

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр
(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ
ІНФОРМАЦІЙНОЇ ПІДТРИМКИ ІНЖЕНЕРА З ПИТАНЬ
ОБСЛУГОВУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
SOFTWARE DEVELOPMENT FOR THE INFORMATIONAL SUPPORT OF A
SOFTWARE MAINTENANCE ENGINEER

Виконав: студент(ка) 4 курсу, групи КНТ-118сп
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Строков В.М.
(прізвище та ініціали)

Керівник Степаненко О.О.
(прізвище та ініціали)

Рецензент Скрупский С.Ю.
(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Строкова Володимира Михайловича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програмного забезпечення для інформаційної підтримки інженера з питань обслуговування програмного забезпечення

Software Development for the Informational Support of a Software Maintenance Engineer

керівник проєкту (роботи) Степаненко Олександр Олексійович, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року 103

2. Строк подання студентом проєкту (роботи) 17 травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми та постановка завдань дослідження. 2. Аналіз вимог до програмного забезпечення. 3. Проєктування та тестування. 4. Керівництво програміста. 5. Керівництво користувача 6. Додатки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Степаненко О.О., доцент		
Нормоконтролер	Камінська Ж.К., асистент		

7. Дата видачі завдання “ 15 ” лютого 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тижні	Розділ 1
3	Провести аналіз способів та методів оновлення програмного забезпечення;	4–5 тижні	Розділ 2
4	Провести обґрунтування вибору методу рішення та інструментів;	6 тиждень	Розділ 3
5	Побудувати структурно-функціональну схему, алгоритми роботи програми	7–8 тижні	Розділ 3
6	Розробити керівництво програміста	9 тиждень	Розділ 4
7	Розробити керівництво користувача	10 тиждень	Розділ 5
8	Оформлення висновків	11 тиждень	Висновки
9	Оформлення пояснювальної записки та документів до неї.	12-13 тиждень	Додатки
10	Нормоконтроль та рецензування.	14–15 тижні	
11	Захист роботи.	16 тиждень	

Студент(ка)

Керівник проєкту (роботи)

(підпис) Строков В.М.
(прізвище та ініціали)

(підпис) Степаненко О.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи
бакалавра: 70 с., 2 табл., 28 рис., 2 дод., 13 джерел.

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, АЛГОРИТМ, ЕЛЕКТРОННА ПОШТА, ЛОГІН, ПРОТОКОЛ, ПАРОЛЬ

Об'єкт дослідження – методи проведення оновлення програмних засобів.

Предмет – способи інформаційної підтримки інженера.

Метою даної роботи є розроблення програмного забезпечення інформаційної підтримки інженера з питань обслуговування програмного забезпечення.

Матеріали, методи та технічні засоби: імперативна, структурована, об'єктно-орієнтована, високорівнева мова програмування зі суворою статичною типізацією змінних Delphi, персональний комп'ютер з процесором Intel Core 7 під управлінням операційної системи Microsoft Windows 10.

Результати. Створено програмну реалізацію програмного засобу інформаційної підтримки інженера з питань обслуговування програмного забезпечення.

Висновки. Розроблено програмну реалізацію програмного забезпечення інформаційної підтримки інженера з питань обслуговування програмного забезпечення, шляхом проведення своєчасного оновлення програмних засобів.

Галузь використання – бухгалтерія, юриспруденція, сфера обслуговування.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 70 pages, 2 tables, 28 figures, 2 appendixes, 13 sources.

SOFTWARE, ALGORITHM, E-MAIL, LOGIN, PROTOCOL, PASSWORD

Object of study is the methods of software updates.

Subject of study are ways of information support of the engineer.

The purpose of the work is to develop software information support engineer for software maintenance.

Materials, methods, and tools: imperative, structured, object-oriented, high-level programming language with strict static typing of Delphi variables, personal computer with Intel Core 7 processor running Microsoft Windows 10 operating system.

Results. The software implementation of the software tool for information support of the software maintenance engineer has been created.

Conclusions. Developed software implementation of software support information engineer for software maintenance, by conducting timely software updates.

The field of use – accounting, jurisprudence, service.

ЗМІСТ

Перелік скорочень та умовних познач	7
Вступ	8
1 Аналіз проблеми та постановка завдань дослідження	10
1.1 Оновлення програмного забезпечення	10
1.2 Існуючі підходи до інформаційної підтримки інженера	12
1.2.1 IObit Software Updater	12
1.2.2 SUMo	13
1.2.3 DriverPack Solution	14
2 Аналіз вимог до програмного забезпечення	16
2.1 Опис вибору методу рішення	16
2.2 Обґрунтування вибору мови програмування	19
3 Проєктування та тестування	23
3.1 Структурно-функціональна схема	23
3.2 Розробка та опис алгоритму	27
4 Керівництво програміста	34
5 Керівництво користувача	39
Висновок	45
Перелік джерел посилання	46
Додаток А Текст програми	48
Додаток Б Слайди презентації	62

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД	база даних;
ІТ	інформаційні технології;
ПЗ	програмні засоби;
ПК	персональний комп'ютер;
СУБД	системи управління базами даних;
SQL	Structured Query Language

ВСТУП

У діяльності підприємства, що представляє собою комплекс великого числа пов'язаних і взаємодіючих підрозділів, оперативна передача достовірної інформації є першорядним і неодмінним фактором нормального функціонування даної структури. За ефективність взаємодії підрозділів відповідає відділ інформаційних технологій, також виникає необхідність підтримувати достовірність інформації, що знаходиться в базах даних системних комплексів програм, що обумовлюється своєчасним оновленням програмного забезпечення у всіх відділах.

Важливими програмними продуктами, які потребують постійного оновлення є, наприклад, єдиний державний реєстр юридичних осіб, фізичних осіб – підприємців та громадських формувань (ЄДР) У даному реєстрі містяться відомості і документи про юридичну особу, а також єдиний державний реєстр індивідуальних підприємців. Ведення даних реєстрів здійснює реєструючий орган в порядку, встановленому Кабінетом Міністрів України.

Оновлення даних програм, що містять додавання до баз даних, висилаються клієнтам по електронній пошті, із зазначенням дати, до якої має відбутися оновлення.

Після того, як клієнт оновив застосування, він повинен відправити лист по електронній пошті. Звіт про факт поновлення приходить в поштову програму MS Outlook в відділ «Інформаційних технологій» сервіс-інженеру.

Дана дипломна робота присвячена розробці програмного забезпечення для інформаційної підтримки інженера з питань обслуговування програмного забезпечення, який здійснює контроль за своєчасністю оновлення програмного забезпечення на підприємстві. Програма повинна здійснювати перевірку надісланих електронних листів. У пояснювальній записці до дипломної роботи міститься докладний виклад всіх кроків, пройдених на етапі розробки програми: постановка задачі, що

включає в себе аналіз пропонованих до системи вимог; вибір методу рішення, що описує застосовувані при вирішенні задачі математичні методи; опис алгоритму, складеного для вирішення поставленого завдання; обґрунтування вибору мови програмування, що використовується для реалізації складеного алгоритму; опис програми, апаратні вимоги та вимоги до ПЗ, для коректної роботи.

Метою даної роботи є розробка програмного забезпечення для інформаційної підтримки інженера з питань обслуговування програмного забезпечення. Програма призначена для збору, зберігання і обробки інформації про надіслані електронні листи про оновлення.

Для досягнення поставленої мети були поставлені наступні завдання:

- провести аналіз способів та методів оновлення програмного забезпечення;
- провести аналіз існуючих підходів для інформаційної підтримки інженера;
- провести обґрунтування вибору методу рішення та інструментів;
- побудувати структурно-функціональну схему;
- побудувати алгоритми роботи програми;
- розробити програму для інформаційної підтримки інженера з питань обслуговування програмного забезпечення.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Оновлення програмного забезпечення

Будь-які програмні продукти, через певний час після установки повинні оновлюватися. Оновлення випускаються для:

- усунення вразливості в системі безпеки;
- забезпечення сумісності з новими комплектуючими комп'ютерів;
- оптимізації програмного коду;
- оновлення нормативних актів та законодавчих баз;
- підвищення продуктивності всієї системи.

Якщо відповідні служби та деякі програмні компоненти системи, які пов'язані з роботою служби оновлення, потребують оновлення для коректного функціонування, то ці оновлення повинні встановлюватися перед перевіркою, завантаженням та встановленням будь-яких інших оновлень [1]. Ці обов'язкові оновлення виправляють помилки, а також забезпечують удосконалення та підтримують сумісність з серверами корпорації Майкрософт, що підтримують роботу служб. Якщо служба оновлення вимкнена, то отримувати оновлення для програмних продуктів буде неможливо.

При завантаженні і встановленні оновлень різних програм в залежності від типу підключення до Інтернету може стягуватися стандартна плата за місцеві або міжміські телефонні перемови, а також плата за користування Інтернетом [1,2].

Всі оновлення поділяються на [3]:

- важливі оновлення забезпечують істотні переваги в безпеці, конфіденційності та надійності. Їх слід встановлювати відразу ж, як тільки вони стають доступні;
- рекомендовані оновлення можуть усувати менш істотні проблеми або робити використання комп'ютера більш зручним. Хоча ці оновлення не призначені для усунення істотних недоліків у роботі комп'ютера або

програмного забезпечення, їх встановлення може призвести до помітних поліпшень;

- необов'язкові оновлення. до яких відносяться драйвери або нове програмне забезпечення, що робить використання комп'ютера більш зручним. До решти оновлень можна віднести всі оновлення, які не входять до складу важливих, рекомендованих або необов'язкових оновлень.

Перевірка відповідності оновлень ПЗ вимогам може бути запущена в наступних випадках (в тому числі і за розкладом перевірки).

- розклад перевірки наявності оновлень програмного забезпечення. Сканування для перевірки відповідності вимогам оновлень програмного забезпечення починається відповідно до розкладу сканування, налаштованому в параметрах агента клієнта оновлень програмного забезпечення;

- дія на сторінці властивостей ПЗ . Користувач може запустити дію «цикл перевірки оновлень програм» або «цикл оцінки розгортання оновлень програм»;

- розклад повторної оцінки розгортання. Оцінка і перевірка розгортання на відповідність оновлень вимогам запускається за розкладом повторної перевірки розгортання, налаштованому в параметрах агента клієнта оновлень.;

- перед завантаженням файлів оновлень . Коли клієнтський комп'ютер отримує політику призначення для нового обов'язкового розгортання, агент клієнта оновлень ПЗ буде оновлюватися ПЗ в локальний кеш клієнта. Перед завантаженням файлів оновлень ПЗ агент клієнта оновлень ПЗ запускає перевірку, щоб переконатися, що це оновлення ПЗ;

- перед встановленням оновлення програмного забезпечення. Безпосередньо перед встановленням оновлення агент клієнта оновлень ПЗ запускає перевірку, щоб переконатися, що це оновлення ПЗ потрібно;

- після установки оновлення програмного забезпечення. Відразу після завершення установки оновлення ПЗ агент клієнта оновлень ПЗ

запускає перевірку, щоб переконатися, що це оновлення більше не потрібно, і створює нове повідомлення про стан, яке вказує, що дане оновлення ПЗ встановлено. Якщо установка завершена, але потрібен перезапуск, повідомлення про стан буде вказувати, що клієнтський комп'ютер очікує перезапуску.

– після перезапуску системи. Якщо клієнтський комп'ютер очікує перезапуску для завершення установки оновлення ПЗ, агент клієнта оновлень ПЗ запускає перевірку після перезапуску, щоб переконатися, що це оновлення більше не потрібно, і створює нове повідомлення про стан, яке вказує, що дане оновлення встановлено. [3]

1.2 Існуючі підходи до інформаційної підтримки інженера

Кожен користувач має на своєму комп'ютері не один десяток встановлених програм. Але все на ПК починає застарівати згодом, якщо встановлене програмне забезпечення, інсталювані програми не будуть своєчасно оновлюватися. Вручну здійснювати контроль, пошук оновлень практично неможливо. Тому програми для автоматичного і своєчасного оновлення наявного на комп'ютері програмного забезпечення до нових версій можна віднести до самого потрібного і корисного ПЗ [1]. В першу чергу, необхідність оновлення стосується операційної системи, програм безпеки та ін.

1.2.1 IObit Software Updater

Є одним з найсучасніших і просунутих засобів, що дозволяє легко і швидко оновити програми в ОС. Варто відзначити корисну функцію створення контрольних точок відновлення, завдяки чому у користувача завжди буде можливість припинити будь-які несприятливі зміни, що зачіпають працездатність і продуктивність системи [4].

Розробники реалізували відмінний інструмент повідомлень з системного трея. Тепер вам не потрібно витратити багато часу на самостійні пошуки того чи іншого оновлення. Утиліта сама сканує комп'ютер, щоб знайти всі встановлені додатки. Інтегрований менеджер пошуку моментально попередить користувача про вихід новинки. Підтримується діагностика і управління автозавантаженням.

Основні переваги:

- широкий спектр налаштувань;
- резервне копіювання даних;
- інтеграція в меню;
- моніторинг в реальному часі;
- сумісність з Windows 7, 8, 10;
- покрокова інструкція для початківців;
- опція перезавантаження ПК (при необхідності);
- блокування небажаного програмного забезпечення;
- можливість запуску разом зі стартом ОС;
- автоматичне і ручне оновлення застосунків, як оновлювати можна вибрати;
- дозволяє завантажувати і встановлювати оновлення одним клацанням миші, запустити процес оновлення всіх компонентів одночасно.

Остання версія ПЗ не підтримує XP та Vista.

1.2.2 SUMo

SUMo – спеціальна програма, створена для установки оновлень на різне ПЗ, що зберігається у ОС. Як і інші програми автоматичного оновлення програм поширюється на безкоштовній основі, не вимоглива до ресурсів ПК [5].

Після завершення процедури моніторингу, в окремому вікні виводить зручний список доступних продуктів. Натисніть одну кнопку і ПЗ почне

оновлюватися. У верхній частині інтерфейсу відображаються рекомендації, за допомогою яких можна зрозуміти, які саме утиліти варто оновлювати, а які ні [5].

Головні переваги Software Update Monitor:

- невеликий розмір файлу встановлення;
- дозволяє дізнатися версію і виробника;
- сумісність з Windows XP, Vista і вище;
- підтримується опція ігнорування оновлень непотрібного ПЗ.

1.2.3 DriverPack Solution

DriverPack Solution в автоматичному режимі встановить драйвера під певну операційну систему для користувача комп'ютера. Остання версія визнана універсальним інсталяційним пакетом драйверів, який завжди можна носити з собою на будь-якому зовнішньому накопичувачі [6].

