уникнення будь-яких складнощів та помилок при роботі з програмним продуктом, а також для отримання коректних результатів необхідно дотримуватись даного сценарію роботи з програмою [24,25].

6. ЕКОНОМІКО-ОРГАНІЗАЦІЙНА ЧАСТИНА

6.1 Розрахунок трудомісткості розробки та впровадження програмного продукту

Трудомісткість розробки та впровадження програмного продукту (ПП) на тему «Система візуалізації структури трикотажу кулірних переплетень з ажурними і рельєфними ефектами» визначається для таких стадій розробки:

- технічне завдання (ТЗ);
- ескізний проект (ЕП);
- технічний проект (ТП);
- робочий проект (РП);
- впровадження (Вп).

Всі ці стадії мають місце тільки при розробці дуже великих і складних ПП; у переважній більшості випадків деякі стадії можуть бути відсутні. Наприклад, може бути відсутньою стадія ЕП, тоді трудомісткість цієї стадії ($T_{ЕП}T_{ЕП}$) враховується в трудомісткості ТП ($T'_{ТП}T'_{ТП}$), що вказано в формулі (6.1):

$$T'_{ТП} = T_{ЕП} + T_{ТП}T'_{ТП} = T_{ЕП} + T_{ТП} \quad (6.1)$$

Стадії ТП і РП можуть об'єднуватися в техноробочий проект ($T_{РП}T_{РП}$), що вказано в формулі (6.2), тоді його трудомісткість складає:

$$T_{ТРП} = 0{,}85\,T_{ТП} + T_{РП}T_{ТРП} = 0{,}85\,T_{ТП} + T_{РП} \quad (6.2)$$

Трудомісткість розробки ПП розраховується на основі типових норм часу на програмування (додатки I, II, III) [26].

На стадіях ТЗ та ЕП трудомісткість в людино-днях визначається в залежності від типу задачі та ступеня новизни за даними додатків I, II.

У випадку участі в роботі розробників постановки задачі та розробників програмного забезпечення їх доля у трудомісткості на стадії технічного завдання становить 0,7 та 0,3, а на стадії ескізного проекту – 0,6 та 0,4.

Вхідні дані представлені в таблиці 6.1.

Таблиця 6.1. Вхідні дані для економіко-організаційного розрахунку

Назва	Значення
Кількість макетів (наборів даних) вхідної інформації; у тому числі:	4
- нормативно довідкова інформація (НДІ)	1
- змінна інформація (ЗІ)	2
- база даних (БД)	1
Кількість різновидів форм вихідної інформації	2
Ступінь новизни групи завдань	«В»
Складність алгоритму	3
Складність організації контролю вхідної і вихідної інформації, яка характеризується такими групами	11/22
Мова програмування	Java
Використання стандартних модулів	30%
Програмний продукт	Не стандартний

На стадіях «Технічний проект» «Робочий проект» і «Впровадження» трудомісткість може бути розрахована залежно від кількості різноманітних форм вхідної і вихідної інформації за формулою (6.3):

$$T_p = a \cdot k^b \cdot l^c, T_p = a \cdot k^b \cdot l^c, \quad (6.3)$$

де *k*–кількість макетів вхідної інформації;

l– кількість різноманітних форм вихідної інформації;

a, b, c – коефіцієнти.

При використанні інформації різних видів розраховується поправочний коефіцієнт за формулою (6.4):

$$K_{П} = \frac{K_1 m + K_2 n + K_3 p}{m + n + p}, \quad (6.4)$$

де K_1, K_2, K_3– поправкові коефіцієнти;

m, n, p – кількість наборів даних відповідно ЗІ, НДІ, БД.

Значення поправкових коефіцієнтів для технічного і робочого проектів приведені в таблиці 6.2.

Таблиця 6.2. Поправкові коефіцієнти для розрахунку трудомісткості робіт

Вид використаної інформації	Ступінь новизни: «В», Група складності алгоритму: 3	
	«Технічний проект»	«Робочий проект»
ЗІ	1,00	1,00
НДІ	0,72	0,48
БД	2,08	0,40

Розрахунок КП для стадії «Технічний проект» за формулою (6.4):

$$K_П = \frac{1.0 \cdot 3 + 0.72 \cdot 1 + 2.08 \cdot 1}{1+3+1} = 1.16$$

Розрахунок КП для стадії «Робочий проект» за формулою (6.4):

$$K_П = \frac{3.00 + 0.48 + 0.4}{1+3+1} = 0.776`$$

При використанні мов програмування високого рівня, норми часу для стадії «Робочий проект» потрібно скоригувати з урахуванням коефіцієнта КМ рівного 1.

Коли при розробці ПП використовуються стандартні модулі норми часу коригують за допомогою коефіцієнта $К_{СТ}К_{СТ}$, значення якого залежить від процентного співвідношення використаних пакетів.

У нашому випадку поправочний коефіцієнт $К_{СТ}К_{СТ}$ = 0,7 при використанні типових програм і стандартних модулів із ступенем використання ПП 30% на стадіях «Робочий проект» та «Впровадження».

При розробці стандартного ПП норму часу слід коригувати за допомогою коефіцієнта $К_{СТ.П}К_{СТ.П}$ рівного 1,2.

Коефіцієнти для розрахунку трудомісткості розробки ПП приведені в таблиці 6.3

Таблиця 6.3. Коефіцієнти для розрахунку трудомісткості розробки ПП на стадіях «Технічний проект», «Робочий проект», «Впровадження»

Комплекс завдань	Стадія розробки	Для розробника постановки задачі			Для розробника програмного забезпечення		
		a	b	c	a	b	c
Система моделювання процесів переносу скалярних полів	ТП	30.04	0,45	0,34	8.34	0,56	0,17
	РП	8.11	0,47	0,49	50.05	0,44	0,42
	Вп	9.10	0,44	0,44	10,89	0,38	0,48

Трудомісткість для стадії «Технічний проект» розраховується за формулою (6.3):

$$T_p(ТП) = 30.04 \cdot 5^{0.45} \cdot 3^{0.34} + 8.34 \cdot 5^{0.56} \cdot 3^{0.17} = 115$$ (людино-дні);

Трудомісткість стадії «Робочий проект» розраховується за формулою (6.3):

$$T_p(РП) = 8.11 \cdot 5^{0.47} \cdot 3^{0.49} + 50.05 \cdot 5^{0.44} \cdot 3^{0.42} = 190$$ (людино-дні).

Трудомісткість стадії «Впровадження» розраховується за формулою (6.3):

$$T_p(ВП) = 9.10 \cdot 5^{0.44} \cdot 3^{0.44} + 10.89 \cdot 5^{0.38} \cdot 3^{0.48} = 64$$ (людино-дні).

Загальна трудомісткість програмування завдань визначається за наступною формулою (6.5):

$$T_з = T_Р \cdot K_П \cdot K_{СК} \cdot K_М \cdot K_{СТ} \cdot K_{СТ.П}. \quad (6.5)$$

Тривалість розробки розраховується за формулою (6.6):

$$T = \frac{T_з}{R \cdot n}, T = \frac{T_з}{R \cdot n}, \quad (6.6)$$

де *R*– кількість людей;

$T_з$– загальна трудомісткість, людино-дні;

N– кількість робочих днів в одному році, днів (n = 250 днів);

T– період розробки, рік.

У таблиці 6.5 приведений розрахунок трудомісткості розробки ПП.

Таблиця 6.4. Розрахунок трудомісткості розробки ПП для всіх стадій

Найменування	Стадії розробки								Всього
	ТЗ	ЕП	ТП		РП		Вп		
			Постановка завдання	Розробка програм	Постановка завдання	Розробка програм	Постановка завдання	Розробка програм	
Трудомісткість, людино-дні	37	77	115		190		64		483
в т.ч. постановка задачі	24	54	89,9	-	29,6	-	29,9	-	-
розробка програм	13	23	-	24,7	-	161,1	-	34,0	-
Поправочні коефіцієнти на: - види інформації Кп	-	-	1,16		0,776		-		-
- складність контролю інформації Кск	-	-	-		1,07		1,07		-
- мова програмування Км.	-	-	-		1		-		-
- використання стандартних модулів Кст	-	-	-		0,6		0,6		-
- розробку стандартних ПП Кст.п	1,2	1,2	1,2		1,2		1,2		-
Скоригована трудомісткість	44	92	160,8		113,6		49,3		460
Трудомісткість з урахуванням об’єднання стадій розробки	44	(92+160,8)*0.85 + 113,6 = 328,5					49,3		422
Кількість працівників	2	4					3		-
Тривалість розробки, років	0,088	0,33					0,0657		0,482

6.2 Кошторис витрат на розробку та впровадження програмного продукту

Кошторис витрат розробляється виконавцем робіт на основі нормативів трудомісткості розробки і впровадження програмного продукту і затверджується замовником робіт або органом, який забезпечує фінансування робіт. Витрати, які включаються в собівартість ПП, групуються відповідно до їх економічного змісту за наведеними нижче статтями [27].

6.2.1 Витрати на оплату праці

До цієї статті витрат належать витрати на виплату основної і додаткової заробітної плати виконавців, обчислені згідно системам оплати праці, які прийняті в організації, включаючи всі види матеріальних і грошових доплат.

Основна заробітна плата розраховується на основі даних про трудомісткість робіт, і посадових окладів основних виконавців. Інформацію про трудомісткість окремих стадій знаходиться в таблиці 6.5.

Таблиця 6.5. Трудомісткість виконання робіт

Стадія	Трудомісткість люд/дні					Всього, люд/дні
	Керівник роботи	Ст. інж.	Програміст	Програміст	Тестувальник	
ТЗ	22	22				44
ТРП	82,125		82,125	82,125	82,125	328,5
Вп	16,43	16,43			16,43	49,3
Всього	120,55	38,43	82,125	82,125	98,55	421,78

Заробітну плату визначають, виходячи з місячних окладів, враховуючи тривалість умовного місяця (21.1 – при 5-денному робочому

тижні). Результати розрахунків основної заробітної плати виконавців знаходиться в таблиці 6.6

Таблиця 6.6. Основна заробітна плата виконавців

Посада	Місячна ставка, грн	Денна заробітна плата, грн	Трудомісткість, люд-дні	Основна заробітна плата, грн
Керівник роботи	10000	437,93	120,55	57132,70
Ст. інж	8000	379,14	38,43	14570,61
Програміст	6000	284,36	82,125	23353,08
Програміст	6000	284,36	82,125	23353,08
Тестувальник	2500	118,48	98,55	11676,20
Всього			421,78	130084,86

Отже, основна заробітна плата становить $З_{осн}$ = 130084,86 грн.

Додаткова заробітна плата (оплата відпусток, премії, одноразові заохочення і тому подібне) розраховується згідно нормативу (формула 6.6), який встановлює підприємство і який в нашому випадку становить 30%:

$$З_{дод} = 0.3 * З_{осн}$$

де $З_{осн}$– основна заробітна плата, грн. (з таблиці 6.7).

$$З_{дод} = 0,3*130084,86=39025,46грн.$$

Сума основної і додаткової заробітної плати складає витрати за статтею «Заробітна плата» або фонд оплати праці:

$$Ф_{зп} = З_{осн} + З_{дод}$$

$$Ф_{зп} = 130084,86+ 39025,46= 169110,32 \text{ грн.}$$

6.2.2 Відрахування на соціальні заходи

До цієї статті належать витрати, здійснювані у порядку та розмірах, передбачених законодавством України, при цьому загальні витрати на страхування становлять 36,86% від фонду оплати праці:

$$В_{заг.страх.} = 0,3686 * Ф_{зп}$$

$$В_{заг.страх.} = 0,3686 * 169110,32= 62334,06 \text{ грн.}$$

У тому числі:

- на обов'язкове державне пенсійне страхування – 33,2%

$$В_{пенс.страх.} = 0,332 * Ф_{зп}$$

$$В_{пенс.страх.} = 0,332 * 169110,32 = 56144,62 \text{ грн.}$$

- на страхування від тимчасової втрати працездатності – 1,4%

$$В_{втр.прац.} = 0,014 * Ф_{зп}$$

$$В_{втр.прац.} = 0,014* 169110,32 = 2367,54 \text{ грн.}$$

- на страхування на випадок безробіття – 1,6%

$$В_{безр.} = 0,016 * Ф_{зп}$$

$$В_{безр.} = 0,016* 169110,32 = 2705,76 \text{ грн.}$$

· на страхування від професійних захворювань та нещасних випадків на виробництві – 0.66%

$$B_{нещ.} = 0{,}0066 * Ф_{зп}$$

$$B_{нещ.} = 0{,}0066 * 169110{,}32 = 1116{,}13 \text{ грн.}$$

6.2.3 Матеріальні витрати

До цієї статті належать витрати на папір, канцелярське приладдя, дискети, картриджі і інші витратні матеріали. Ці витрати в середньому складають 4% від основної заробітної плати:

$$B_{мат.} = 0{,}035 * З_{осн}$$

$$B_{мат.} = 0{,}035*130084{,}86 = 4552{,}97 \text{грн.}$$

6.2.4 Витрати на спеціальне устаткування

Витрати на спеціальне устаткування не передбачені.

6.2.5 Витрати на службові відрядження

Витрати на службові відрядження не передбачені.

6.2.6 Експериментально-виробничі витрати

До експериментально-виробничих витрат відносять оплату машинного часу, пов'язаного з підготовкою і налагодженням програм. Витрати розраховуються, виходячи з кількості годин машинного часу, необхідного для виконання потрібного об'єму обчислюваних робіт по темі і вартості однієї машинної години .

Кількість годин машинного часу для даних завдань розраховується за формулою:

$$T_p = a \cdot k^b \cdot l^c,$$

де k – кількість макетів вхідної інформації;

l – кількість різноманітних форм вихідної інформації;

a, b, c – коефіцієнти.

$$T_p(РП) = 21.06 \cdot 5^{0.42} \cdot 3^{0.56} = 76{,}6$$

Результати розрахунків за цією статтею приведені в таблиці 6.7.

Таблиця 6.7 Витрати на оплату машинного часу

Роботи, які виконуються на ЕОМ	Тривалість виконання робіт, години.	Вартість однієї машино-год, грн.	Сума витрат, грн.
Розробка інтерфейсу	6	5	30
Написання програмного коду	65,6	5	328
Тестування програми	5	5	25
Разом	76,6		383

6.2.7 Накладні витрати

Витрати по цій статті охоплюють витрати на оплату праці управлінського персоналу з нарахуваннями, оплату службових відряджень, консультаційно-інформаційних витрат, ремонт і техобслуговування інших основних фондів, крім ПК, оренду приміщення та ін. Розраховуються за формулою:

$$B_{нак.} = 0{,}5 * Ф_{зп}$$

$$B_{нак} = 169110{,}32 * 0{,}5 = 84555{,}16 \text{ (грн.)}$$

6.2.8 Прибуток

Прибуток визначається у відсотках від суми витрат. Прибуток складає 10% від суми витрат, для даного ПП вона складе:

$$Пр = 0,1 * (Ф_{зп} + В_{заг.страх} + В_{мат} + В_{екс} + В_{нак.})$$

$$Пр = 0,1*(169110,32+62334,06+4552,97+383+84555,16) = 32093,55 \text{ (грн.)}$$

Податок на прибуток (25% від прибутку):

$$Н_{пр} = 0,25 * 32093,55 = 8023,38 \text{ (грн.)}$$

6.2.9 Загальні витрати

Загальні витрати обчислюються як сума витрат і прибутку:

$$З_{заг} = Ф_{зп} + В_{заг.страх.} + В_{мат} + В_{екс} + В_{нак.} + Пр$$

$$З_{заг} = 169110,32 + 62334,06 + 4552,97 + 383 + 84555,16 + 32093,55 = 353029,06 \text{ (грн.)}$$

6.2.10 Податок на додану вартість

ПДВ обчислюється у розмірі 20% від загальних витрат:

$$ПДВ = З_{заг} * 0.2$$

$$ПДВ = 0,2*353029,06 = 70605,81 \text{ (грн.)}$$

6.2.11 Повна вартість роботи, виконаної власними силами

Повна вартість роботи обчислюється як сума загальних витрат і ПДВ:

$$S_{пов} = ПДВ + З_{заг}$$

$$S_{пов} = 70605,81 + 353029,06 = 423634,87 \text{ (грн.)}$$

Загальні підсумки витрат зводяться в кошторис вартості роботи і приведені в таблиці 6.8.

Таблиця 6.8. Кошторис вартості роботи

Стаття витрат	Норматив %	Сума грн.	Питома вага статті, %
1. Заробітна плата, всього		169110,32	39,91
у тому числі: Основна		130084,86	
Додаткова	30%	39025,46	
2.Відрахування на соціальні заходи у тому числі :	36,86%	62334,06	14,7
на обов'язкове державне пенсійне страхування	33.2%	56144,62	
на страхування від тимчасової втрати працездатності	1.4%	2367,54	
на страхування на випадок безробіття	1,6%	2705,76	
на страхування від професійних захворювань та нещасних випадків на виробництві	0,66%	1116,13	
3.Матеріали	3,5%	4552,97	1,07
4.Спеціальне обладнання	Витрати непередбачені		
5.Відрядження			
6.Експериментально-виробничі витрати		383	0,09
7.Накладні витрати	50%	84555,16	19,95
7а.Сума витрат		320935,51	
8.Прибуток	10%	32093,55	7,57
у тому числі податок на прибуток	25%	8023,38	
9.Загальні витрати		353029,06	
10.ПДВ	20%	70605,81	16,67
11.Повна вартість роботи, виконаної власними силами		423634,87	100
12. Договірна ціна		423634,87	

6.3. Визначення економічного ефекту

Річний економічний ефект від використання ПП розраховується по формулі:

$$E_P = E_1 - E_H * K, \tag{6.7}$$

де E – річна економія від функціонування ПП;

E_H – нормативний коефіцієнт ефективності капіталовкладень для програмного забезпечення може становити 0,33;

K – обсяг інвестицій, пов’язаних із створенням і впровадженням ПП.

Кількість інвестицій складається з витрат на розробку й впровадження ПП (відповідно розрахункам, методика якого наведене в п.3) і вартості придбання й монтажу технічних коштів, необхідних для використання ПП.

Річна економія від функціонування ПП може мати три складових:

- економія витрат роботи на розв'язок завдань із застосуванням ПП;
- економія витрат на випуск продукції із застосуванням ПП;
- економія витрат у сфері використання продукції, виготовленої із застосуванням ПП.

У даній роботі враховується тільки перша складова річної економії. Економія витрат роботи на розв'язок завдань із застосуванням ПП розраховується по формулі (6.8).

$$E_1 = \sum_{i=1}^{m} (C_{1i} - C_{2i}) * Qi \tag{6.8}$$

де m – кількість типів завдань, які вирішує дане ПП,

C_{1i} і C_{2i} – витрати на розв’язок і-го типу завдань без застосування ПП і із застосуванням ПП,

Q_i – кількість завдань і-го типу, які розв’язуються на протязі року.

$$C_{ji} = \left[T_{ji} * C_{ji} \left(1 + \frac{H_д}{100}\right) * \left(1 + \frac{H_{від}}{100}\right) * \left(1 + \frac{H_{нв}}{100}\right) + T_{мчji} * C_{мч} \right] * \left(1 + \frac{H_{пдв}}{100}\right), \quad (6.9)$$

де T_{ji} – витрати часу на розв'язок завдань без ПП (j=1) і із застосуванням ПП (j=2), годин,

C_{ji}– погодинна заробітна плата фахівця, який вирішує завдання і-го типу без ПП (j=1) і із застосуванням ПП (j=2), залежно від складності завдання й кваліфікації,

$H_д$, $H_{від}$, $H_{нв}$ і $H_{пдв}$ – нормативи відповідно додаткової зарплати, відрахувань на соціальні заходи, накладних витрат і нормативу ПДВ,

$Тмч_{ji}$ – витрати машинного часу на розв'язок завдань без ПП (j=1) і із застосуванням ПП (j=2), годин,

$C_{мч}$ – вартість однієї години роботи ПП.