Можливості DriverPack Solution

- система Windows не потребує резервної копії драйверів пристроїв на комп'ютері;
- вбудована база доповнень і пакета драйверів не потребує підключення до мережі;
- є режим автоматичного оновлення драйверів і системних компонентів;
- можливість установки додаткового програмного забезпечення на ОС Windows;
- вбудований багатомовний інтерфейс і наявність власної інтеграції різних драйверів в програмну базу.
- всі актуальні драйвера оптимізовані з 64-розрядними ОС;
- драйвер пак можна встановлювати безпосередньо з флешки, а також dvd і hdd.

Недоліки:

- оновлення драйверів на офіційному сайті проводиться досить рідко;
- для просунутих користувачів і системних адміністраторів краще використовувати експертний режим і проводити діагностику стану ПК в ньому.

1.3 Висновки до розділу

В розділі було досліджено проблему вчасного оновлення програмного забезпечення на підприємствах.

Також було розглянуто та проаналізовано існуючі підходи до інформаційної підтримки інженера з питань програмного забезпечення.

Провівши аналіз недоліків та існуючих підходів до інформаційної підтримки інженера з питань обслуговування програмного забезпечення було прийнято рішення створення власного продукту.

2 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Опис вибору методу рішення

Для вирішення поставленого завдання потрібно провести аналіз вимог і визначення специфікацій.

Вхідна інформація:

- дата початку;
- дата кінця;
- адреса електронної пошти;
- домен;
- логін;
- пароль;
- протокол отримання пошти.

Вихідна інформація:

- автор листа;
- адреса пошти;
- тема листа;
- сервер;
- текст повідомлення;
- прикріплений файл;
- дата листа.

Вихідні дані надходять до програми шляхом ручного введення в спеціально структурований файл або шляхом введення в саму програму.

Розділяється символ – «|»

Дата початку – дата, до якої має бути здійснене оновлення клієнтом (вказується в листі).

Дата кінця – сьогоднішня дата.

Структура файлу вхідних даних представлена в табл. 2.1

Таблиця 2.1 – Структура файлу вхідних даних

Адреса пошти
Адреса сервера вхідної пошти (сервер POP3)
Логін
пароль
Протокол отримання пошти

Адреса пошти – адреса електронної поштової «ящика», куди приходять листи з даними про оновлення. Адреса «ящика» електронної пошти складається з двох частин, розділених символом «@». Перша частина являє собою ідентифікатор поштового «ящика» (зазвичай називають «login» або «реєстраційний запис»). Друга частина – ідентифікатор Інтернет – провайдера, який надав адресу (також називають «доменом»).

Адреса сервера вхідної пошти (сервер POP3) – пишеться як: pop. "Домен". (Якщо, наприклад, поштова скринька зареєстрований на сервері www.gmail.com, то адреса сервера вхідної пошти буде pop.gmail.com).

Логін – перша частина електронної пошти («реєстраційний запис»).

Пароль – пароль на вхід в електронний «ящик».

Протокол (protocol) – стандарт, який визначає поведінку функціональних блоків при передачі даних. Протокол задається набором правил взаємодії функціональних блоків, розташованих на одному рівні; реалізується однією або групою програм; описує синтаксис повідомлення, імена елементів даних, операції керування та стану. (POP3 (Post Office Protocol version 3) – протокол, що підтримує як прийом, так і відправку повідомлень електронної пошти.)

Вихідний звіт «Вхідні електронні листи» планується представити у вигляді табл. 2 2

Автор листа	Електронна адреса пошти	Тема листа	Сервер	Текст повідомлення	Прикріплений файл	Дата оновлення

Таблиця 2.2 – Табличний вигляд звіту «Вхідні електронні листи»

Автор листа – при відправці пошти ім'я, прізвище автора листа прикріплюється автоматично.

Електронна адреса пошти – адреса електронної поштової «ящика», з якого надійшов лист з даними про оновлення.

Тема листа – відображається та тема листа, яку відправник вказав в листі.

Сервер – Поштовий сервер [Postal Server], що забезпечує прийом / передачу і маршрутизацію електронних листів користувачів в Інтернеті.

Розробляється програма повинна здійснювати:

- введення і формування вихідних даних в програму;
- додавання, коригування та видалення інформації;
- встановлення зв'язку з поштовим сервером;
- перевірку наявності вхідних повідомлень, а так само їх фільтрацію;
- висновок даних на екран і на тверді копії;
- блокування некоректних дій з боку користувача при роботі з системою.

Програмна система повинна мати дружній графічний інтерфейс.

Після аналізу вхідних і вихідних даних, функцій, які буде повинна виконувати програма, можна скласти структурно-функціональну схему програми. За принципом модульного структурування функції можуть бути реалізовані у вигляді окремих модулів. В основу модульного структурування покладено принцип функціональної декомпозиції, при якій структура системи описується в термінах ієрархії її функцій і передачі інформації між окремими функціональними елементами. Логічна модель проєктованої програми повинна містити наступні модулі:

Введення (додавання) і формування – здійснює введення (додавання) вихідних даних (діапазон дат, адреса електронної пошти, адреса сервера вхідної пошти (сервер POP3), логін, пароль, протокол отримання пошти);

Редагування – внесення зміни і видалення, додавання наявних даних;

Облік вхідних електронних листів – здійснює фільтрацію пошти за датою, зазначеної в листі;

Збереження – здійснює збереження вихідних даних в файл.

Висновок – здійснює висновок даних на екран і на тверді копії.

2.2 Обґрунтування вибору мови програмування

Дану програму слід реалізувати на мові високого рівня. Розглянемо переваги і недоліки деяких з них.

Delphi – це продукт Borland International для швидкого створення додатків. Високопродуктивний інструмент візуального побудови додатків включає в себе справжній компілятор коду і надає засоби візуального програмування, кілька схожі на ті, що можна виявити в Microsoft Visual Basic або в інших інструментах візуального проєктування. В основі Delphi лежить мова Object Pascal, який є розширенням об'єктно-орієнтованої мови Pascal. У Delphi також входять локальний SQL-сервер, генератори звітів, бібліотеки візуальних компонентів, і інші інструменти, необхідні для того, щоб відчувати себе абсолютно впевненим при професійній розробці інформаційних систем або просто програм для Windows-середовища [7].

Перш за все, Delphi призначений для професійних розробників, бажаючих дуже швидко розробляти програми в архітектурі клієнт-сервер. Delphi виробляє невеликі за розмірами (до 15-30 Кбайт) високоефективні виконавчі модулі (.exe і .dll), тому в Delphi повинні бути перш за все зацікавлені ті, хто розробляє продукти на продаж. З іншого боку невеликі за розмірами і швидко виконувані модулі означають, що вимоги до клієнтських

робочих місць істотно знижуються - це має важливе значення і для кінцевих користувачів [8].

Переваги Delphi в порівнянні з аналогічними програмними продуктами.

- швидкість розробки програми;
- висока продуктивність розробленого додатка;
- низькі вимоги розробленого додатка до ресурсів комп'ютера;
- нарощення за рахунок вбудовування нових компонент і інструментів в середу Delphi;
- можливість розробки нових компонент і інструментів власними засобами Delphi (існуючі компоненти і інструменти доступні в початкових кодах);
- вдале опрацювання ієрархії об'єктів.

Система програмування Delphi розрахована на програмування різних додатків і надає велику кількість компонентів для цього.

До того ж роботодавців цікавить перш за все швидкість і якість створення програм, а ці характеристики може забезпечити тільки середовище візуального проєктування, здатна взяти на себе значні обсяги рутинної роботи по підготовці додатків, а також узгодити діяльність групи постановників, кодувальників та тестерів. Можливості Delphi повністю відповідають цим вимогам і підходять для створення систем будь-якої складності [9].

Visual Basic легко використовувати для вивчення і використання. Він дозволяє розробляти не тільки прості програми з графічним інтерфейсом, але і створювати складні додатки. Програмування в VB являє собою поєднання візуальних компонентів і контролів, визначення атрибутів і дій для компонентів, написання додаткового коду для розширення функціональних можливостей. Певні за замовчуванням значення і дії для компонентів дозволяють створити просту програму без написання коду програмістом. Ранні версії мали певні проблеми з продуктивністю програм, але з

використанням сучасних комп'ютерів і компіляцією власного коду дана проблема стала менш значимою.

Компіляція програм була введена в Visual Basic 5, але при цьому виконуються програми досі вимагають наявності певних бібліотек для запуску. Ці бібліотеки включені в Windows, починаючи з Windows 2000, але для більш ранніх версій Windows вони повинні поставлятися разом з програмою [9,10].

Форми створюються за допомогою технології «перетягни та кинь» (drag and drop). Елементи управління просто розміщуються на формі (наприклад, поля введення, кнопки, та ін.). Елементи управління мають свої атрибути і обробники подій. Багато атрибутів можуть бути змінені під час виконання програми, що дозволяє створювати програми, динамічно реагують на дії користувача. Basic може створити виконувані програми (EXE файли), елементи керування ActiveX, бібліотеки DLL, але, перш за все, використовується для розробки Windows додатків. Діалогові вікна з обмеженими функціональними можливостями можуть використовуватися для забезпечення підказок. Елементи управління забезпечують основні функціональні можливості програми, а програміст може розширити логіку програми за допомогою обробників подій. Наприклад, випадаючий список автоматично відображає список і дозволяє користувачеві вибрати елемент. Оброблювач подій викликається для виконання додаткового коду в залежності від обраного елемента [11].

Мова використовує довідкову інформацію для збору сміття, має велику бібліотеку сервісних об'єктів, об'єктно-орієнтовану розробку. На відміну від багатьох інших мов програмування, Visual Basic не залежить від реєстру. Порівняння рядків залежить від реєстра, але може бути виконано без урахування реєстру.

Компілятор Visual Basic поставляється разом з іншими мовами Visual Studio, але обмеження в інтегрованому середовищі розробки не дозволяють створення деяких видів додатків.

Система програмування Microsoft Visual C ++ являє собою реалізацію середовища розробки для поширеного мови системного програмування C ++, виконану компанією Microsoft. Ця система програмування в даний час побудована у вигляді інтегрованого середовища розробки, що включає в себе всі необхідні засоби для розробки результуючих програм, орієнтованих на виконання під керуванням ОС типу Microsoft Windows різних версій. Основу системи програмування Microsoft Visual C ++ складає бібліотека класів MFC (Microsoft foundation classes)[12]. Дана бібліотека реалізована у вигляді класів C ++ всі основні органи управління і інтерфейсу ОС. Також до її складу входять класи, що забезпечують розробку додатків для архітектури клієнт-сервер і трирівневої архітектури (в сучасних версіях бібліотеки). Система програмування Microsoft Visual C ++ дозволяє розробляти будь-які програми, які працюють в середовищі ОС типу Microsoft Windows, в тому числі серверні або клієнтські результуючі програми, які здійснюють взаємодію між собою по одній із зазначених вище архітектур [13].

2.3 Висновки до розділу

Для вирішення поставленого завдання було проведено аналіз вимог і визначення специфікацій. Після аналізу було зроблено висновок, що оптимально використовувати для подання інформаційних матеріалів мову Delphi, яка є мовою високого рівня і дозволяє швидко і ефективно створювати додатки.

Для розробки програми "Облік вхідних електронних листів" було обрано інтегроване середовище програмування Delphi 9 версії для програмування додатків ОС Windows.

3 ПРОЄКТУВАННЯ ТА ТЕСТУВАННЯ

3.1 Структурно-функціональна схема

Взаємозв'язок модулів програми відображає структурно-функціональна схема, представлена на рис. 3.1.

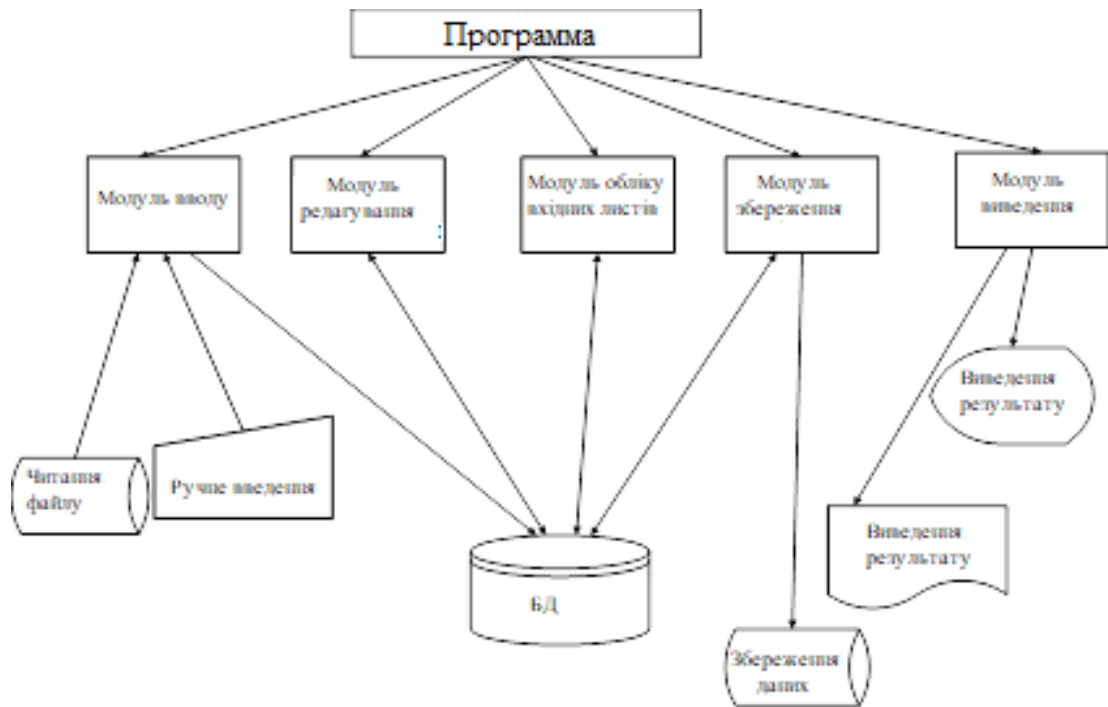


Рисунок 3.1. – Схема структурно-функціональних модулів програми «Облік вхідних електронних листів»

Зі схеми видно, що введення (додавання) і формування даних здійснюється за допомогою модуля введення (додавання) і формування, причому введення вихідних даних здійснюється як через файл, так і через ручне введення в програму. Вхідні дані надходять в модуль редагування і пошуку для здійснення функцій зміни, видалення даних. Модуль обліку по введених даних встановлює зв'язок з поштовим сервером, здійснює фільтрацію і передає дані в модуль збереження. Модуль збереження здійснює функцію збереження даних в файл (MS Excel). Модуль виведення здійснює вивід необхідної інформації на екран і на тверді копії.

Після аналізу роботи всіх модулів можна скласти діаграму потоків даних програми «Облік вхідних електронних листів». На основі структурно-функціональної схеми побудуємо контекстну діаграму (рис. 3.2).