Таблиця 6.10 Дані для розрахунку економічного ефекту.

Задача	Без ЕОМ			З ЕОМ			Частота використання за рік
	Витрати часу, дні	Тарифна ЗП, грн.		Витрати часу, год	Зарплата, грн.		
		за місяць	за день		За місяць	За годину	
Проектування системи	5	4000	189,57	3	4000	23,7	75

Отже, загальні витрати на рішення даного завдання без використання програмного продукту, складуть (6.9):

C_{11}= (5*8)*(4000/(21,1*8))*(1+0,3)*(1+0,3686)*(1+0,5)*(1+0,2)=3036 (грн.)

Загальні витрати на рішення данного завдання з використанням створенного програмного продукту, складуть (6.9):

C_{21}=[3*(4000/(21,1*8))*(1+0,3)*(1+0,3686)*(1+0,5)+5*3]*(1+0,2)=245,7 (грн.)

Тоді економія витрат праці з використанням створенного програмного продукту дорівнює (6.8):

$$E_1 = (C_{11} - C_{21}) * 75 = (3036 - 245,7) * 75 = 209272,5 \text{ (грн.)}$$

Тоді річний ефект від використання створенного програмного продукту дорівнює (6.7):

$$E_р = E_1 - E_н * K = 209272,5 - 0,33 * 423634,87 = 69472,99 \text{ (грн.)}$$

6.4Визначення ціни виробу

Серед різних методів ціноутворення на ранніх стадіях проектування ПП досить поширений метод лімітних цін. При цьому визначаються нижня та верхня межа ціни.

6.4.1 Нижня межа ціни

Захищає інтереси розробника ПП і передбачає, що ціна повинна покрити витрати розробника, пов'язані з розробкою та впровадженням ПП, і забезпечити прийнятний рівень рентабельності, не нижчий за той, що він має при впровадженні вже розроблених ПП. Визначається за формулою 6.10:

$$Ц_{НМ} = C_п * (1 + p_н / 100) * (1 + н_{ПДВ} / 100), \quad (6.10)$$

де $C_п$ – повна собівартість розробки ПП,

$н_{ПДВ}$ – норматив ПДВ, %,

$p_н$ – нормативний рівень рентабельності, %.

Рентабельність визначається як відношення чистого прибутку до собівартості ПП:

$$p_н = (32093,55 / 423634,87) * 100 = 7,576\%$$

Тоді нижня межа ціни буде рівна:

$$Ц_{НМ} = 423634,87*(1+7,576/100)*(1+20/100) = 546874,10 \text{ (грн.)}$$

6.4.2 Верхня межа ціни

Захищає інтереси споживача і визначається тією ціною, яку споживач згоден заплатити за продукцію з кращою для нього якістю. Верхня межа ціни може бути обрахована, виходячи з рівня якості ПП $(К_{як})$ або на базі економічного ефекту від запровадження нового ПП. Визначається за формулою:

$$Ц_{ВМ} = С_п + к_е * Е * Т,$$

де $С_п$ – повна собівартість розробки та впровадження ПП,

$к_е$ – доля економічного ефекту, яка залишається в розпорядженні розробника (найчастіше – 0.3),

$Е$ – очікуваний економічний ефект від застосування ПП,

$Т$ – термін користування програмним продуктом.

Тоді верхня межа ціни буде дорівнювати:

$$Ц_{ВМ} = 423634,87 + 0.3 * 69472,99 * 2 = 840472,81 \text{ (грн.)}$$

6.5 Техніко-економічне обґрунтування розробки та вдосконалення програмного продукту на основі функціонально-вартісного аналізу

На основі вивчення організаційно-економічної суті об’єкту, досліджуються основні функції, які буде реалізовано. При цьому будують функціональну модель об’єкту, що представлена на рисунку 6.1.

Головна функція реалізує ціль розробки. Основні функції – ті, заради яких об’єкт створюється. Кожна з основних функцій може мати декілька варіантів реалізації. Вони використовуються для опису морфологічної карти.

За даними таблиці 6.10 проводиться порівняльний аналіз всіх можливих варіантів реалізації функцій ПП. Варіанти, які мають суттєві недоліки, не відповідають умовам технічного завдання, виключаються з подальшого розгляду. На основі цієї карти виконують якісну оцінку варіантів. Для обмеження кількості варіантів, будують позитивно-негативну матрицю.

Рисунок. 6.1 –Функціональна модель візуалізації структури трикотажу кулірних переплетень

В позитивно-негативній матриці оцінюють переваги і недоліки варіантів рішень. Результати аналізу наведено у таблиці 6.10.

Рисунок. 6.2 – Морфологічна карта

Для характеристики ПП, який розробляється використовуються наступні параметри:

- X_1 – об’єм пам’яті на жорсткому диску, який займає встановлений программний продукт, Кб;

- X_2 – потреби в об’ємі оперативної пам’яті, який необхідний для роботи програми, Мб;
- X_3 – час, що витрачається на задання вхідних даних, с;
- X_4 – час, що витрачається на створення коду тестування, с;
- X_5 – наглядність результатів тестування, доля одиниці;
- X_6 – можливість вибору способу відображення результатів, доля одиниці.

Коефіцієнт вагомості варіанта реалізації функції обчислюється за наступною формулою:

$$\varphi_i = \frac{b_i}{\sum_{i=1}^{n} b_i}$$

де b_i – вагомість і-го параметра за результатами оцінок експертів, і обчислюється за формулою:

$$b_i = \sum_{j=1}^{n} a_{ij}$$

де $a_{ij}a_{ij}$ – коефіцієнти переваги, дані усіма експертами по і-му параметру.

Результати розрахунку вагомості параметрів наведені у таблиці 6.11.

Результати розрахунку показників технічного рівня варіантів виконання функцій наведені у таблиці 6.12.

Таблиця 6.10Позитивно-негативна матриця варіантів реалізації функцій

Основна ф-ція	Варіанти реалізації	Переваги	Недоліки
F1	а	Наочність	Необхідність введення великої кількості даних

	б	Не потрібно вводити дані вручну	Ймовірність похибки при завантажені
F2	а	Простий у реалізації	Складність математичної моделі
F3	а	Необхідна характеристика	
	б	Зручно для перегляду	Труднощі при друкуванні
F4	а	У вихідному коді одразу враховані точки зупинки	Необхідно розробити спеціальний механізм
	б	Наочність. Легкість редагування	Великий об’єм

На основі порівняльного аналізу варіантів реалізації функцій за їх перевагами і недоліками і коефіцієнтів вагомості параметрів вибираємо варіанти реалізації функцій:

Таблиця 6.11 Вагомість параметрів

	X1	X2	X3	X4	X5	X6	bi	φi
X1	1	1,5	0,5	0,5	1,5	0,5	5,5	0,171
X2	0,5	1	0,5	1,5	1,5	0	5	0,156
X3	1,5	1,5	1	0,5	0,5	0,5	5,5	0,171
X4	1,5	1,5	0,5	1	0	1	5,5	0,171
X5	0,5	1,5	1,5	0,5	1	1,5	6,5	0,203
X6	0,5	0,5	0	1,5	0,5	1	4	0,125
Сума							32	1

Показники технічного рівня варіантів виконання функцій

1) $F_1а + F_2а + F_3а + F_4а$;

2) $F_1а + F_2а + F_3б + F_4а$;

3) $F_1а + F_2а + F_3а + F_4б$.

Показник рівня якості k-того варіанта реалізації основних функцій виробу розраховується за формулою:

$$K_{mpk} = K_{mp}(F_{1k}) + K_{mp}(F_{2k}) + \ldots + K_{mp}(F_{nk})$$

де $K_{mp}(F_{1k})$ – показник технічного рівня першої функцій k-того варіанту реалізації основних функцій виробу.

$$K_{mp1} = (0{,}684+0{,}855)+(0{,}78+1{,}197)+(1{,}218+0{,}125)+1{,}368 = 6{,}007.$$

$$K_{mp2} = (0{,}684+0{,}855)+(0{,}78+1{,}197)+(0{,}513+0{,}625)+1{,}368 = 5{,}302.$$

$$K_{mp3} = (0{,}684+0{,}855)+(0{,}78+1{,}197)+(1{,}218+0{,}125)+0{,}342 = 5{,}561.$$

Найкращим на етапі функціонального аналізу є варіант, якому відповідає найбільше значення коефіцієнта технічного рівня.

Таблиця 6.12. Результати розрахунку показників технічного рівня варіантів виконання функцій

Основна функція	Варіант реалізації функції	Параметри, які приймають участь в реалізації	Абсолютне значення параметра	Оцінка параметра в балах, bi	Коефіцієнт вагомості. φi	Показник технічного рівня, Ктр(Fi)
F1	а	X1	7,43	4	0,171	0,684
		X4	2,12	5	0,171	0,855
	б	X1	8,23	11	0,171	1,881
		X4	2,5	1	0,171	0,171
		X4	2,34	8	0,171	1,368
F2	а	X2	8,21	5	0,156	0,78
		X4	3,24	7	0,171	1,197
F3	а	X5	4,46	6	0,203	1,218
		X6	8,89	1	0,125	0,125
	б	X3	2,22	3	0,171	0,513
		X6	4,89	5	0,125	0,625
F4	а	X4	2,27	8	0,171	1,368
	б	X4	8,37	2	0,171	0,342

Як видно з розрахунків, кращим є перший варіант, у якого коефіцієнт технічного рівня має максимальне значення рівне 6,007.

Аналіз функцій, які реалізуються програмним продуктом, повинен бути доповнений вартісним аналізом. Для цього розраховуються витрати, які необхідні для розробки ПП, тобто визначається функціонально-необхідна вартість виробу по всіх варіантах реалізації, які досліджуються. Якщо проводиться ФВА робіт, які мають науково-дослідний характер (наприклад, створення програмного продукту), то функціонально-необхідні витрати визначаються шляхом розрахунку кошторису витрат на проведення науково-дослідних робіт. Методика розрахунку кошторису витрат на НДДКР.

Далі наведено укрупнену методику визначення функціонально-необхідних витрат на розробку програмного продукту.

Функціонально необхідні витрати на створення ПП ($C_ф$) визначаються за формулою:

$$C_ф = C_з + C_{від} + C_м + C_п, \quad (6.11)$$

де $C_з$ – оплата праці розробників, грн.;

$C_{від}$ – відрахування на соціальні заходи (37,5% від фонду оплати праці), грн.;

$C_м$ – вартість машинного часу, необхідного для розробки і налагодження ПП, грн.;

$C_н$ – накладні витрати в розмірі 50-150% від витрат на оплату праці, грн.

Опис вхідних даних до формули 6.11:

$C_з$ – основна заробітна плата розробників, становить 130084,86 грн.;

$C_{від}$ – 37,5% від фонду оплати праці 169110,32 грн., становить 63416,37 грн.;

$C_м$ – вартість машинного часу (див. таблицю 6.8, сума витрат грн.), становить 383 грн.;

$C_п$ – накладні витрати в розмірі 50-150% від витрат на оплату праці, грн., витрати складають 84555,16 грн.

Розрахунок згідно формули 6.11:

$$C_ф = C_з + C_{від} + C_м + C_п = 130084{,}86 + 63416{,}37 + 383 + 84555{,}16 = 278439{,}39 \text{ грн.}$$

Вартісний аналіз варіантів реалізації функцій завершується визначенням коефіцієнта техніко-економічного рівня кожного варіанта (K_{mepj}), який розраховується за формулою:

$$K_{mepj} = K_{mpj} / C_{фj}$$

де K_{mepj} – коефіцієнт технічного рівня j-того варіанту;

$C_{фj}$ – величина функціонально-необхідних витрат j-того варіанту, грн.

$$K_{mep1} = 6{,}007/320935{,}51 = 1{,}87 \cdot 10^{-5};$$

$$K_{mep2} = 5{,}302/320935{,}51 = 1{,}65 \cdot 10^{-5};$$

$$K_{mep3} = 5{,}561/320935{,}51 = 1{,}73 \cdot 10^{-5}.$$

Найкращий варіант визначається за максимальним значенням коефіцієнта техніко-економічного рівня.

$$K_{mep(\max)} = 1{,}87 \cdot 10^{-4}.$$

На в бажано встановити область ефективного використання найбільш ефективного і базового варіантів. Для цього розраховують зведені витрати по базовому і новому варіантах (Зв) за формулами (6.12) і (6.13):

$$З_{вб} = (C_б^п + E_н * K_б^п) * Q_р, \qquad (6.12)$$

$$З_{вн} = (С^{п}_{н} + Е_{н} * К^{п}_{н}) * Q_{р} + Е_{н} * К_{ндр}, \quad (6.13)$$

де $З_{вб}$, $З_{вн}$ – річні зведені витрати відповідно по базовому і новому варіантах

$С^{п}_{б}, С^{п}_{н}$ – питомі поточні витрати на одиницю продукції відповідно по базовому і новому варіантах;

$К^{п}_{б}, К^{п}_{н}$ – питомі капітальні витрати відповідно по базовому і новому варіантах, грн.;

$К_{ндр}$– витрати на розробку виробу, визначаються за кошторисом витрат;

$Е_{н}$ – нормативний коефіцієнт ефективності;

$Q_{р}$ – річні продуктивні можливості нового варіанту (наприклад, кількість розрахунків у рік певної задачі за допомогою ПП), шт.

Розрахунок за формулами 6.12 та 6.13:

$С^{п}_{б}$ = (((60*4*8,7*(1+0,3)*(1+0,375)*(1+0,67))+60*2,2)*(1+0,2) =

= (6232,94+132) * 1,2 = 8241,54 грн.;

$С^{п}_{н}$ = (((10*2*8,7*(1+0,3)*(1+0,375)*(1+0,67))+10*2,2)*(1+0,2) =

= (519,41+22)*1,2 = 534,23 грн.;

$$К^{п}_{б} = (Сумар\ витрат) / Q_{р1};$$

$$К^{п}_{н} = (Сумар\ витрат) / Q_{р2};$$

$К^{п}_{б}$ = 320935,51/13 = 24687,34 грн.;

$К^{п}_{н}$ = 320935,51/13 = 24687,34 грн.;

$Е_{нб}$ = 0,33; $Е_{нн}$ = 0,33;

$K_{ндр}$ = 320935,51 грн.;

$Q_{рб}$ = 13 шт., $Q_{рн}$ = 13 шт.;

$З_{вб} = (C_б^п + E_{нб} * K_б^н) * Q_{рб}$ = (8241,54 + 0,33 * 24687,34) * 13 = = 213048,71 (грн.);

$З_{вн} = (C_н^п + E_{нн} * K_н^п) * Q_{рн} + E_н * K_{ндр}$ =

= (534,23 +0,33*24687,34)*13+0,33*320935,51 = 193542,65 (грн.).

Основні техніко-економічні показники базового і нового варіантів зведені у таблицю 6.13.

Таблиця 6.13.Основні техніко-економічні показники базового і нового варіантів

Показник	Одиниці виміру	Варіанти	
		Базовий	Новий
Питомі поточні витрати на одиницю продукції	Гривні	8241,54	534,23
Питомі капітальні витрати	Гривні	24687,34	24687,34
Коефіцієнт ефективності	Безрозмірний	0,33	0,33
Річні продуктивні можливості	Штуки	13	13
Витрати на розробку виробу	Гривні	320935,51	320935,51
Річні зведені витрати	Гривні	213048,71	193542,65

Область ефективного використання базового і нового варіантів зображено на рисунку 6.3.

Отже з проведеного аналізу та рисунку 6.3 видно, що новий варіант виконання функцій доцільно використовувати навіть тоді, коли потреба у вирішенні задач, що вирішує ПП, виникають дуже рідко.

Висновки по 6 розділу

В результаті роботи розраховано трудомісткість розробки та впровадження програмного продукту, розраховано кошторис на розробку ПП за наступними статтями: витрати на оплату праці, відрахування на соціальні заходи, матеріальні витрати, витрати на спеціальне обладнання, витрати на службові відрядження, експериментально-виробничі витрати та накладні витрати. На основі розрахованих витрат визначено прибуток та податок на додану вартість.

Також в роботі визначено економічний ефект від застосування ПП, розраховано діапазон мінімальної та максимальної ціни на розглянутий програмний продукт. Проведене техніко-економічне обґрунтування розробки (вдосконалення) ПП на основі функціонально-вартісного аналізу, в ході якого встановлено найефективніший варіант розробки продукту з найбільшим коефіцієнтом технічного рівня.

7. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

Широке промислове та побутове використання ПК актуалізувало питання охорони праці їхніх користувачів. Найбільш повним нормативним документом щодо забезпечення охорони праці користувачів ПК є «Державні санітарні норми і правила роботи з візуальними дисплейними терміналами (ВДТ) електронно-обчислювальних машин» ДСанПіН 3.3.2.007-98.

Дотримання вимог цих правил може значно знизити наслідки несприятливої дії на працівників шкідливих та небезпечних факторів. Виходячи з цього, роботодавець повинен забезпечити гігієнічні й ергономічні вимоги щодо організації робочих приміщень для експлуатації ВДТ, робочого середовища, робочих місць з ВДТ, режиму праці і відпочинку при роботі з ВДТ тощо, які викладені у Правилах [26].

7.1 Загальна характеристика приміщення

Згідно ДСанПіН 3.3.2.007-98 («Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин») неприпустиме розміщення приміщень для роботи з ЕОМ у підвалах або на цокольних поверхах.

Дане приміщення перебуває на 4 поверсі 6 поверхового будинку, з одним вікном, орієнтованим на захід. Стіни приміщення пофарбовані матовою фарбою світло-салатового кольору, стеля пофарбована у білі кольори, підлогу з паркету коричневого кольору.

Ця робота виконується сидячи, не має потреби у фізичному навантаженні, а тому й відноситься до категорії робіт «Легка – Іа».

В приміщені знаходить 6 робочих місць, які обладнані комп'ютерами з 22 дюймовими моніторами.

Геометричні параметри приміщення:

- довжина a=L=10 м., ширина b=D=6 м., висота H=3 м;
- загальна площа приміщення $S=L*D=60$ м^2;
- загальний об'єм приміщення $V=S*H=168$ м^3;
- кількість працюючих в даному приміщенні N=6;
- площа, що доводиться на одного працюючого $S2=S/N=10$ м^2;
- об'єм, що доводиться на одного працюючого $V2=V/N=28$ м^3.

Вимоги, яким повинні задовольняти робочі приміщення, представленні в ДНАОП 0.00.1.28-10 і ДСанПіН 3.3.2.007-98:

- об'єм приміщення не менше 20 м^3 на людину;
- площа приміщення не менш 6 м^2 на людину.

Отже норми по площі й обсягу на 6 чоловік дотримані.

Порівняння фактичних і нормативних характеристик робочого місця наведено в таблиці 7.1.

Таблиця 7.1. Порівняння характеристик робочого місця

Параметр	Нормативне значення	Фактичне значення
Площа приміщення на одного співробітника	6,0 м^2	10 м^2
Обсяг приміщення на одного співробітника	20 м^3	28 м^3

Таблиця 7.1. Продовження

Ширина робочого столу	600-1400 мм	800 мм
Висота простору для ніг	600 мм	750 мм
Ширина простору для ніг	500 мм	700 мм
Глибина на рівні коліна	450 мм	500 мм
Глибина на рівні витягнутої ноги	650 мм	700 мм
Висота робочого столу	680-800 мм	750 мм
Висота від очей до термінала	400-800 мм	600 мм
Відстань від очей до термінала	>600 мм	700 мм
Глибина стільця	380 мм	380 мм
Ширина стільця	>400 мм	450 мм
Висота стільця	400-500 мм	500 мм
Висота спинки стільця	300±20 мм	320 мм
Ширина спинки стільця	380 мм	380 мм

З таблиці 7.1 видно, що майже всі характеристики робочого місця відповідають нормативним вимогам (ДНАОП 0.00.1.28-10 і ДСанПіН 3.3. 2-007-98).