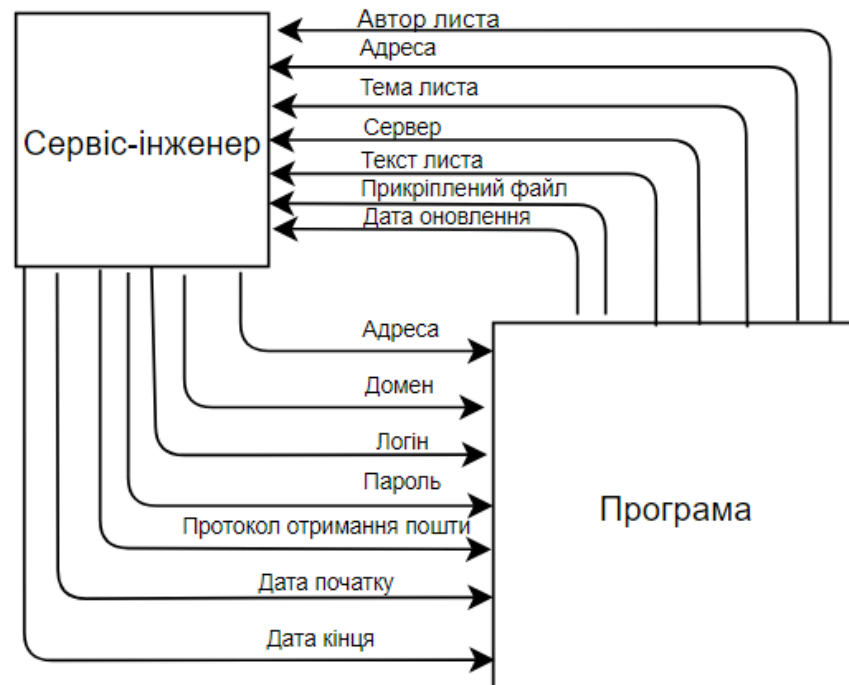


Рисунок 3.2 - Контекстна діаграма програми «Облік вхідних листів»

Діаграма потоків даних (DFD) представлена на рис. 3.3.

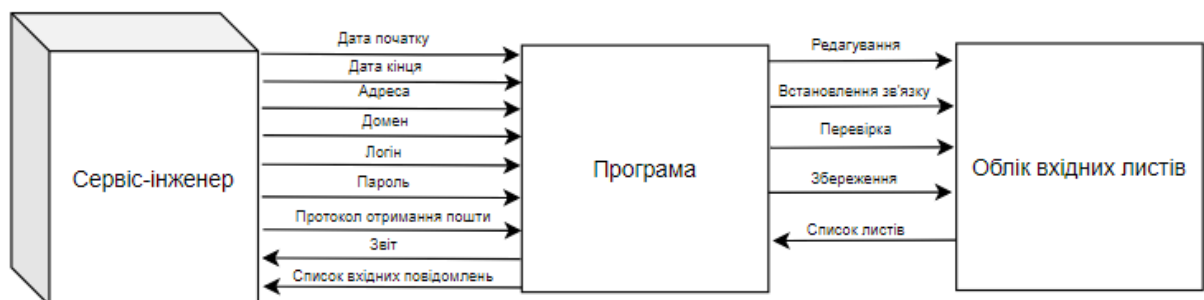


Рисунок 3.3. – Діаграма потоків даних програми «Облік вхідних електронних листів».

З діаграми видно: співробітник відділу інформаційних технологій задає вихідні дані (діапазон дат, адреса електронної пошти, домен, логін, пароль, протокол отримання пошти), ПЗ введення даними перевіряє пошту, фільтрує вхідні листи. На екран виводиться зведений список листів у вигляді табл. 3.1, які відповідають критеріям фільтрації.

Деталізуючи контекстну діаграму, складемо деталізовану діаграму потоків даних першого рівня. Вона представлена на рис. 3.4.

З діаграми видно, що в підсистему введення (додавання) і формування вводяться дані (діапазон дат, адреса електронної пошти, домен, логін, пароль, протокол отримання пошти), які зберігаються в ОЗУ. Підсистемі редагування і під час пошуку передаються зведений список вхідних даних. Після пошуку та складання списків вони виводяться на екран монітора. Список з внесеними змінами передається в підсистему збереження для їх збереження в ОЗУ та ПЗУ. Підсистема виведення здійснює висновок звіту «Вхідні електронні листи» на екран монітора і на тверді копії.

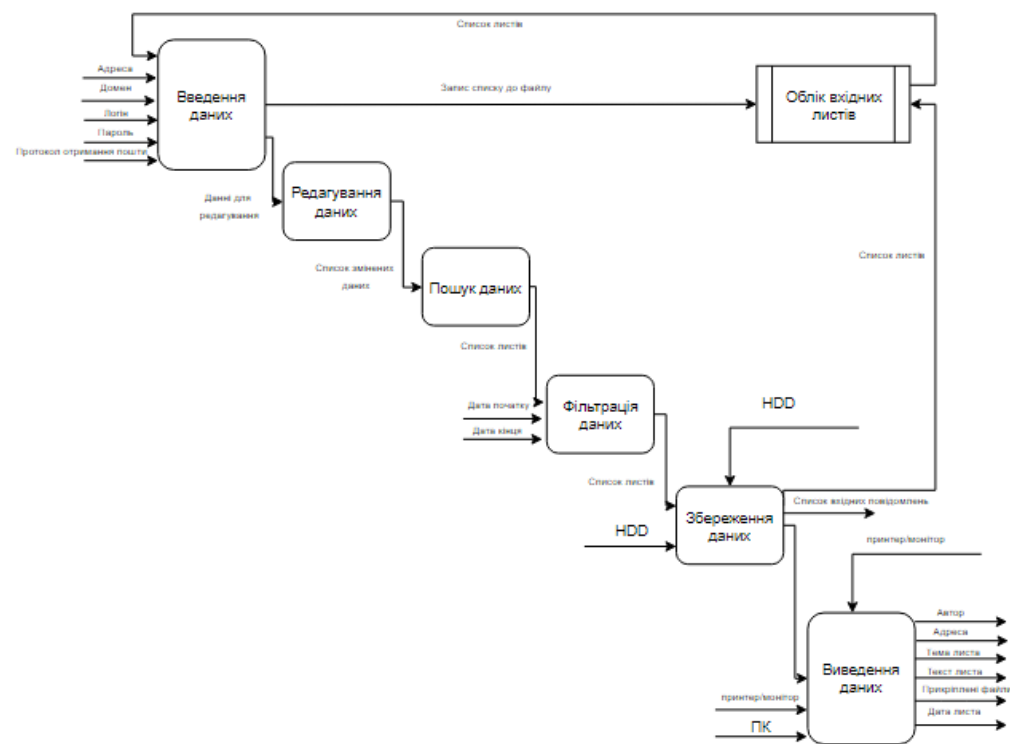


Рисунок 3.4. – Деталізована діаграма програми «Облік вхідних електронних листів».

Оскільки програма повинна мати дружній інтерфейс, розробимо діаграму станів. Перше вікно реалізує введення даних, а так само, при бажанні користувач, їх завантаження, редагування, видалення і додавання. Друге вікно – встановлення зв'язку з поштовим сервером для перевірки листів. Далі слід ініціалізація виконання. Діаграма станів представлена на рис. 3.5.

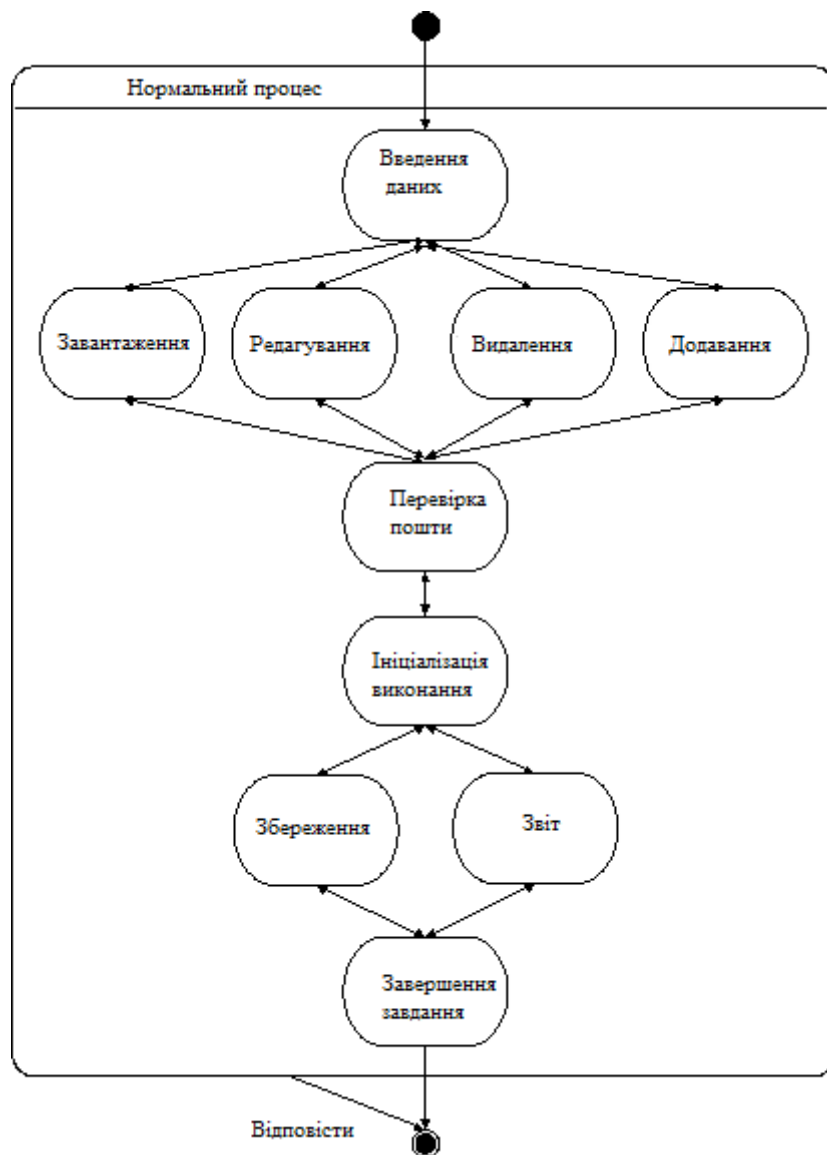


Рисунок 3.5 – Діаграма станів програми «Облік вхідних електронних листів»

3.2 Розробка та опис алгоритму

Після вивчення предметної області, визначення вхідних і вихідних даних, функцій програми можна приступити до розробки алгоритму у вигляді структурних схем.

На рис. 3.6 представлена схема керуючої програми.

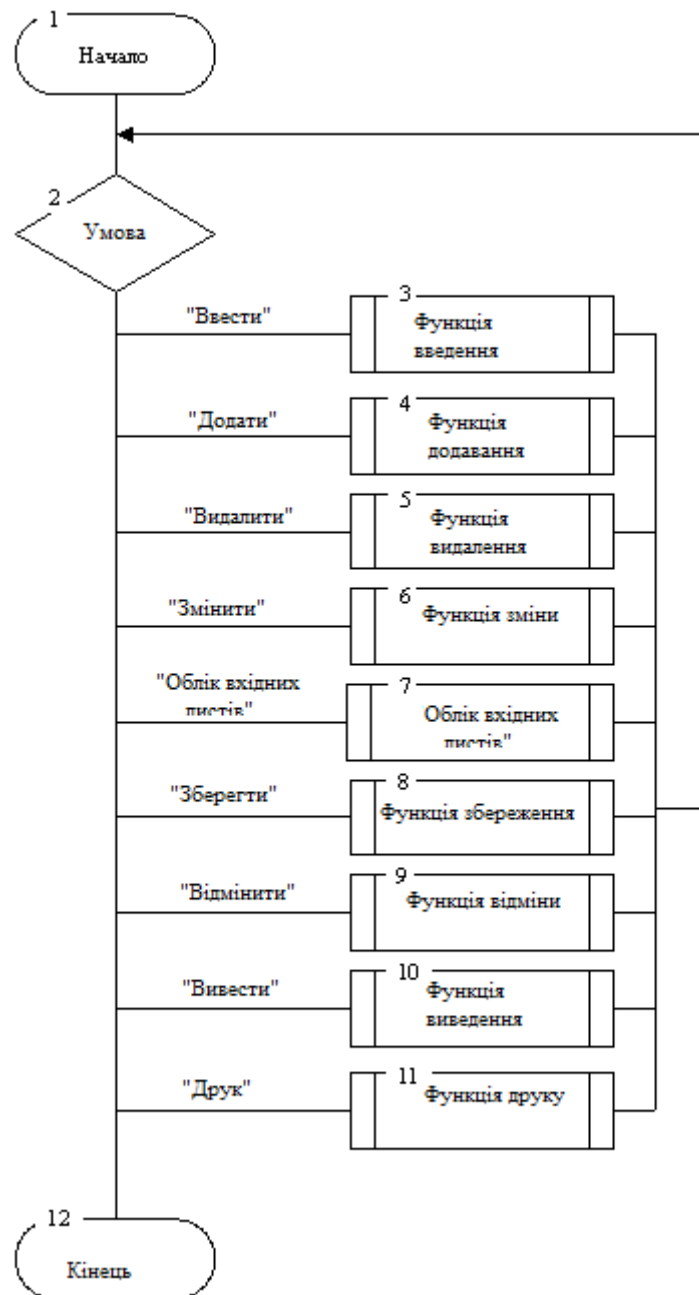


Рисунок 3.6 – Структурна схема керуючої програми

На рис. 3.7 представлено структурну схему функції «Введення».

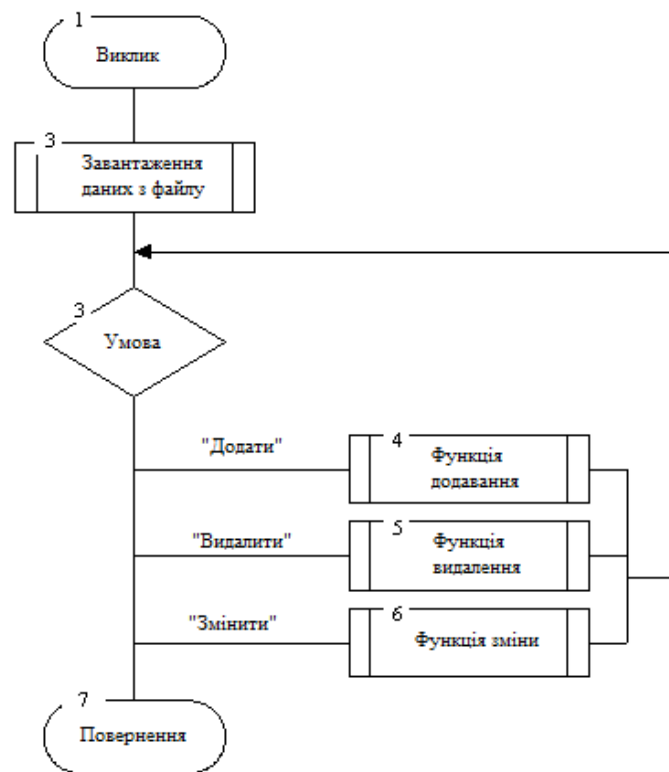


Рисунок 3.7 – Структурна схема функції «Введення»

Структурна схема функції додавання представлена на рис. 3.8.