В приміщенні є одне вікно з подвійними склопакетом, що виходить на захід, а також двері, одні відкриваються назовні – це відповідає вимогам до приміщень, обладнаних ВДТ.

Загальна характеристика робочого приміщення задовольняє вимогам, поставленим в ДНАОП 0.00.1.28-10 [26], ДСанПиН3.3.2.007-98, СНиП 2.01-85 [27].

Робоче місце де перебуває ЕОМ оснащене комп’ютерним столом з підставкою під монітор і висувну стільницю для клавіатури, а також спеціальним відділенням для системного блоку. Стілець - крісло має м’яку

обшивку, спинку з регульованим кутом нахилу й можливість зміни висоти сидіння також підлокітники довжиною 470 мм і шириною 40 мм, з висотою над сидінням 230 мм, і відстанню між ними 500 мм, що повністю відповідає нормативним вимогам і створює максимально комфортні умови роботи з ЕОМ.

Розміщення екрана забезпечує зручне спостереження. Рекомендується нахил монітора на 20^0. Екран і клавіатура повинні розташовуватись на оптимальній відстані. Монітор 22” має розмір 1680х1050 крапок і забезпечує частоту відновлення екрану 59 Гц.

Для клавіатури кут нахилу становить $5\text{-}10^0$, а від краю стола відстань становить від 10 см[26].

7.2 Мікроклімат робочого приміщення

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з’єднанням температури, вологості, швидкості переміщення повітря.

Робота програміста за енерговитратами відноситься до категорії легких робіт Іа, Іб, тому повинні дотримуватися вимоги згідно ДСН 3.3.6.042-99, що наведені у таблиці 7.2.

Таблиця 7.2. Параметрів мікроклімату відповідно до ДСН 3.3.6.042-99

Період року	Оптимальні			Допустимі			Фактичні		
	t°C	W,%	V, м/с	t°C	W,%	V, м/с	t°C	W,%	V, м/с
Теплий	23-25	40-60	0,1	22-28	55	0,1-0,2	23-26	60	0,1
Холодний	22-24	40-60	0,1	21-25	75	Не > 0,1	22-24	50	0,1

Всі розглянуті параметри мікроклімату знаходяться близько до нормативних значень.

На період проведення аналізу приміщення, тобто на середину листопада, температура становила 23°С, а відносна вологість – 50%,

швидкість повітря 0,1 м/с. Ці параметри задовольняють вимогам до робочого місця на холодний період року для 2 розряду робіт (ДСН 3.3.6.042-99 «Мікроклімат виробничих приміщень») [28].

7.3 Виробничий шум

Приміщення перебуває в будинку, розміщеному на достатній відстані від гучної вулиці й поблизу немає гучних об'єктів.

Згідно ДСанПіН 3.3. 2-007-98, ДСН 3.3. 6-037-99 рівень шуму не повинен перевищувати 50 дБА (таблиця 7.3).

Таблиця 7.3. Допустимі рівні звукового тиску і рівні звуку для постійного (непостійного) широкосмугового (тонального) шуму

Характер робіт	Допустимі рівні звукового тиску (*дБ*) в стандартизованих октавних смугах частот із середньогеометричними значеннями (*Гц*)									Допустимий рівень звуку (*дБА*)
	31,5	63	125	250	500	1000	2000	4000	8000	
Програміст	86	71	61	54	49	45	42	40	38	50

Основним джерелом шуму в досліджуваному приміщенні є:

- кондиціонер LG DSB-F22 L=39 дБА, n=1, t=8 год;
- принтеру CanonMP-156 L=37 дБА, n=1, t=0,035 год;
- Вентилятори комп'ютерів L=18дБА, n=4, t=8 год.

Шум у приміщенні ділиться по походженню на механічний (робота принтеру) і аеродинамічний (робота кулерів у ПК).

Розрахунок загального шуму в кімнаті:

$$L_{EKB} = 10\lg\frac{1}{T}\sum_{i=1}^{n} t_i 10^{0,1Li},$$

де L_{EKB} = 39дБА> 50 дБА,

T – час роботи (8 годин),

t_i – час роботи за зміну і-го джерела,

L_i – рівень звуку і-го джерела.

Рівень шуму на робочому місці в момент роботи принтеру не перевищує 39дБА, відповідно до його документації.

Загальний шумL_{EKB} = 39дБАне перевищує допустиме значення 50 дБА.

7.4 Природне освітлення

В кабінеті має місце система бічного природного освітлення – через один віконний проріз в стіні. Вікно приміщене орієнтовано на захід, що не відповідає вимогам ДНАОП 0.00.1.28-10.

Характеристика вікна:

L=1.25 м;

W=2.2 м;

H=L*W=2,75 м^2.

Розрахуємо відношення H/S і зрівняємо його зі світловим коефіцієнтом який приймемо 1/7.

$$2{,}75/60 = 0.045 < 0{,}143$$

Стан природного освітлення повністюне відповідає ДБН В.2.5-28-2006, тому необхідно використати штучне освітлення [28].

7.5 Штучне освітлення

Для штучного освітлення у приміщенні використовуються люмінесцентні лампи, які в порівнянні з лампами розжарювання мають ряд істотних переваг: за спектральним складом світла вони близькі до природного світла; мають підвищену світлову віддачу (у 2-5 разів вищу, ніж у ламп розжарювання); мають триваліший термін служби (до 10 тис. годин).

Розрахунок штучного освітлення проведемо для кімнати площею 60 м^2, ширина якої складає 6м, довжина – 10м, висота – 3м, за методом коефіцієнта використання світлового потоку.

Для визначення потрібної кількості світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

$$F = \frac{ESKZ}{n},$$

де FF – світловий потік, що розраховується, ЛмЛм,

EE – нормована мінімальна освітленість, ЛкЛк;

$\text{E} = 300\,\text{ЛкE} = 300\,\text{Лк}$,

SS – площа освітлюваного приміщення,

ZZ – відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1,1... 1,2, в нашому випадку Z =1,1);

KK – коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників (значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку $\text{К} = 1{,}5\text{К} = 1{,}5$),

nn – коефіцієнт використання світлового потоку, (виражається відношенням світлового потоку, що падає на розрахункову поверхню, до

сумарного потоку всіх ламп, і обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{ст}\rho_{ст}$) і стелі ($\rho_{стелі}\rho_{стелі}$), значення коефіцієнтів дорівнюють $\rho_{ст} = 40\%\rho_{ст} = 40\%$ і $\rho_{стелі} = 60\%,\rho_{стелі} = 60\%$.

Обчислимо індекс приміщення за формулою:

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

де SS – площа приміщення, SS = 60 м^2,

hh – розрахункова висота підвісу, hh = 2,9 м,

AA – ширина приміщення, AA = 6 м,

BB – довжина приміщення, BB = 10 м.

Підставивши значення отримаємо: $i = 0,77i = 0,77$. Знаючи індекс приміщення, за [29] знаходимо $n = 0,22n = 0,22$. Підставимо всі значення у формулу для визначення світлового потоку FF:

$$F = \frac{300 * 1.5 * 60 * 1.1}{0.22} = 4500000 \text{ Лм}$$

Для освітлення використані люмінесцентні лампи типу ЛБ 40-1, світловий потік яких $F = 4320$ Лм$F = 4320$ Лм. Необхідна кількість ламп визначається за формулою:

$$N = \frac{F}{Fл},$$

де NN – визначуване число ламп,

FF – світловий потік, $F = 45000$ Лм$F = 45000$ Лм,

$FлFл$ – світловий потік однієї лампи, $Fл = 4320$ Лм$Fл = 4320$ Лм.

$$N = \frac{45000}{4320} = 11.$$

В приміщенні використовуються світильники типу ЛПО. Кожен світильник комплектується двома лампами. Тобто необхідно використовувати 6 світильників із 12 працюючими лампами в них.

У приміщенні, де аналізувалось робоче місце програміста працює 7 ламп, тому рівень штучного освітлення не задовольняє санітарним нормам.

7.6 Аналіз стану електробезпеки

В приміщенні використовується однофазна електрична сітка з напругою 220 В і частотою 50 Гц. Користувачами електроенергії є 6 комп'ютерів і кондиціонер.

Приміщення відноситься до категорії II - з підвищеною електричною небезпекою, тому що наявність батареї поряд з системними блоками може спричинити небезпеку, у даному випадку в небезпеці знаходитися 2 робочих місця.

ЕОМ, периферійні пристрої ЕОМ, електроприводи і кабелі за ступенем захисту відповідають класу зони за ПВЕ, мають апаратуру захисту від струму короткого замикання і інших аварійних режимів. Застосовуються провідники з важко горючою і негорючою ізоляцією. Лінія електромережі виконана як окрема групова трьох провідникова мережа шляхом прокладки фазового, нульового робочого і нульового захисного провідників. Нульовий захисний провідник використовується для заземлення. Всі прилади, які використовуються підключені до електромережічерез електрофільтри і всі вони з'єднуються в ІБП, який захищає прилади від можливих коротких

замикань в електропроводі або різких перепадів напруги. В приміщенні встановлений аварійний резервний вимикач, який може повністю вимкнути живлення приміщення, окрім освітлення.

В даному приміщенні не достатньо однієї мережі, так як дозволяється підключати менш ніж 5 комп'ютерів в одну мережу. Крім того можливе використання інших електричних приладів під час роботи даного відділу. Головним захистом від ураження людини струмом при пробитті фази на корпус системного блоку є занулення. Для поліпшення стану в офісі необхідно відсунути ті столи, які знаходяться недалеко від батареї, що забезпечить необхідну електробезпеку приміщення, що розглядається, згідно ДНАОП 0.00.1.28-10.