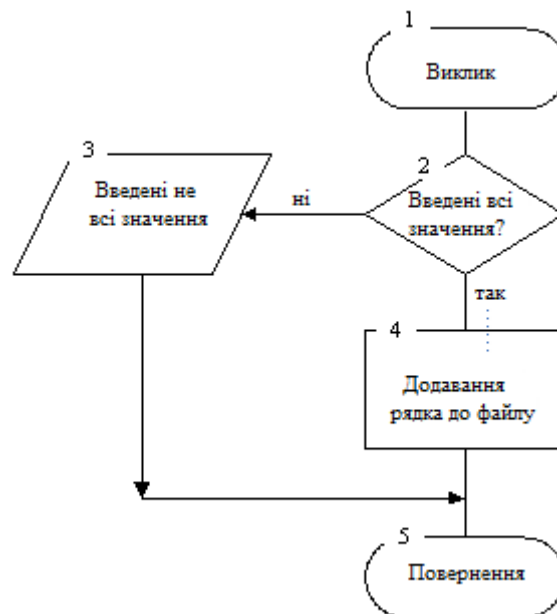


Рисунок 3.8 – Структурна схема функції «Додати»

Структурна схема функції видалення представлена на рис. 3.9.

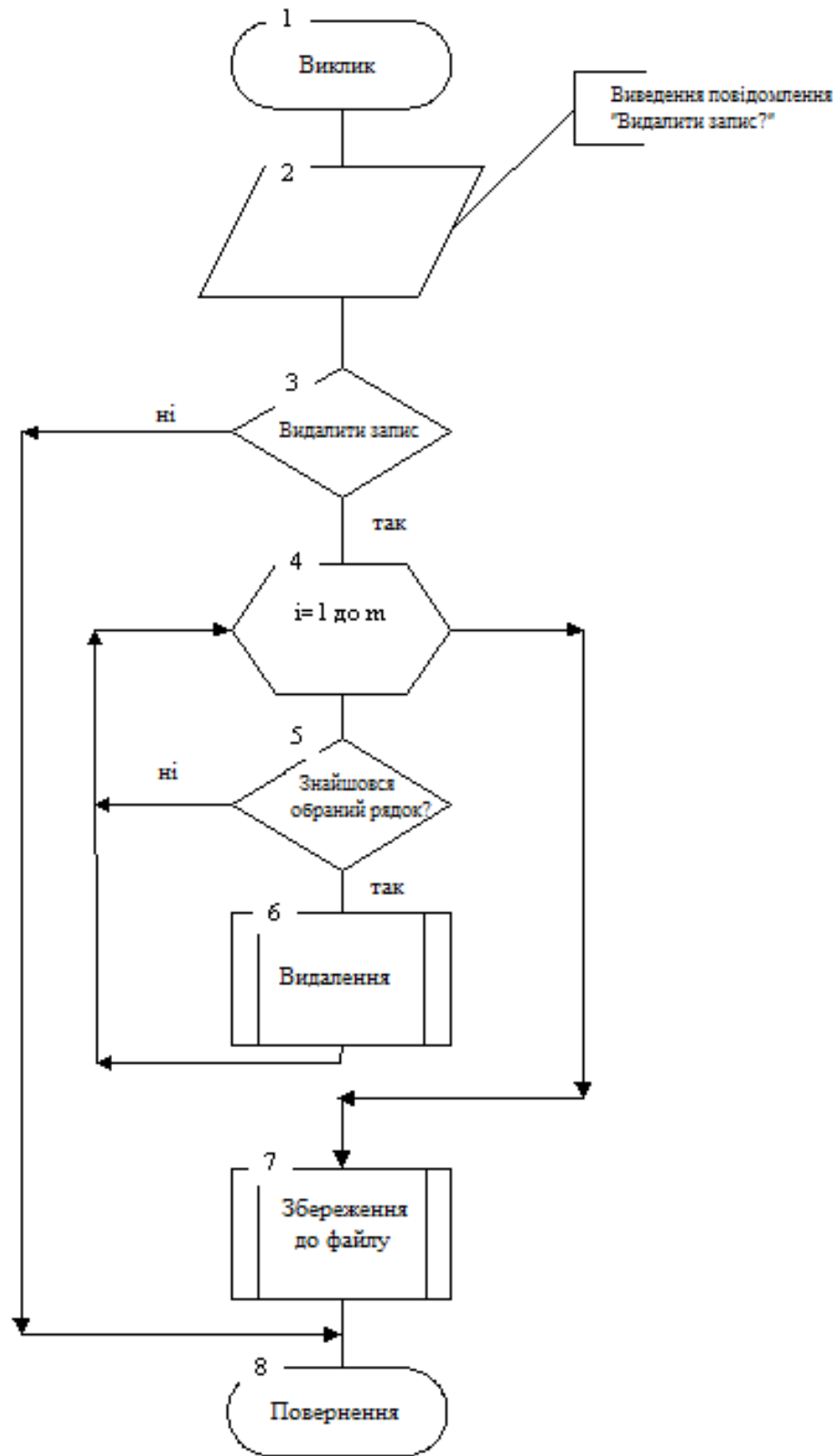


Рисунок 3.9 – Структурна схема функції «Видалити»

Структурна схема функції заміни представлена на рис. 3.10.

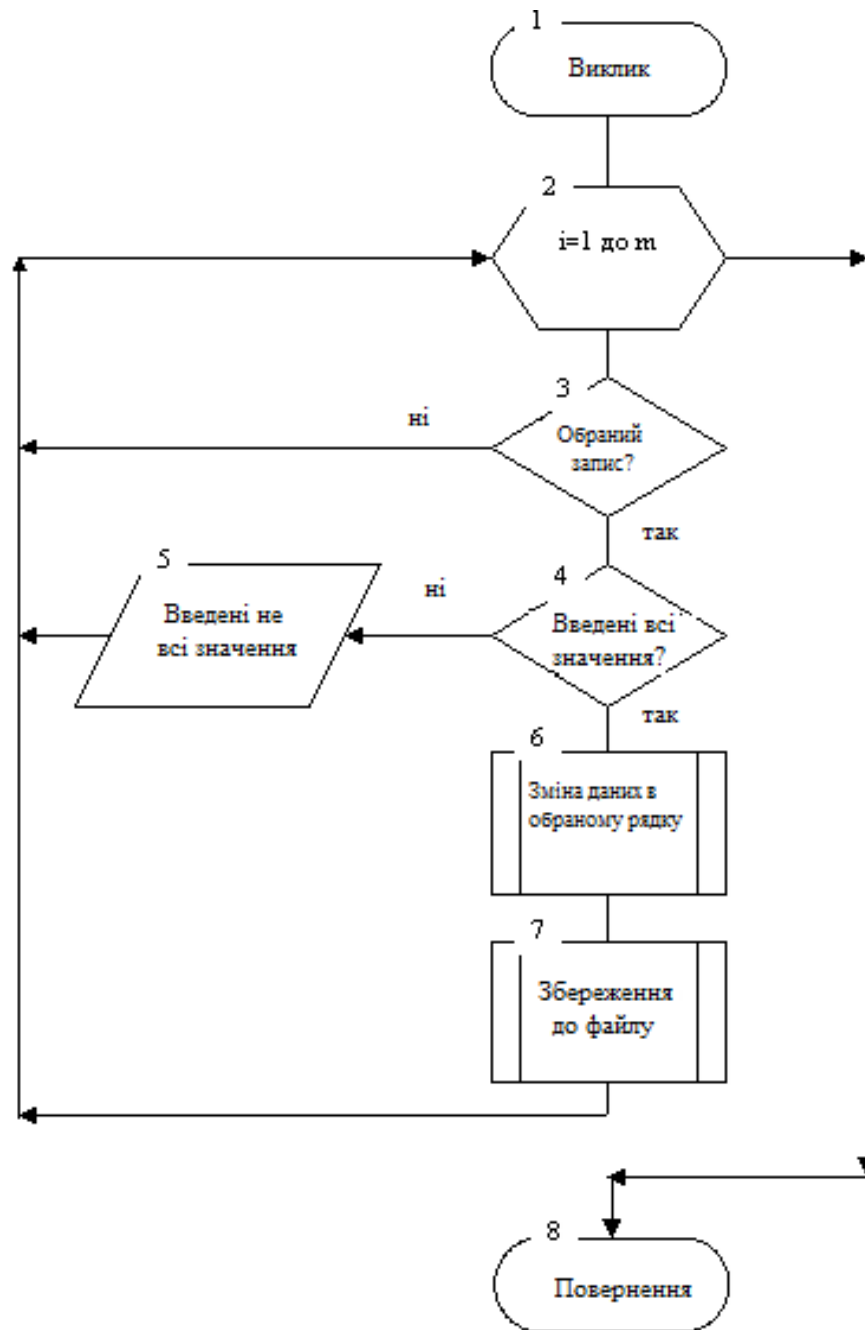


Рисунок 3.10 – Структурна схема функції «Зміна»

Структурна схема функції обліку вхідних листів представлена на рис.

3.11.

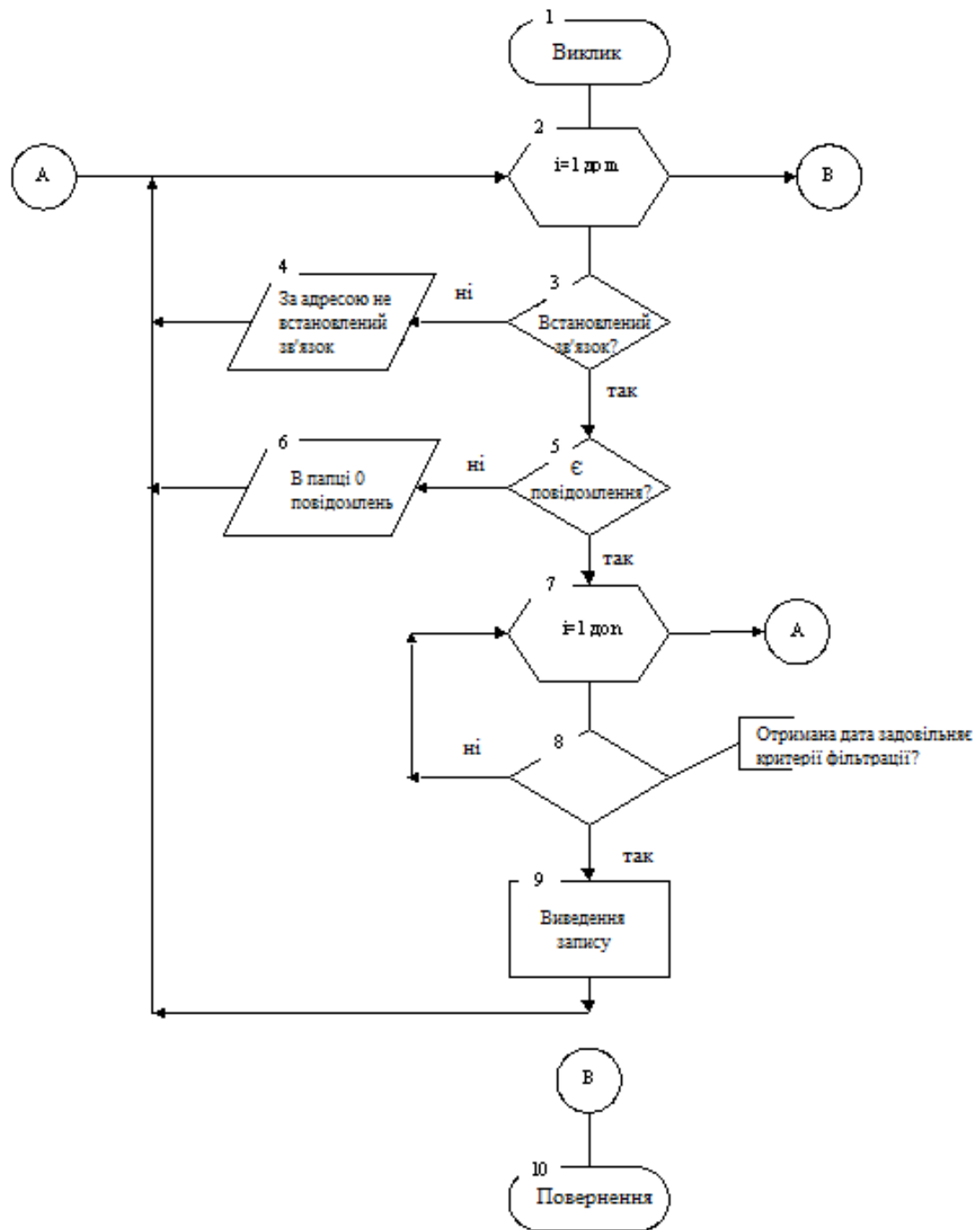


Рисунок 3.11 – Структурна схема функції «Облік вхідних листів»

Структурна схема функції збереження представлена на рис. 3.12.



Рисунок 3.12 – Структурна схема функції «Збереження»

Структурна схема функції виведення представлена на рис. 3.13.



Рисунок 3.13 – Структурна схема функції «Виведення»

На рис. 3.14 представлена структурна функція друку.

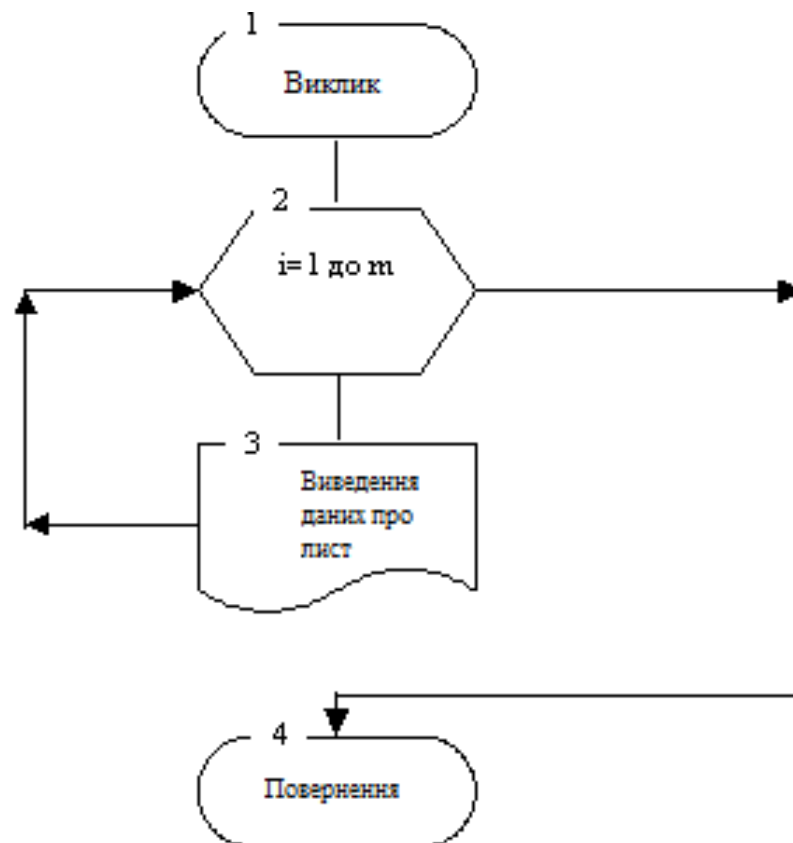


Рисунок 3.14 – Структурна схема функції «Друк»

3.3. Висновки до розділу

В розділі було побудовано структурно-функціональну схему програми, контекстну діаграму і діаграму потоків та визначені вхідні та вихідні дані. Після вивчення предметної області, визначення вхідних і вихідних даних, функцій програми було розроблено наступні алгоритми у вигляді структурних схем: структурна схема керуючої програми, Структурна схема функції «Введення», Структурна схема функції «Додати», Структурна схема функції «Видалити», Структурна схема функції «Зміна», Структурна схема функції «Облік вхідних листів», Структурна схема функції «Збереження», Структурна схема функції «Виведення», Структурна схема функції «Друк».

4 КЕРІВНИЦТВО ПРОГРАМІСТА

Дане програмне забезпечення розроблено відповідно до методу модульного структурування, з елементами об'єктно-орієнтованого програмування. Програма складається з наступних модулів:

- Check_up_letters.exe – виконуваний файл проєкту;
- UnitConnectForm(.dfm, .pas, .ddp, .dcu) – модуль з'єднання;
- Unit1(.dfm, .pas, .ddp, .dcu) – головний модуль програми;
- Unit2(.dfm, .pas, .ddp, .dcu) – модуль редагування (додавання, видалення, збереження);
- Unit3(.dfm, .pas, .ddp, .dcu) – модуль введення / додавання;
- Вхідні електронні листи.xls – звіт «Вхідні електронні листи»;
- Unit4(.dfm, .pas, .ddp, .dcu) – модуль обліку вхідних електронних листів;
- Unit5(.dfm, .pas, .ddp, .dcu) – модуль збереження;
- Unit6(.dfm, .pas, .ddp, .dcu) – модуль виведення.

Даний програмний продукт розроблений на платформі ОС Windows 10 в середовищі програмування Delphi 9.0

При створенні програми використовувався персональної комп'ютер наступної конфігурації: Intel Core i5-8250U (1.6 - 3.4 ГГц) / RAM 8 ГБ / SSD 512 ГБ / Intel UHD Graphics 620 .

Програма, яка не містить синтаксичних помилок проте може містити логічні помилки, що не дозволяють програмі виконувати закладені в ній функції. Логічні помилки можуть бути пов'язані з алгоритмом програми або з невірним розумінням роботи апаратури, під'єднаної до портів мікроконтролера.

Вбудований в склад інтегрованого середовища програмування відладник дозволяє налагодити ті ділянки коду програми, що не залежать від роботи апаратури, яка не входить до складу мікросхеми мікроконтролера.

Зазвичай це відноситься до обчислення математичних виразів або перетворення форматів представлення даних.

Для налагодження програм зазвичай застосовують три способи:

- покрокове налагодження програм з заходом в підпрограми;
- покрокове налагодження програм з виконанням підпрограми як одного оператора;
- виконання програми до точки зупинки.

Покрокове налагодження програм полягає в тому, що виконується один оператор програми, далі контролюються ті змінні, на які повинен був впливати даний оператор.

Якщо в програмі є вже налагоджені підпрограми, то підпрограму можна розглядати, як один оператор програми і скористатися другим способом налагодження програм.

Якщо в програмі існує досить велика ділянка програми, вже налагоджена раніше, то її можна виконати, не контролюючи змінні, на які вона впливає. Використання точок зупину дозволяє пропускати вже налагоджену частину програми. Точка зупину встановлюється в місцях, де необхідно перевірити вміст змінних або просто проконтролювати, передається чи управління даним оператору.

Практично у всіх відладчика підтримується ця властивість (а також виконання програми до курсора і вихід з підпрограми). Потім налагодження програми триває в покроковому режимі з контролем локальних і глобальних змінних.

На рис. 4.1 представлений вид екранної форми «Редагування» при тестуванні функції Видалити.

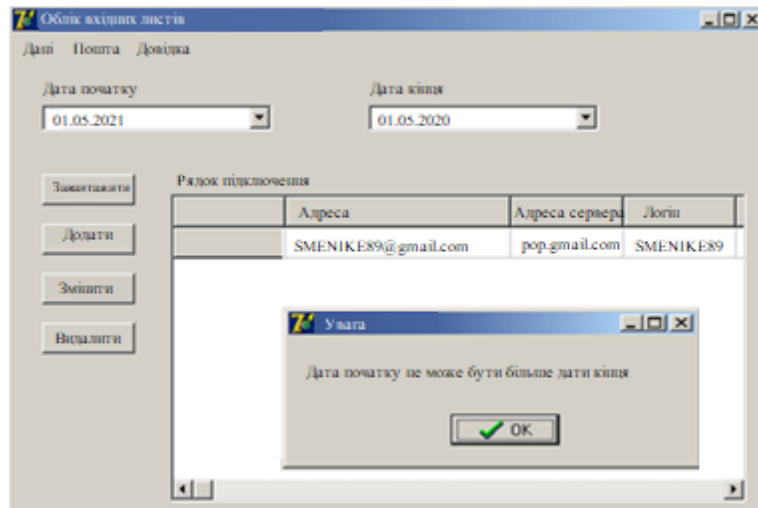


Рисунок 4.1 – Вид екранної форми «Редагування» при тестуванні функції Видалити

Вид екранної форми попередження при тестуванні функції зміни представлений на рис 4.2.

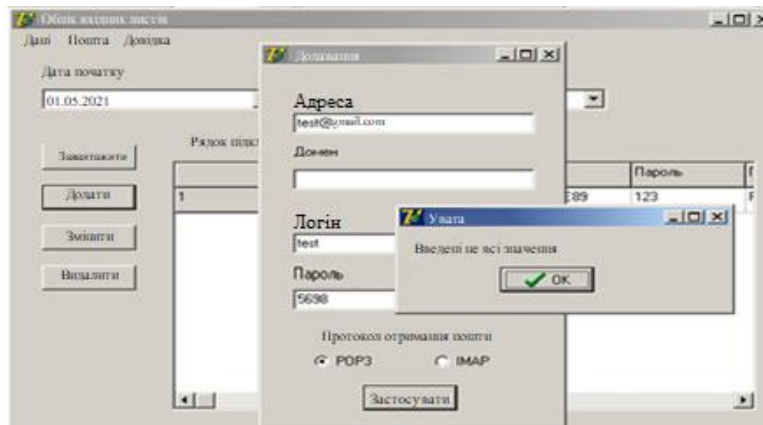


Рисунок 4.2 – Вид екранної форми попередження при тестуванні функції Зміна

Вид екранної форми попередження при тестуванні функції «Облік вхідних електронних листів» представлений на рис. 4.3

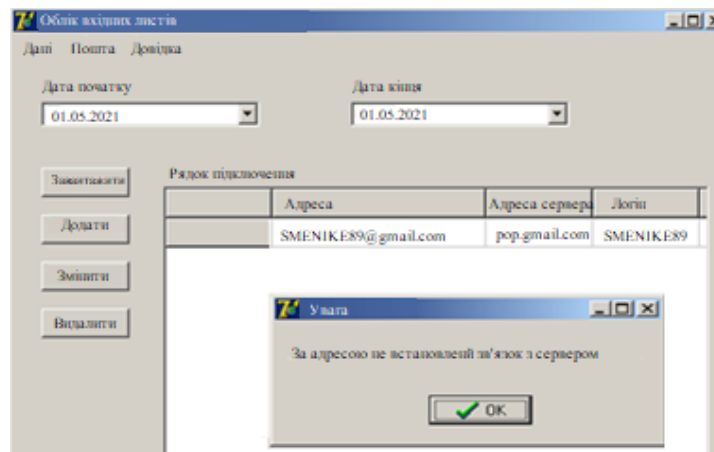


Рисунок 4.3 – Вид екранної форми попередження при тестуванні функції
«Облік вхідних електронних листів»

Вид екранної форми попередження про відсутність листів при тестуванні функції «Облік вхідних електронних листів» представлений на рис. 4.4.

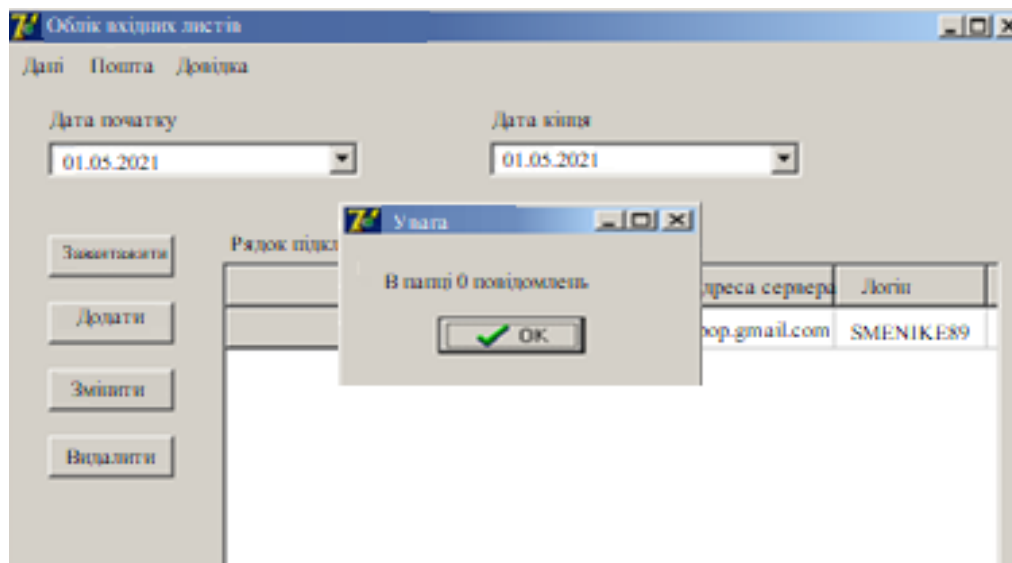


Рисунок 4.4 – Вид екранної форми попередження при тестуванні функції
«Облік вхідних електронних листів»

Вид екранної форми «Перевірка пошти» при тестуванні функції «Облік вхідних електронних листів» представлений на рис. 4.5

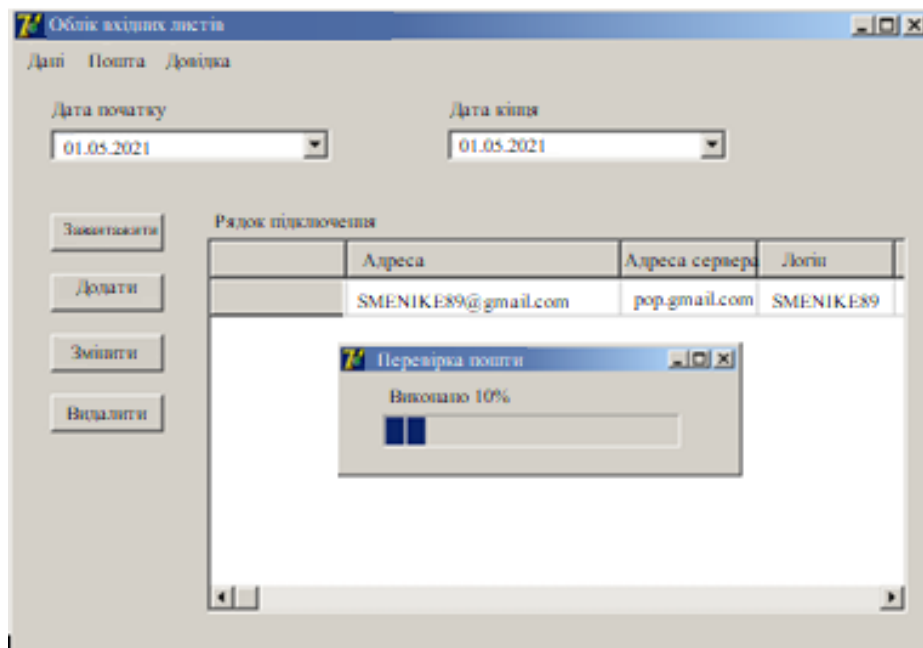


Рисунок 4.5 – Вид екранної форми «Перевірка пошти» при тестуванні функції «Облік вхідних електронних листів»

5 КЕРІВНИЦТВО КОРИСТУВАЧА

Для запуску програми необхідно натиснути кнопкою по іконі програми, після чого буде доступна перша форма програми. Вид екранної форми представлений на рис.5.1.

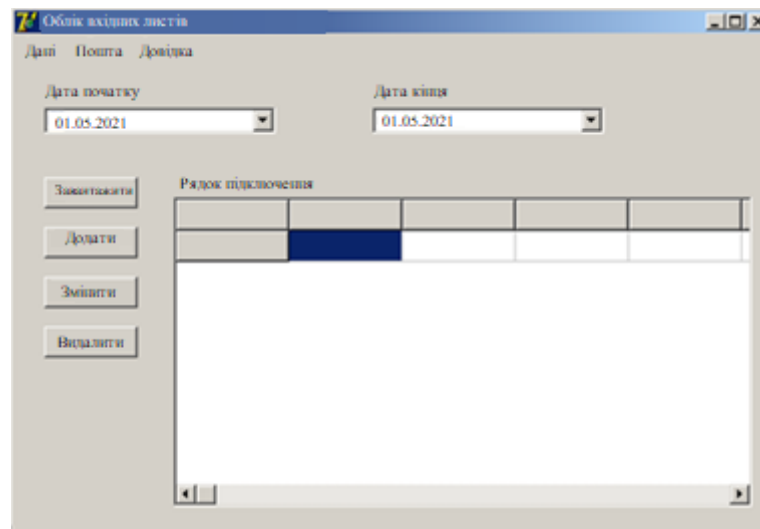


Рисунок 5.1 – Вид екранної форми «Облік вхідних листів»

Для додавання нової електронної пошти, листи на якій необхідно перевірити, треба натиснути кнопку «Додати» або «Дані» – «Додати», ввести нові дані, та натиснути кнопку «Застосувати». Вид екранної форми представлений на рис.5.2.