7.7 Безпека в надзвичайних ситуаціях

Інженери,архітектори та конструктори при роботі над проектом беруть до уваги безліч факторів, вплив яких в подальшому відображається на конструкції та структурі тієї чи іншої будівлі. Одним з факторів, який суттєво впливає на планування будівлі є забезпечення безпеки осіб, які перебувають в приміщенні, що включає в себе і випадки при різних надзвичайних ситуаціях (НС) (від найбільш ймовірних, таких як пожежа і до можливості виникнення стихійного лиха). Таким чином при проектуванні повинні бути розраховані та розроблені оптимальні та найбільш ефективні шляхи евакуації із закритого приміщення, що передбачає раціональне розташування основних входів-виходів та наявність відповідної кількості запасних виходів.

7.7.1 Аналіз можливих природних умов

Приміщення знаходиться в географічно безпечній зоні, тому імовірність надзвичайної ситуації природного характеру практично відсутня.

7.7.2 Аналіз можливих виробничих умов

Можливими надзвичайними ситуаціями на робочому місці оператора ПЕОМ можуть бути пожежа і поразка оператора електричним струмом. Джерелом небезпеки є кабелі, розетки, електрообладнання.

Кожне робоче місце оператора ПЕОМ має:

- одну розетку з роз'ємом RJ-45 для підключення комп'ютерного обладнання (на робочому місці IT спеціаліста та співробітника операційних департаментів – три розетки RJ-45);
- одну розетку з роз'ємом RJ-45 для підключення телефонного устаткування;
- одну розетку 220 вольт для підключення живлення до комп'ютерного обладнання (стабілізовані і заземлення) або джерело безперебійного живлення;
- одну розетку 220 вольт для підключення побутових світильників;

Причинами виникнення пожежі в розглянутому приміщенні можуть бути недотримання правил експлуатації виробничого обладнання та електричних пристроїв, несправність елементів ПЕОМ, паління в приміщенні, несправність або відсутність вентиляційної системи, самозаймання речовин і матеріалів, підпал. Крім того, можуть існувати причини електричного характеру – короткі замикання, перевантаження, великі перехідні опори, іскріння і електричні дуги, статичну електрику.

Займання може виникнути при взаємодії горючих речовин, окислювача і джерел запалювання . У розглянутому приміщенні згідно ГОСТ 12.1.044-89 ССБТ «вогнестійкість» до горючих елементів можна віднести дерев'яні столи, тумби та двері, ізоляцію силових кабелів. Облицювання стін, шафи, вікна виконані з негорючих матеріалів.

7.7.3 Пожежна безпека

В даному приміщенні знаходяться горючі і важко горючі речовини і горючі матеріали: меблі, які мають деталі з дерева, пластика і синтетичної тканини, папір, книжки, лінолеум, пластикові корпуси моніторів, тому це приміщення відноситься до категорії пожежонебезпечна В.

Відповідно до правил настроювання електроустаткуванні приміщення, що аналізується, відноситься до пожежонебезпечноїзони класу П-IIа. Для запобігання пожежі в кімнаті є 2 ручних вуглекислотних вогнегасника ОУ-3 (які не можна використовувати в випадку спалахування комп'ютерної техніки). Згідно ППБУ-95 на кожні 20 м^2 приміщення достатньо встановити 2 вогнегасника. Дане приміщення має площу 60 м^2, тому кількість вогнегасників не відповідає нормі. На сходинках знаходиться спеціальний щит пожежного гідранта з належним рукавом.

Приміщення, що розглядається, обладнано датчиками централізованої системи пожежної сигналізації. З працівниками проводиться інструктаж з пожежної і виробничої безпеки.

В розглянутому приміщені є один евакуаційний вихід, що допускається для приміщення категорії «В»,оскільки число робітників 6 людей. В якості заходів по безпеці пожарної безпеки прийняті наступні міри:

- назначений відповідальний за пожежну безпеку приміщення;
- розроблений план евакуації людей (рисунок 7.1);
- люди ознайомлені з правилами пожежної безпеки, з правилами використовування і розміщення приладів пожежогасіння;

Будинок має два виходи – головний і спеціальний евакуаційний вихід. Коридор між виробничими приміщеннями має два виходи на різні сходинки, одні з яких ведуть до головного виходу, а інші – до спеціального

евакуаційного виходу. Сходинки обладнані системою аварійного освітлення.

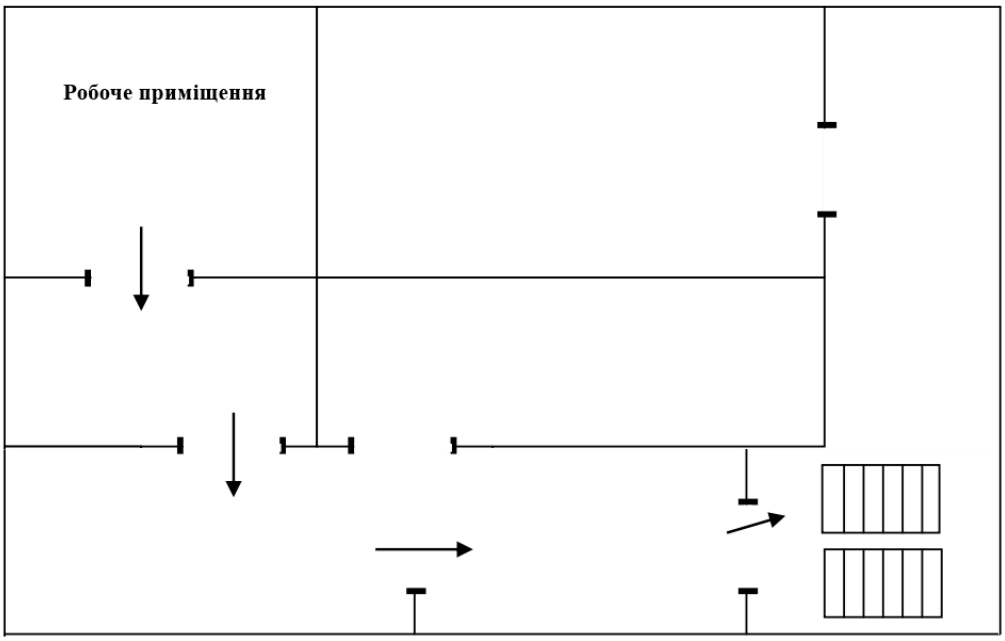


Рисунок 7.1 – План евакуації при пожежі

В приміщенні всі чинники пожежної безпеки задовольняють існуючим правилам і нормам [29].

В результаті аналізу пожежної безпеки було встановлено, що приміщення належить до зони П-IIа і категорії приміщень В відповідно до ОНТП24-86. В приміщенні передбачені заходи, які направлені на мінімізацію ризику виникнення пожежі, але кількість вогнегасників не відповідає нормі.

ВИСНОВКИ

При виконанні поставленої задачі було проведено аналіз існуючих програмних комплексів для моделювання та візуалізації структури трикотажу кулірних та основов’язаних переплетень з ажурними і рельєфними ефектами.

Проведено аналіз існуючих систем моделювання та візуалізації структури переплетень трикотажу та обґрунтовано необхідність розробки системи, що дозволила б використовувати аналітичний (цифровий) запис переплетень для побудови графічного запису, так як існуючі програмні рішення не надають такої можливості в повному обсязі або вимагаються значних затрат часу та високої кваліфікації користувачів системи.

Обґрунтовано вибір використання мови програмування Java у програмному середовищі EclipseJuno, тому що це середовище є повнофункціональним пакетом для швидкої розробки програмнихдодатків, а мова Java має такі переваги як універсальність, ефективність, сумісність з різними платформами, безпека.

Розроблено програмну систему візуалізації структури переплетень трикотажу з наступними основними можливостями: задання кількості гребінок, задання цифрового запису переплетень для кожної гребінки,задання рапорту переплетень по висоті, та рапорту проробки кожної гребінки, відображення отриманих результатів у вигляді графічного запису переплетень, можливість повторного використання графічного запису, за рахунок збереження результатів у файл. Порівнюючи реальний приклад переплетення трикотажу та отримані результати підтверджена коректність моделювання.

Розроблено алгоритм для побудови основних складових графічного запису, що дозволяє формалізувати процес моделювання та візуалізації структури переплетень трикотажу.

Користувачами програмної системи можуть бути фахівці легкої промисловості, студенти та викладачі вузів. Систему можна застосовувати для навчання і підвищення кваліфікації проектувальників, а також в якості навчальних САПР при підготовці інженерів-технологів трикотажного виробництва, а також інженерів-системотехніків – розробників автоматизованих систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кудрявін Л.А., Шустов Є.Ю. Розробка методів візуалізації структури трикотажу при його автоматизованому проектуванні – Київ.: Промтекстиль, 2005.
2. Шалов І.І., Кудрявін Л.О. Основи проектування трикотажного виробництва з елементами САПР – Харків.: Легпром, 2017.
3. Дзюба В.І. Система котирування трикотажу // Сучасні інформаційні та енергозберігаючі технології життєзабезбечення людини: Праці 3-й междунар. наук-практ. конф. СИЭТ-98, 1998 г. – Каменец-Подільский, 1998.
4. Зинов'ева В.О., Морозова Л.В. Про класифікацію основовязаних переплетінь // Технологія текстильної промисловості. – 2002.-№2.
5. Морозова Л.В., Власова О.Ю. Дослідження технології петле утворення тамбурним способом на основов'язальній машині – Харків: Мрія, 2020.
6. Зинов'ева В.О., Морозова Л.В. Про класифікацію кулірних переплетінь // Технологія текстильної промисловості. – 2002.-№3.
7. Мелихова С.В., Морозова Л.В. Трикотаж одношарових похідних основов'язаних переплетінь // Тези доповідей Міжд. наук-техн. конф. «Сучасні наукоємні технології і перспективні матеріали текстальної та легкої промисловості (Прогрес – 2012) – Київ: ІГТА, 2012. – с.64-66.
8. Погорелов А. В. Геометрія: Навч. посібник для ВНЗ. – 2-е вид. – Київ.: Наука, 2003.
9. Роджерс Д. Математичні основи машинної графіки: Пер. з англ. / Д. Роджерс, Дж. Адамс – Київ.: Наука, 2001.
10. Морозова Л.В., Меліхова С.В. Особливості будови трикотажу подвійних похідних двогребеневих основовязаних переплетень // Матеріали IX Міжнародної науково-практичної конференції «Наука-сервіс» – Харків.: ХДУХТ, 2004. – С.41-44.