Якщо користувач не вводить дані, програма повідомить користувачеві про помилку. Вид екранної форми представлений на рис.5.3.

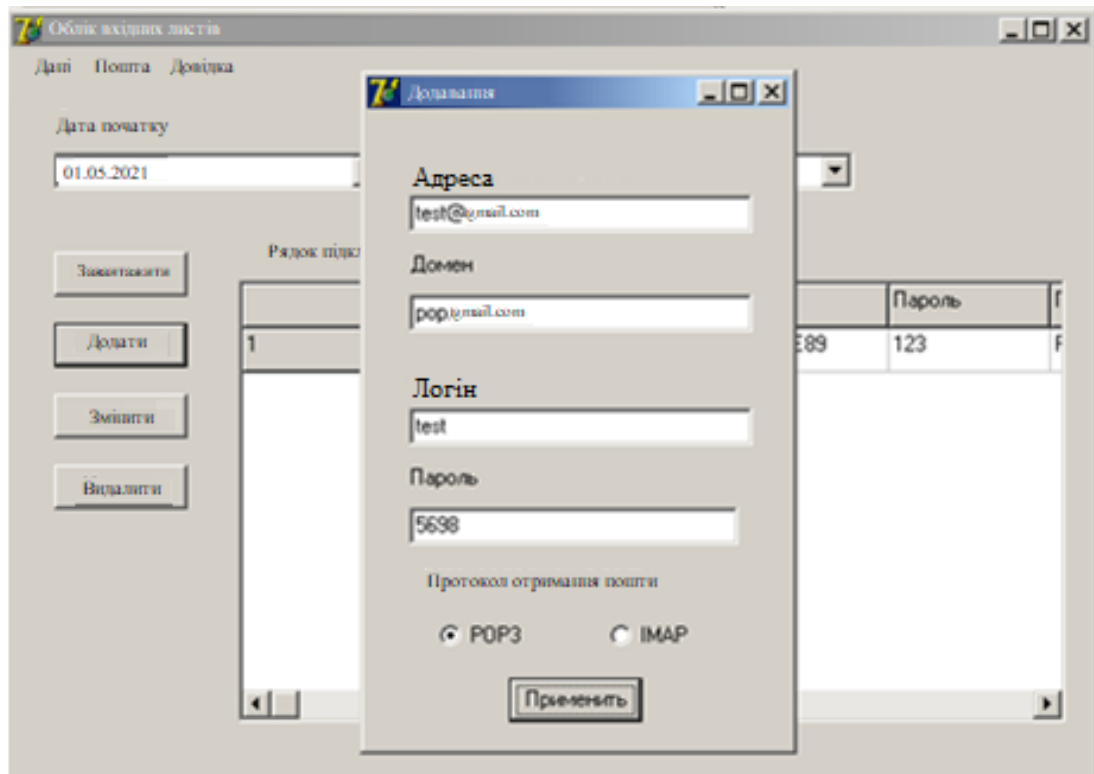


Рисунок 5.2 – Вид екранної форми «Додавання»

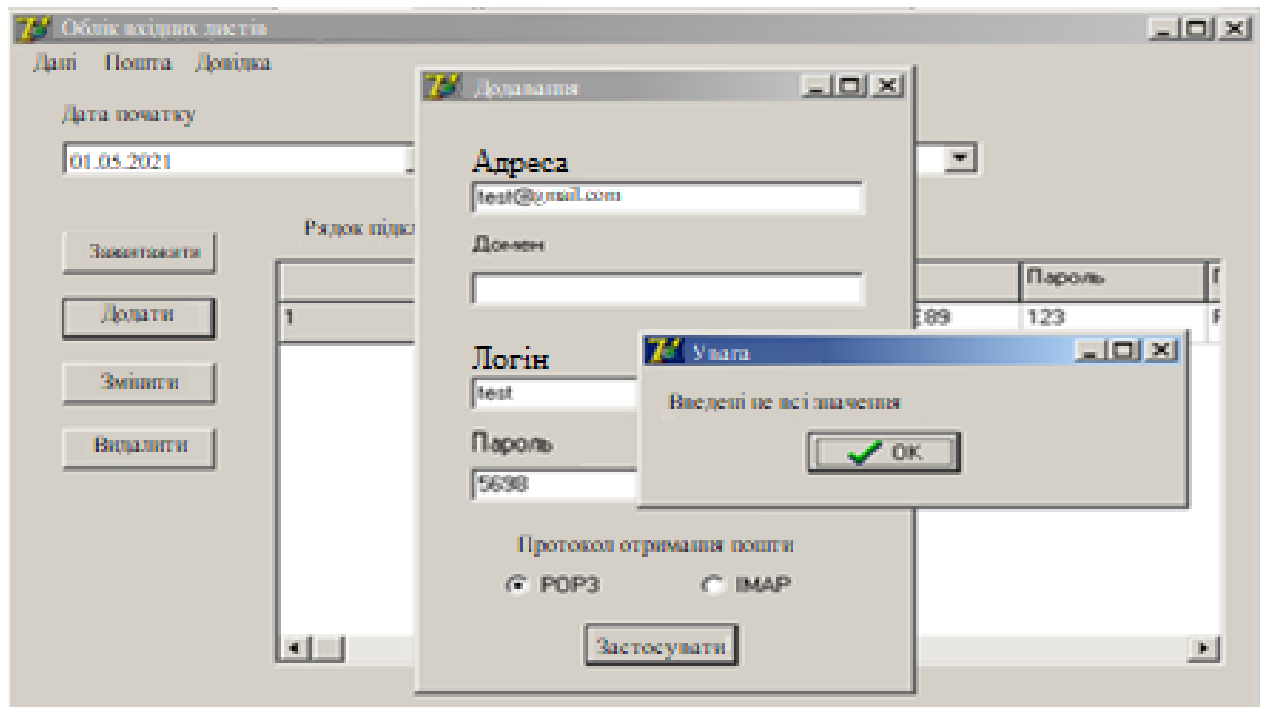


Рисунок 5.3 – Вид екранної форми повідомлення про помилку

Для зміни вже внесених даних про електронну адресу необхідно натиснути кнопку «Змінити». Вид екранної форми представлений на рис.5.4.

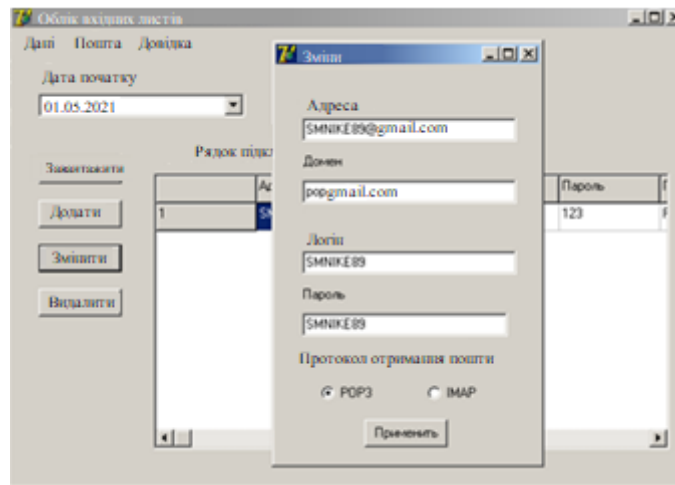


Рисунок 5.4 – Вид екранної форми «Зміна»

Якщо список адрес, за якими необхідно провести перевірку, вже створено в форматі * txt, то програма може завантажити його. Для цього необхідно натиснути кнопку «Завантажити», в діалоговому вікні вибрати файл, і натиснути «Відкрити». Вид екранної форми представлений на рис.5.5.

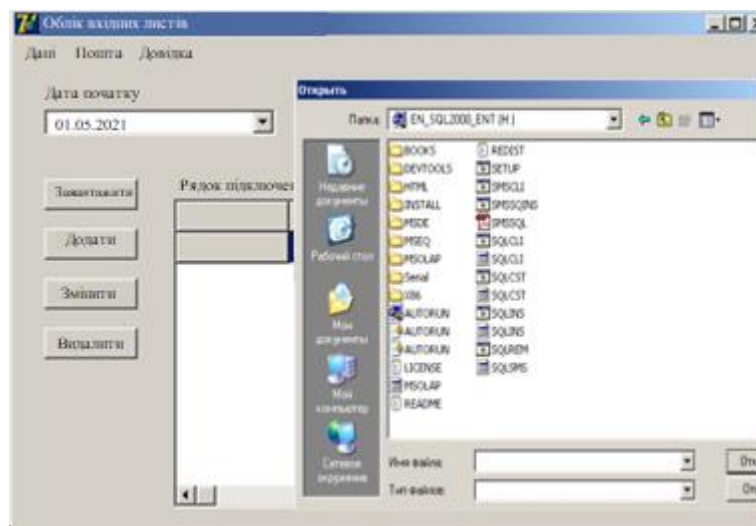


Рисунок 5.5 – Вид екранної форми «Відкрити»

Після додання адрес до таблиці «Рядки підключення», можна провести перевірку пошти на наявність вхідних повідомлень. Для цього

необхідно натиснути «Пошта» – «Перевірка пошти», користувачу буде показано стан перевірки вхідних електронних листів. Вид екранної форми представлений на рис.5.6.

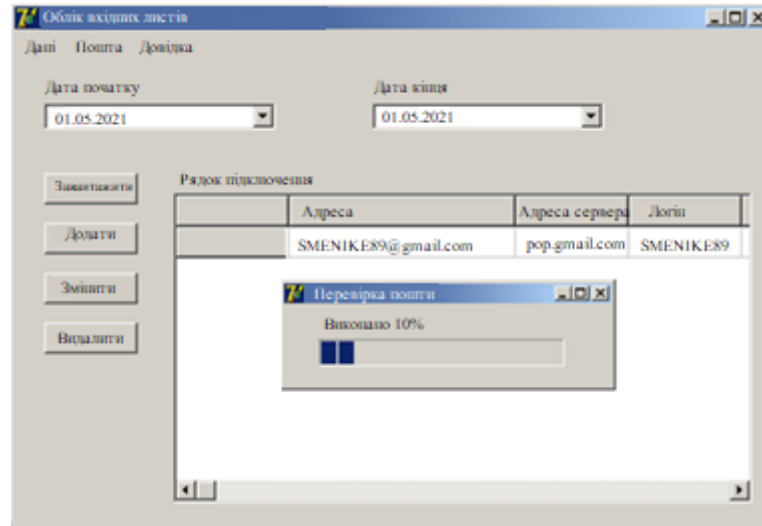


Рисунок 5.6 – Вид екранної форми «Перевірка пошти»

Якщо зв'язок не встановлений з будь-яких причин, програма повідомить користувачеві номер рядка підключення, через який не встановлено з'єднання з сервером. Вид екранної форми представлений на рис.5.7.

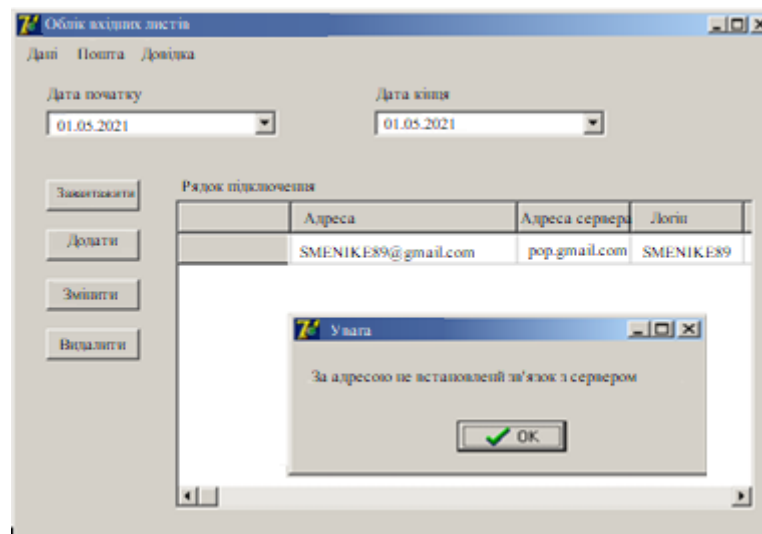


Рисунок 5.7 – Вид екранної форми повідомлення користувачу про помилку

У разі якщо зв'язок встановлено, але в папці «Вхідні» немає листів, які відповідають критеріям запити, програма покаже відповідне повідомлення користувачу. Вид екранної форми представлений на рис.5.8.

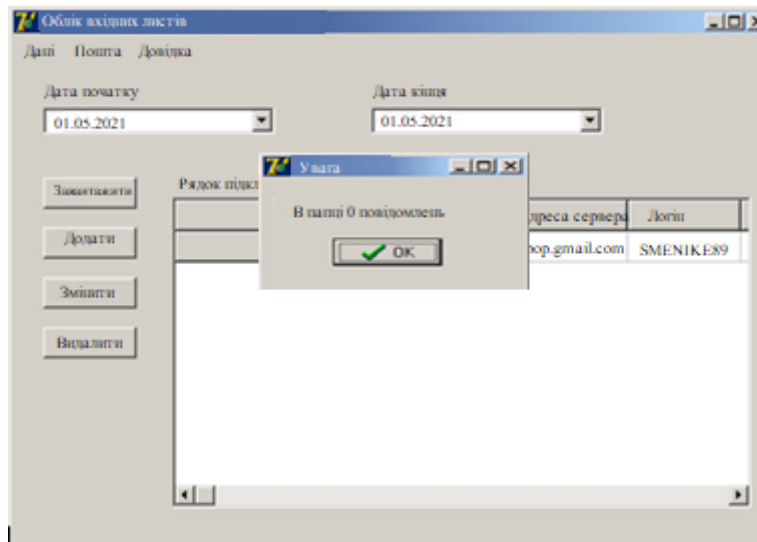


Рисунок 5.8 – Вид екранної форми повідомлення користувачу про помилку

Після перевірки вхідних електронних листів, з'явиться вікно, що показує користувачеві дані про листи, які відповідають критеріям запити. Вид екранної форми представлений на рис.5.9.