11. Зінов'єва В.А., Морозова Л.В., Шленнікова О.А. Комплексний графічний метод проектування структури та малюнка основовязаного трикотажу / В.А. Зінов'єва, Л.В. Морозова, О.А. Шленнікова // Вісн. вузів. Технологія текстильної промисловості, 2005. - №3,.
12. Кочеткова О.В., Казначеєва А.А. Розробка концептуальної моделі трикотажу основовязаних переплетень // Вісник Київського національного університету технологій та дизайну. – Київ, 2012. - №4.
13. Морозова Л.В. До питання про класифікацію основовязаних переплетень // Збірник матеріалів науково-технічної конференції «Актуальні проблеми проектування та технології виготовлення матеріалів спеціального призначення» (Текстиль – 2005) – Кировоград КНТУ, 2005.
14. Труєвцев А.В. Класифікація кулірних неповних переплетень / Вісті вузів. Технол. легкої пром-сті. – 1990. - № 2.
15. Бугар К.С., Морозова Л.В. Розробка трикотажу літнього асортименту з ажурним ефектом // Наука-сервіс: XI-а Міжнародна науково-практична конференція. Секція «Проблеми та рішення теоретичних і прикладних задач сервісних технологій». Ч.1: Збірник наукових статей, ГОУВПО «КГУС».- Київ., 2006.
16. Меліхова С.В., Морозова Л.В. Трикотажне полотно для захисного одягу від комах-кровосисів // «Наука-сервіс»: XI Міжнародна науково-практична конференція. Секція «Проблеми та рішення теоретичних і прикладних завдань сервісних технологій». Ч.II: Збірник наукових статей, ГОУВПО «КГУС».- Київ., 2006. - с.161-163.
17. Дзюва В.І. Порівняльний аналіз систем кодування трикотажу / Сучасні інформаційні технології життєзабезпечення людини. СІЕТ-99. – Харків, 1999.
18. Конопальцева Н.М., Морозова Л.В. Індустрія одягу: від сьогодення до майбутнього // Теоретичні та прикладні проблеми сервісу. – 2007. – № 3.

19. Півкіна К.С., Морозова Л.В. Прогнозування нових структур основовязаного трикотажу // Теоретичні та прикладні проблеми сервісу. – 2007. - №3.
20. Кочеткова О.В. Системи автоматизованого проектування трикотажу. Кулірний трикотаж. – Миколаїв: СкіфДизайн, 2021.
21. Кудрявін Л.А. Проектування трикотажу малюнкових та комбінованих переплетень з використанням ЕОМ – Миколаїв: СкіфДизайн, 2021.
22. Морозова Л.В., Бугар К.С. Розробка нових видів ластичних філейних переплетень // Матеріали Всеросійської наукової конференції аспірантів та молодих вчених «Сучасні проблеми сервісу та туризму» Частина II – 2007, с.40-42.
23. Зінов'єва В.А., Морозова Л.В., Меліхова С.В. Трикотажне полотно для захисного одягу від комах, що живляться кров'ю/Вісті вищих навчальних закладів, Технологія текстильної промисловості. – 2008. - №6.
24. Кудрявін Л.А. Автоматизоване проектування основних параметрів трикотажу – Львів.: Легпромбитіздат, 2017.
25. Данчул А.Н. Системотехнічні завдання створення САПР: Практичний посібник – К.: Вища школа, 1990.
26. Методичні вказівки до виконання організаційно-економічного розділу дипломних проектів /За редакцією Чернявського А.Т.. – К.: НТУУ "КПІ", 1999. – 66с.
27. Закон України "Про оподаткування прибутку підприємств". – Відомості Верховної Ради України, 1995р., №4.
28. Закон України "Про податок на додану вартість". – "Голос України" № 82, 15.05.1997 р.
29. Маркович А.А. Порівняльне дослідження методів інтелектуального аналізу даних та сфери їх застосування. / Матеріали XIII Всеукр. наук.-техн. конф., 05 травня 2026 р. Запоріжжя: ТДАТУ, 2026. С. 152-154