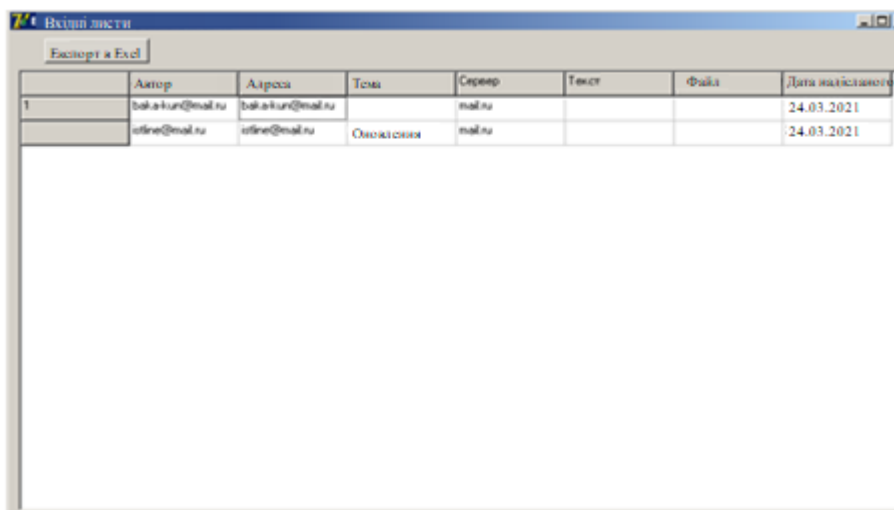


Рисунок 5.9 – Вид екранної форми «Вхідні електронні листи».

Для експорту в Excel необхідно натиснути кнопку «Експорт в Excel».

Сформований файл буде мати назву «Вхідні електронні листи».

Для виходу з програми необхідно закрити всі відкриті вікна.

ВИСНОВОК

В дипломній роботі була спроектована і реалізована програма обліку вхідних електронних листів на малому підприємстві. Дана програма забезпечує наступні основні функції: з'єднання з сервером по рядках підключень; перевірку наявності вхідних повідомлень; перевірку дати, зазначеної в листі, на входження в зазначений діапазон дат; формування результату: звіту вхідних електронних листів за певний період.

Реалізація системи проводилася відповідно до вимог замовника. Вибір середовища програмування був обумовлений також основними характеристиками більшості використовуваного програмного забезпечення на малих підприємствах. При розробці програми використовувалися Borland Delphi 9.0. В процесі написання програми велика увага була приділена зручності роботи користувача і побудови дружнього інтерфейсу.

Введення програми забезпечить автоматизований безпомилковий облік всіх електронних листів, що містять відомості про оновлення, дасть можливість формування достовірної та повної інформації про всіх клієнтів, що здійснили поновлення для ПЗ.

В ході виконання дипломної роботи були виконані наступні завдання:

- проведено аналіз способів та методів оновлення програмного забезпечення;
- проведено аналіз існуючих підходів для інформаційної підтримки інженера;
- проведено обґрунтування вибору методу рішення та інструментів;
- побудовано структурно-функціональну схему;
- побудовано алгоритми роботи програми;
- розроблено програму для інформаційної підтримки інженера з питань обслуговування програмного забезпечення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Обновление программного обеспечения интегрированных серверов [Электрон. ресурс]. – 2020 – Режим доступа: <https://www.ibm.com/docs/ru/i/7.1?topic=management-software-updates>
2. Различные типы обновлений Windows [Электрон. ресурс]. – 2019 – Режим доступа: <https://techarks.ru/windows/obnovleniya-windows/razlichnye-tipy-obnovlenij-windows/>
3. Планирование обновлений программного обеспечения в Configuration Manager [Электрон. ресурс]. – 2021 – Режим доступа: <https://docs.microsoft.com/ru-ru/mem/configmgr/sum/plan-design/plan-for-software-updates>
4. IObit Software Updater [Электрон. ресурс]. – 2021 – Режим доступа: <https://softcatalog.info/ru/programmy/iobit-software-updater>
5. SUMo [Электрон. ресурс]. – 2021 – Режим доступа: <https://softcatalog.info/ru/obzor/programmy-dlya-obnovleniya-programm>
6. DriverPack Solution [Электрон. ресурс]. – 2021 – Режим доступа: <https://softcatalog.info/ru/programmy/driverpack-solution>
7. Климова, Л. М. Delphi 7. Основы программирования. Решение типовых задач. Самоучитель [Текст] / Л.М. Климова. – М.: КУДИЦ–Образ, – 2017. – 480 с.
8. Культин Н. Основы программирования в Delphi 7 [Текст] / Н. Культин,с//. М.: СПб: БХВ, – 2018. – 608 с.
9. Культин, Н. Delphi 6. Программирование на Object Pascal [Текст]/ Н. Культин // М.: БХВ-Петербург, – 2019. – 528 с.
10. Ревич, Ю. Нестандартные приемы программирования на Delphi [Текст] / Ю. Ревич.//– М.: БХВ-Петербург, 2016. – 560 с.
11. Campbell C. Parallel Programming with Microsoft® Visual C++® [Текст] / С. Campbell.// Гостехиздат, 2016. – 784 с.

12. Давыдов, В. Visual C++. Разработка Windows-приложений с помощью MFC и API-функций [Текст]/ В. Давыдов. // М.: БХВ-Петербург, 2014. – 576 с.

13. Довбуш, Галина Visual C++ на примерах [Текст]/ Г. Довбуш , А.Хомоненко. // М.: БХВ-Петербург, – 2012. – 528 с.

ДОДАТОК А
Текст програми

Текст модуля вхідних даних

```
unit chek_up_letters;
```

```
interface
```

```
uses
```

```
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  
Dialogs, StdCtrls, IdBaseComponent, IdComponent, IdTCPConnection,  
IdTCPClient, IdMessageClient, IdIMAP4, IdCoder, IdCoder3to4, IdCoderMIME,  
ExtCtrls, Menus, INIFiles, IdPOP3, IdMessage, IdTCPServer, IdSMTPServer,  
IdSMTP;
```

```
type
```

```
TForm1 = class(TForm)
```

```
IdIMAP41: TIdIMAP4;
```

```
IdDecoderMIME1: TIdDecoderMIME;
```

```
Timer2: TTimer;
```

```
IdPOP31: TIdPOP3;
```

```
IdMessage1: TIdMessage;
```

```
Button2: TButton;
```

```
Memo2: TMemo;
```

```
Listbox1: TListBox;
```

```
Button3: TButton;
```

```
Button4: TButton;
```

```
Button5: TButton;
```

```
IdSMTPServer1: TIdSMTPServer;
```

```
Label1: TLabel;
```

```
Label2: TLabel;
```

```
procedure FormCreate(Sender: TObject);
```

```
procedure Button4Click(Sender: TObject);
```

```
procedure Listbox1DbClick(Sender: TObject);
```

```
procedure Button5Click(Sender: TObject);
```

```
procedure Button3Click(Sender: TObject);
```

```
procedure Button2Click(Sender: TObject);
```

```
procedure Button6Click(Sender: TObject);
```

```

private
{ Private declarations }
public
{ Public declarations }
end;

```

```

var
Form1: TForm1;

```

```

implementation

```

```

uses Math, adress, em;

```

```

{$R *.dfm}

```

```

function KOI8_RToUTF16Str(const s: string): WideString;
var
i,j: integer;
begin
j:= length(s);
setLength(Result,j);
for i:=1 to j do
Result[i]:=KOI8_RToUTF16Char(s[i]);
end;

```

```

function DecodeStr(Text: string): string;
var
Temp, Dec: string;
P,P2:Integer;
begin
Temp:=Text;
Result:="";

```

```

while pos('=?windows-1251?',Temp)>0 do
begin
P:=pos('=?windows-1251?',Temp);

```

```

// P2:=pos('?=',Copy(Temp,P,Length(Temp)))+P;
P2:=pos('?=',Temp);
Dec:=Copy(Temp,p+17,P2-(P+17));
Dec:=Form1.IdDecoderMIME1.DecodeString(Dec);
Result:=Result+Copy(Temp,1,P-1)+Dec;
Temp:=Copy(Temp,P2+2,length(Temp));
end;
Result:=Result+Temp;
Temp:=Result;

Result:="";
while pos('=?koi8-r?',Temp)>0 do
begin
P:=pos('=?koi8-r?',Temp);
P2:=pos('?=',Temp);
Dec:=Copy(Temp,p+11,P2-(P+11));
try
if Dec<>""
then Dec:=KOI8_RTtoUTF16Str(Form1.IdDecoderMIME1.DecodeString(Dec));
except

end;
Result:=Result+Copy(Temp,1,P-1)+Dec;
Temp:=Copy(Temp,P2+2,length(Temp));
end;
Result:=Result+Temp;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
Application.Title:='Перевірка та відправлення пошти (POP3, IMAP)';
//
ListBox1.Items.LoadFromFile(ExtractFilePath(ParamStr(0))+ 'emails.txt');
try
ListBox1.Selected[0]:=true;
except

end;

```

end;

```
procedure TForm1.Button4Click(Sender: TObject);
```

```
begin
```

```
Form2.ShowModal;
```

```
end;
```

```
procedure TForm1.ListBox1DbClick(Sender: TObject);
```

```
begin
```

```
Form2.ShowModal;
```

```
end;
```

```
procedure TForm1.Button5Click(Sender: TObject);
```

```
var
```

```
i: integer;
```

```
begin
```

```
if MessageDlg('Ви дійсно хочете видалити адресу?', mtConfirmation,  
[mbYes, mbNo],0)=mrNo
```

```
then Exit
```

```
else
```

```
begin
```

```
for i:=0 to Form1.ListBox1.Items.Count-1 do
```

```
begin
```

```
if Form1.ListBox1.Selected[i]
```

```
then
```

```
begin
```

```
ListBox1.Items.Delete(i);
```

```
ListBox1.Items.SaveToFile(ExtractFilePath(ParamStr(0))+ 'emails.txt');
```

```
Break;
```

```
end;
```

```
end;
```

```
end;
```

```
end;
```

```
procedure TForm1.Button3Click(Sender: TObject);
```

```
begin
```

```
ListBox1.Items.Add('new@new.ru|new.mail.ru|user|password|POP3');
```

```
end;
```

```

procedure TForm1.Button2Click(Sender: TObject);
var
i: integer;
s,usr: string;
imap: boolean;
begin
Memo2.Clear;
imap:=false;
for i:=0 to Form1.ListBox1.Items.Count-1 do
begin
if Form1.ListBox1.Selected[i]
then
begin
s:=Form1.ListBox1.Items.Strings[i];
Break;
end;
end;
//
IdIMAP41.Port:=25;
IdPOP31.Port:=110;
//
usr:=Copy(s,1,pos('|',s)-1);
Delete(s,1,pos('|',s));
IdIMAP41.Host:=Copy(s,1,pos('|',s)-1);
IdPOP31.Host:=IdIMAP41.Host;
//
Delete(s,1,pos('|',s));
IdIMAP41.Username:=Copy(s,1,pos('|',s)-1);
IdPOP31.Username:=IdIMAP41.Username;
//
Delete(s,1,pos('|',s));
IdIMAP41.Password:=Copy(s,1,pos('|',s)-1);
IdPOP31.Password:=IdIMAP41.Password;
//
Delete(s,1,pos('|',s));
if s='POP3'
then imap:=false

```

```

else
if s='IMAP'
then imap:=true;
////////// перевірка пошти
Memo2.Clear;
Timer2.Enabled:=true;
Timer2.Interval:=1000*60*5; // раз в 5 хвилин
try
if imap=true
then
begin
IdIMAP41.Connect(30000);
IdIMAP41.SelectMailBox('INBOX');
IdIMAP41.RetrieveAllHeaders(IdIMAP41.MailBox.MessageList);
//
if IdIMAP41.MailBox.MessageList.Count=0
then ShowMessage('В папці 0 повідомлень!');
end
else
begin
IdPOP31.Connect(1000);
if IdPOP31.CheckMessages=0
then ShowMessage('В папці 0 повідомлень!');
end;
if imap=true
then
begin //IMAP4
Memo2.Lines.Add('-----');
-----');
Memo2.Lines.Add('Ім'я поштової скриньки (IMAP): '+usr+' (всього повідомлень:
'+IntToStr(IdIMAP41.MailBox.MessageList.Count)+'));
Memo2.Lines.Add('-----');
-----');
for i:=0 to IdIMAP41.MailBox.MessageList.Count-1 do
begin
Memo2.Lines.Add(''+IntToStr(i+1)+'':
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].From.Text));

```

Від:

```

Memo2.Lines.Add('
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].From.Address));
Memo2.Lines.Add('
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].From.Name));
Memo2.Lines.Add(' Тема: '+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].Subject));
Memo2.Lines.Add('
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].MsgId));
Memo2.Lines.Add('
Attachment
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].AttachmentEncoding));
Memo2.Lines.Add('
Charset:
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].CharSet));
Memo2.Lines.Add('
Текст
'+DecodeStr(IdIMAP41.MailBox.MessageList.Messages[i].Body.Text));
Memo2.Lines.Add(' ');
end;
IdIMAP41.Disconnect;
end
else
begin //POP3
Memo2.Lines.Add('-----
-----');
Memo2.Lines.Add('Ім'я поштової скриньки (POP): '+usr+' (всього повідомлень не відомо)');
Memo2.Lines.Add('-----
-----');
for i:=1 to IdPOP31.CheckMessages do
begin
IdMessage1.Clear;
IdPOP31.RetrieveHeader(i,IdMessage1);
Memo2.Lines.Add(' '+IntToStr(i)+' От: '+DecodeStr(IdMessage1.From.Text));
Memo2.Lines.Add(' Адреса: '+DecodeStr(IdMessage1.From.Address));
Memo2.Lines.Add(' Ім'я: '+DecodeStr(IdMessage1.From.Name));
Memo2.Lines.Add(' Тема: '+DecodeStr(IdMessage1.Subject));
Memo2.Lines.Add(' Сервер: '+DecodeStr(IdMessage1.MsgId));
Memo2.Lines.Add(' Attachment Encoding: '+DecodeStr(IdMessage1.AttachmentEncoding));
Memo2.Lines.Add(' Charset: '+DecodeStr(IdMessage1.CharSet));
Memo2.Lines.Add(' Текст повідомлення: '+DecodeStr(IdMessage1.Body.Text));
Memo2.Lines.Add(' ');
end;

```

```

IdPOP31.Disconnect;
end;
except

end;
end;

procedure TForm1.Button6Click(Sender: TObject);
begin
Form3.ShowModal;
end;

end

unit adress;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls;

type
TForm2 = class(TForm)
Edit1: TEdit;
Label1: TLabel;
Label2: TLabel;
Edit2: TEdit;
Label3: TLabel;
Edit3: TEdit;
Label4: TLabel;
Edit4: TEdit;
Button1: TButton;
RadioButton1: TRadioButton;
RadioButton2: TRadioButton;
procedure FormShow(Sender: TObject);
procedure Button1Click(Sender: TObject);
private

```

```

{ Private declarations }
public
{ Public declarations }
end;

```

Текст модуля редагування

```

var
Form2: TForm2;

implementation

uses Unit1;

{$R *.dfm}

procedure TForm2.FormShow(Sender: TObject);
var
i: integer;
s: string;
begin
for i:=0 to Form1.ListBox1.Items.Count-1 do
begin
if Form1.ListBox1.Selected[i]
then
begin
s:=Form1.ListBox1.Items.Strings[i];
//
Edit1.Text:=Copy(s,1,pos('|',s)-1);
Delete(s,1,pos('|',s));
Edit2.Text:=Copy(s,1,pos('|',s)-1);
Delete(s,1,pos('|',s));
Edit3.Text:=Copy(s,1,pos('|',s)-1);
Delete(s,1,pos('|',s));
Edit4.Text:=Copy(s,1,pos('|',s)-1);
Delete(s,1,pos('|',s));
if s='POP3'
then RadioButton1.Checked:=true

```

```

else
if s='IMAP'
then RadioButton2.Checked:=true
else RadioButton1.Checked:=true;
if Edit1.Text=""
then Edit1.Text:=Form1.ListBox1.Items.Strings[i];
//
Break;
end;
end;
end;

procedure TForm2.Button1Click(Sender: TObject);
var
i: integer;
s: string;
begin
for i:=0 to Form1.ListBox1.Items.Count-1 do
begin
if Form1.ListBox1.Selected[i]
then
begin
if (Edit1.Text<>") and (Edit2.Text<>")
and (Edit3.Text<>") and (Edit4.Text<>")
then
begin
s:=Edit1.Text+'|'+Edit2.Text+'|'+Edit3.Text+'|'+Edit4.Text;
if RadioButton1.Checked
then s:=s+'|'+ 'POP3'
else s:=s+'|'+ 'IMAP';
Form1.ListBox1.Items.Strings[i]:=s;
Form1.ListBox1.Items.SaveToFile(ExtractFilePath(ParamStr(0))+ 'emails.txt');
end
else ShowMessage('Одне з полів не заповнено!');
Break;
end;
end;
end;
Close;

```

end;

end.