ДОДАТКИ

ДОДАТОК 1

Побудова графічного запису структури переплетень трикотажу
Текст програми

```
package com.makalex.diplom.calc;

import static java.lang.Math.max;

import java.awt.Shape;
import java.awt.geom.Arc2D;
import java.awt.geom.Line2D;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

import com.makalex.diplom.model.DataModel;
import com.makalex.diplom.model.G;

public class GraphRecordBuilder {

private int step;
private int r;
private double x0;
private double y0;

private static final int LEFT  = -1;
private static final int RIGHT = 1;
private static final int NONE  = 0;

private static final int CIRCLE_TOP_LEFT  = 2;
private static final int CIRCLE_TOP_RIGHT = 3;
private static final int CIRCLE_BOTTOM_LEFT  = 4;
private static final int CIRCLE_BOTTOM_RIGHT = 5;

public static final int TANGENT_TYPE1 = 6;
public static final int TANGENT_TYPE2 = 7;
```

```
public GraphRecordBuilder(double x0, double y0, int step, int r) {
this.x0 = x0;
this.y0 = y0;
this.step = step;
this.r = r;
}

public GShapes[] build(DataModel data) {
int v1, v2, V12;
int a1, b1, a2, b2;
double x1, y1, x2, y2;

GShapes[] gShapes = new GShapes[data.gCount()];
for (int i = 0; i < data.gCount(); i++) {
        List<Shape> shapes = new ArrayList<>();
        G g = data.getG(i);
        for (int rowIdx = 0; rowIdx < g.getRows().size()-1; rowIdx++) {
                a1 = g.getRow(rowIdx).getA();
                b1 = g.getRow(rowIdx).getB();
                a2 = g.getRow(rowIdx+1).getA();
                b2 = g.getRow(rowIdx+1).getB();

                v1  = getDirection(a1 - b1);
                v2  = getDirection(a2 - b2);
                V12 = getDirection(max(a1, b1) - max(a2, b2));

                x1 = x0-Math.min(a1, b1)*step;
                y1 = step*rowIdx;
                x2 = x0-Math.min(a2, b2)*step;
                y2 = step*(rowIdx+1);

                shapes.addAll(buildPart(v1, v2, V12, x1, y1, x2, y2));
        }

        GShapes gs = new GShapes(g.isVisible());
        for (int j = 0; j < g.getColorCount(); j++) {
                gs.addThreadShapes(
                                new ThreadShapes(
                                copyShapesMovedByX(shapes, -j*step), g.getColor(j) ));
```

```
                                        }
                        gShapes[i] = gs;
                }
                return gShapes;
        }

        private List<Shape> buildPart(
                                int v1, int v2, int V12, double x1, double y1, double x2, double y2) {

        if (v1 != NONE && v2 != NONE) {

        if (v1 == RIGHT && v2 == RIGHT && V12 == RIGHT) {
                x1 += r;
                return part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_RIGHT, TANGENT_TYPE2);
        }
        else if (v1 == LEFT && v2 == LEFT && V12 == LEFT) {
                x1 += -r;
                return part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_LEFT, TANGENT_TYPE1);
        }
        else if (v1 == RIGHT && v2 == RIGHT && (V12 == LEFT || V12 == NONE)) {
                                return part_circle_circle_outer(x1, y1, x2, y2, LEFT);
        }
        else if (v1 == LEFT && v2 == LEFT && (V12 == RIGHT || V12 == NONE)) {
                return part_circle_circle_outer(x1, y1, x2, y2, RIGHT);
        }
        else if (v1 == RIGHT && v2 == LEFT && (V12 == RIGHT || V12 == NONE)) {
                x1 += r;
                return part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_RIGHT, TANGENT_TYPE1);
        }
        else if (v1 == LEFT && v2 == RIGHT && (V12 == LEFT || V12 == NONE)) {
                x1 += -r;
                return part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_LEFT, TANGENT_TYPE2);
        }
        else if (v1 == RIGHT && v2 == LEFT && V12 == LEFT) {
                return part_circle_circle_inner(x1, y1, x2, y2, LEFT);
        }
        else if (v1 == LEFT && v2 == RIGHT && V12 == RIGHT) {
                return part_circle_circle_inner(x1, y1, x2, y2, RIGHT);
        }
```

```
else
        throw new IllegalArgumentException("Can not define method");
}
else if (v1 == NONE && v2 != NONE) {
x1 += step/2;
y1 += r;

List<Shape> s = new ArrayList<>();
if (v2 == RIGHT && V12 == RIGHT) {
        s.addAll(part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_RIGHT, TANGENT_TYPE2));
        s.add(part_protyajka(x1, y1, y1-r));
}
else if (v2 == LEFT && V12 == LEFT) {
        s.addAll(part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_LEFT, TANGENT_TYPE1));
        s.add(part_protyajka(x1, y1, y1-r));
        }
else if (v2 == RIGHT && (V12 == LEFT || V12 == NONE)) {
        s.addAll(part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_LEFT, TANGENT_TYPE2));
        s.add(part_protyajka(x1, y1, y1-r));
}
else if (v2 == LEFT && (V12 == RIGHT || V12 == NONE)) {
        s.addAll(part_point_circle(x1, y1, x2, y2, CIRCLE_TOP_RIGHT, TANGENT_TYPE1));
        s.add(part_protyajka(x1, y1, y1-r));
}
else
        throw new IllegalArgumentException("Can not define method");
        return s;
}
else if (v1 != NONE && v2 == NONE) {
x2 += step/2;
y2 += -r;

List<Shape> s = new ArrayList<>();
if (v1 == RIGHT && V12 == RIGHT) {
        x1 += r;
        s.add(part_point_point(x1, y1, x2, y2));
        s.add(part_protyajka(x2, y2, y2+r));
        }
else if (v1 == LEFT && (V12 == LEFT || V12 == NONE)) {
```

```
            x1 += -r;
            s.add(part_point_point(x1, y1, x2, y2));
            s.add(part_protyajka(x2, y2, y2+r));
}
else if (v1 == RIGHT && (V12  == LEFT || V12 == NONE)) {
s.addAll(part_point_circle(x2, y2, x1, y1, CIRCLE_BOTTOM_RIGHT, TANGENT_TYPE1));
            s.add(part_protyajka(x2, y2, y2+r));
}
else if (v1 == LEFT && V12 == RIGHT) {
            s.addAll(part_point_circle(x2, y2, x1, y1, CIRCLE_BOTTOM_LEFT, TANGENT_TYPE2));
            s.add(part_protyajka(x2, y2, y2+r));
}
else
            throw new IllegalArgumentException("Can not define method");
            return s;
}
else if (v1 == NONE && v2 == NONE) {
x1 += step/2;
x2 += step/2;
y1 += r;
y2 += -r;
List<Shape> s = new ArrayList<>();
s.add(part_point_point(x1, y1, x2, y2));
s.add(part_protyajka(x1, y1, y1-r));
s.add(part_protyajka(x2, y2, y2+r));
return s;
}
else
throw new IllegalArgumentException("Can not define method. v1="+v1+", v2="+v2+", direction="+V12);
}
private int getDirection(int d) {
if (d < 0)
            return LEFT;
else if (d > 0)
            return RIGHT;
else
            return NONE;
}
private Shape part_point_point(double x1, double y1, double x2, double y2) {
```

```
                return new Line2D.Double(x1, y0-y1, x2, y0-y2);
        }
                private Shape part_protyajka(double x, double y1, double y2) {
                return new Line2D.Double(x, y0-y1, x, y0-y2);
        }
                private List<Shape> part_point_circle(double px, double py, double cx, double cy, int
circlePosition, int tangentType) {
                double[] res = Calculation.point_circle(px, py, cx, cy, r, tangentType);
                double tdx = res[0];
                double tdy = res[1];
                double angle = res[2];
                                double startAngle, extent;
                if (tangentType == TANGENT_TYPE1) {
                        if (circlePosition == CIRCLE_TOP_LEFT || circlePosition == CIRCLE_TOP_RIGHT) {
                                startAngle = angle-90;
                                extent = 270-angle;
                        }
                        else if (circlePosition == CIRCLE_BOTTOM_RIGHT) {
                                startAngle = angle-90;
                                extent = 90-angle;
                        }
                        else
                                throw new IllegalArgumentException("Not supported circle position: " +
circlePosition);
                } else if (tangentType == TANGENT_TYPE2) {
                        if (circlePosition == CIRCLE_TOP_LEFT || circlePosition == CIRCLE_TOP_RIGHT) {
                                startAngle = 0;
                                extent = 90+angle;
                        }
                        else if (circlePosition == CIRCLE_BOTTOM_LEFT) {
                                startAngle = 180;
                                extent = 270+angle;
                        }
                        else
                                throw new IllegalArgumentException("Not supported circle position: " +
circlePosition);
                } else
                        throw new IllegalArgumentException("Not supported tangent type: " + tangentType);
```

```
                Line2D line = new Line2D.Double(px, y0-py, cx+tdx, y0-(cy+tdy));
                Arc2D arc = new Arc2D.Double(cx-r, y0-cy-r, 2*r, 2*r, startAngle, extent, Arc2D.OPEN);

                List<Shape> shapes = new ArrayList<>();
                Collections.addAll(shapes, line, arc);
                return shapes;
        }
                private List<Shape> part_circle_circle_outer(double x1, double y1, double x2, double y2, int
interRowDirection) {
                double[] res = Calculation.circle_circle_outer(x1, y1, x2, y2, r);
                double tdx = res[0];
                double tdy = res[1];
                double angle = res[2];

                Line2D line;
                Arc2D arc1, arc2;
                if (interRowDirection == LEFT) {
                        line = new Line2D.Double(x1-tdx, y0-y1+tdy, x2-tdx, y0-y2+tdy);
                        arc1 = new Arc2D.Double(x1-r, y0-y1-r, 2*r, 2*r, 180+angle, 180-angle, Arc2D.OPEN);
                        arc2 = new Arc2D.Double(x2-r, y0-y2-r, 2*r, 2*r, 0, 180+angle, Arc2D.OPEN);
                } else if (interRowDirection == RIGHT) {
                        line = new Line2D.Double(x1+tdx, y0-y1+tdy, x2+tdx, y0-y2+tdy);
                        arc1 = new Arc2D.Double(x1-r, y0-y1-r, 2*r, 2*r, 180, 180-angle, Arc2D.OPEN);
                        arc2 = new Arc2D.Double(x2-r, y0-y2-r, 2*r, 2*r, -angle, 180+angle, Arc2D.OPEN);
                } else
                        throw new IllegalArgumentException("Not supported direction: " +
interRowDirection);

                List<Shape> shapes = new ArrayList<>();
                Collections.addAll(shapes, line, arc1, arc2);
                return shapes;
        }

        private List<Shape> part_circle_circle_inner(double x1, double y1, double x2, double y2, int
interRowDirection) {
                double[] res = Calculation.circle_circle_inner(x1, y1, x2, y2, r);
                double tdx = res[0];
                double tdy = res[1];
                double angle = res[2];
```

```
                        Line2D line;
                Arc2D arc1, arc2;
                if (interRowDirection == LEFT) {
                        line = new Line2D.Double(x1-tdx, y0-y1+tdy, x2+tdx, y0-y2-tdy);
                        arc1 = new Arc2D.Double(x1-r, y0-y1-r, 2*r, 2*r, 180+angle, 180-angle, Arc2D.OPEN);
                        arc2 = new Arc2D.Double(x2-r, y0-y2-r, 2*r, 2*r, angle, 180-angle, Arc2D.OPEN);
                } else if (interRowDirection == RIGHT) {
                        line = new Line2D.Double(x1+tdx, y0-y1+tdy, x2-tdx, y0-y2-tdy);
                        arc1 = new Arc2D.Double(x1-r, y0-y1-r, 2*r, 2*r, 180, 180-angle, Arc2D.OPEN);
                        arc2 = new Arc2D.Double(x2-r, y0-y2-r, 2*r, 2*r, 0, 180-angle, Arc2D.OPEN);
                } else
                        throw new IllegalArgumentException("Not supported direction: " +
interRowDirection);

                List<Shape> shapes = new ArrayList<>();
                Collections.addAll(shapes, line, arc1, arc2);
                return shapes;
        }
        private List<Shape> copyShapesMovedByX(List<Shape> shapes, double moveValue) {
                List<Shape> res = new ArrayList<>();
                for (Shape shape : shapes) {
                        if (shape instanceof Line2D) {
                                Line2D line = (Line2D) shape;
                                res.add(new Line2D.Double(
                                                line.getX1() + moveValue, line.getY1(),
                                                line.getX2() + moveValue, line.getY2()));
                        } else if (shape instanceof Arc2D) {
                                Arc2D arc = (Arc2D)shape;
                                res.add(new Arc2D.Double(
                                                arc.getX() + moveValue, arc.getY(),
                                                arc.getWidth(), arc.getHeight(),
                                                arc.getAngleStart(), arc.getAngleExtent(),
                                                arc.getArcType()));
                        } else
                                throw new IllegalArgumentException("Not supported shape: " +
shape.getClass().getName());
                }
                return res;
        }}
```

ДОДАТОК 2

Опис програмного модуля побудова графічного запису структури переплетень трикотажу

АНОТАЦІЯ

Важливою задачею динамічно розвиваючої економіки є необхідність скорочення термінів та вартості інженерної підготовки виробництва при одночасному підвищенні складності і якості розроблюваних інноваційних проектів. Вирішити дану проблему можливо лише на базі автоматизованого функціонального, конструкторського та технологічного проектування, шляхом розвитку, вдосконалення та впровадження у виробництво відповідних автоматизованих систем.

Розроблений програмний модуль відповідає за побудову графічного запису структури переплетень трикотажу. Отримавши на вході цифровий запис, що представляє собою запис зсувів гребінки, модуль створює масив графічних примітивів, які передаються системі візуалізації графічного запису. Система візуалізації відображає отримані дані у вікні користувача.

1 ЗАГАЛЬНІ ВІДОМОСТІ

Модуль представляє собою програмний додаток, що виконує побудову графічного запису переплетень трикотажу.

Графічний запис представляє собою масив графічних примітивів (прямих, дуг та ін.), які реалізовані за допомогою стандартної бібліотеки для роботи з графікою Java 2D, що надає широкі можливості при роботі з графічними примітивами.

У додатку розглядається програмний модуль GraphRecordBuilder.class з кодом УКР.НТУУ “КПІ” ТЕФ.ТР8108_14С 12-1. Модуль написаний мовою програмування Java. Для його запуску потрібно встановити JVM (Java Virtual Machine).

2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Модуль призначений для побудови графічного запису структури переплетення трикотажу, з можливістю подальшого його відображення на екрані комп’ютера.

Класи, що представляють собою реалізацію модулю відповідають за наступні дії:

- перевірка даних на коректність вводу;
- аналіз переданих даних та визначення відповідної функції, що відповідає за побудову певної ділянки графічного запису;
- передача створеного графічного запису у систему візуалізації.

3 ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Модуль був розроблений в середовищі Eclipse Juno 4.2 на комп’ютері під управлінням операційної системи Windows Seven.

Мінімальні системні вимоги для використання модуля:

- процесор: Intel Pentium IV чи AMD Athlon XP;
- графічна карта: NVIDIA GeForce 7600 512 Мбайт або ATI Radeon X1650 512 Мбайт;
- жорсткий диск ємністю 10 Мбайт;
- оперативна пам’ять: 512 Мбайт;
- монітор, клавіатура, мишка.

4 ВХІДНІ ТА ВИХІДНІ ДАНІ

Вхідними даними є цифровий запис переплетень трикотажу, що вводиться користувачем у вікні редагування або завантажується з файлу, що був раніше збережений та який містить цифровий запис.

Вихідними даними є масив графічних примітивів (які представляють собою графічний запис переплетення), які передаються у систему візуалізації для їх подальшого відображення у панелі користувача.