Текст модуля виведення

unit em;

interface

uses

infiles, Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, IdComponent, IdTCPConnection, IdTCPClient, IdMessageClient,
IdSMTP, ComCtrls, StdCtrls, Buttons, ExtCtrls, IdBaseComponent, IdMessage;

type

TForm3 = class(TForm)

MailMessage: TIdMessage;

SMTP: TIdSMTP;

AttachmentDialog: TOpenDialog;

ledHost: TLabeledEdit;

ledFrom: TLabeledEdit;

ledTo: TLabeledEdit;

ledCC: TLabeledEdit;

ledSubject: TLabeledEdit;

ledAttachment: TLabeledEdit;

Memo2: TMemo;

Label1: TLabel;

btnSendMail: TBitBtn;

Memo1: TMemo;

Button1: TButton;

LabeledEdit1: TLabeledEdit;

LabeledEdit2: TLabeledEdit;

procedure btnSendMailClick(Sender: TObject);

procedure SMTPStatus(ASender: TObject; const AStatus: TIdStatus;

const AStatusText: string);

procedure Button1Click(Sender: TObject);

private

public

{ Public declarations }

end;

var

Form3: TForm3;

implementation

{ \$R *.dfm }

procedure TForm3.btnSendMailClick(Sender: TObject);

begin

Memo1.Clear;

SMTP.Host:=ledHost.Text;

SMTP.Port:=25;

Smtp.AuthenticationType:=atLogin; // atNone

Smtp.Username:=LabeledEdit1.Text;

Smtp.Password:=LabeledEdit2.Text;

//

MailMessage.From.Name:=ledFrom.Text;

MailMessage.Subject:=ledSubject.Text; // тема

MailMessage.From.Address:=ledFrom.Text; // адреса відправника

MailMessage.Recipients.EmailAddresses:=ledTo.Text+', '+ledCC.Text; // получатель + копия

MailMessage.Body.Text:=Memo2.Text; // текст повідомлення

if FileExists(ledAttachment.Text)

then TIdAttachment.Create(MailMessage.MessageParts,ledAttachment.Text);

try

try

SMTP.Connect(1000);

SMTP.Send(MailMessage);

ShowMessage('Лист надіслано!');

except on E:Exception do

begin

Memo1.Lines.Insert(0,'Статус відправлення: ERROR - '+E.Message);

ShowMessage('Лист не відправлено!');

```
end;  
end;  
finally  
if SMTP.Connected  
then SMTP.Disconnect;  
end;  
end;  
  
procedure TForm3.SMTPStatus(ASender: TObject; const AStatus: TIdStatus;  
const AStatusText: string);  
begin  
Memo1.Lines.Insert(0,'Статус відправлення: '+AStatusText);  
end;  
  
procedure TForm3.Button1Click(Sender: TObject);  
begin  
if AttachmentDialog.Execute  
then ledAttachment.Text:=AttachmentDialog.FileName;  
end;  
  
end.
```

ДОДАТОК Б
Слайди презентації

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА
ПОЛІТЕХНІКА»

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
ДЛЯ ІНФОРМАЦІЙНОЇ ПІДТРИМКИ
ІНЖЕНЕРА З ПИТАНЬ ОБСЛУГОВУВАННЯ
ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Виконав: студент 4курсу, групи КНТ-118сп
Керівник: к.т.н., доц. Кафедри ПЗ

Строков В.М.
Степаненко О.О.

1

Рисунок Б.1 – Слайд №1

Об'єкт дослідження – методи проведення оновлення програмних засобів.

Предмет – способи інформаційної підтримки інженера.

Метою даної роботи є розроблення програмного забезпечення інформаційної підтримки інженера з питань обслуговування програмного забезпечення.

Для досягнення поставленої мети були поставлені наступні завдання:

- провести аналіз способів та методів оновлення програмного забезпечення;
- провести аналіз існуючих підходів для інформаційної підтримки сервіс-інженера;
- провести обґрунтування вибору методу рішення та інструментів;
- побудувати структурно-функціональну схему;
- побудувати алгоритми роботи програми;
- розробити програму для інформаційної підтримки сервіс-інженера з питань обслуговування програмного забезпечення.

2

Рисунок Б.2 – Слайд №2

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Таблиця 1 Структура файлу вхідних даних

Адреса пошти
Адреса сервера вхідної пошти (сервер POP3)
Логін
пароль
Протокол отримання пошти

Таблиця 2 – Табличний вигляд звіту «Вхідні електронні листи»

Автор листа	Електронна адреса пошти	Тема листа	Сервер	Текст повідомлення	Прикріплені й файл	Дата оновлення

3

Рисунок Б.3 – Слайд №3

ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

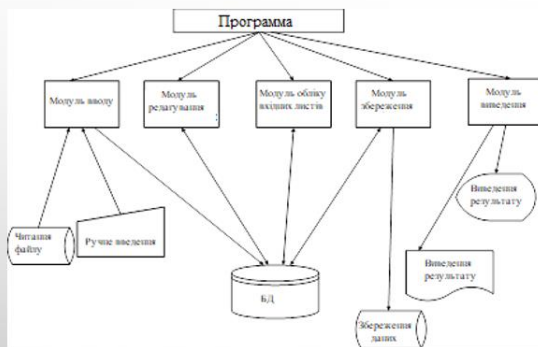


Рисунок 1 – Схема структурно-функціональних модулів програми «Облік вхідних електронних листів».

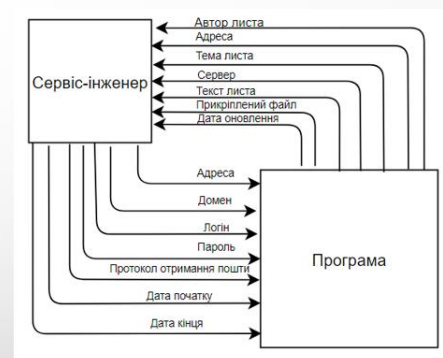


Рисунок 2 - Контекстна діаграма програми «Облік вхідних електронних листів»

4

Рисунок Б.4 – Слайд №4

ДІАГРАМИ ПОТОКІВ ТА СТАНІВ ПРОГРАМИ

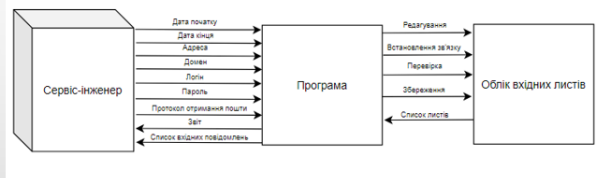


Рисунок 3 – Діаграма потоків даних програми «Облік вхідних електронних листів».

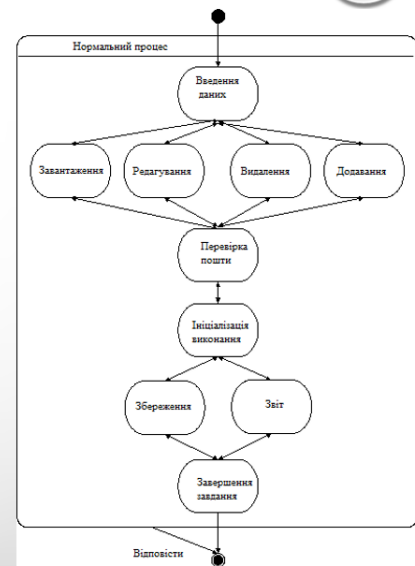


Рисунок 4 – Діаграма станів програми «Облік вхідних електронних листів»

Рисунок Б.5 – Слайд №5

ДЕТАЛІЗОВАНА ДІАГРАМА ПРОГРАМИ

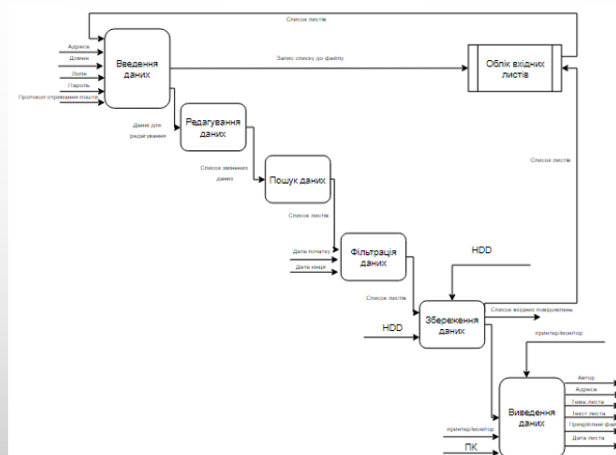


Рисунок 5 – Деталізована діаграма програми «Облік вхідних електронних листів».

Рисунок Б.6 – Слайд №6

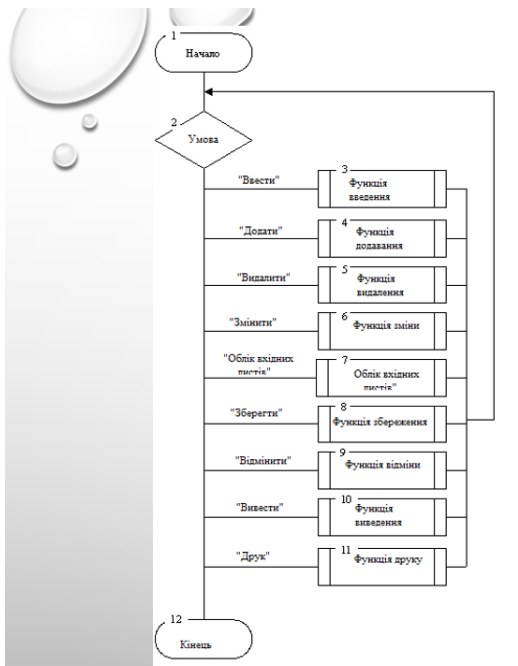


Рисунок 6 – Структурна схема керуючої програми

РОЗРОБКА ТА ОПИС АЛГОРИТМУ

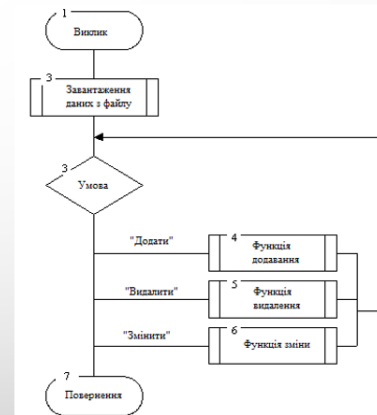


Рисунок 7 – Структурна схема функції «Введення»

Рисунок Б.7 – Слайд №7

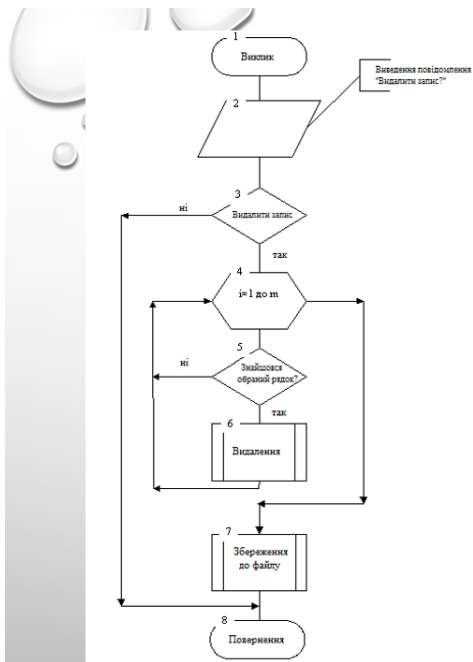


Рисунок 8 – Структурна схема функції «Видалити»

РОЗРОБКА ТА ОПИС АЛГОРИТМУ

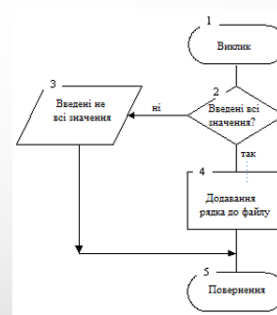


Рисунок 9 – Структурна схема функції «Додати»

Рисунок Б.8 – Слайд №8

РОЗРОБКА ТА ОПИС АЛГОРИТМУ



Рисунок 10 – Структурна схема функції «Виведення»

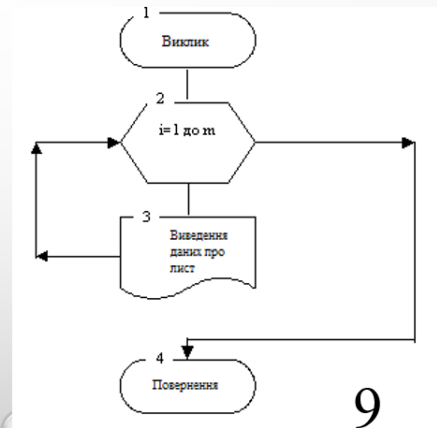


Рисунок 11 – Структурна схема функції «Друк»

Рисунок Б.9 – Слайд №9

ТЕСТУВАННЯ ПРОГРАМИ

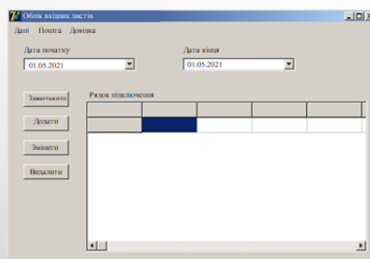


Рисунок 12 – Вид екранної форми «Облік вхідних листів»

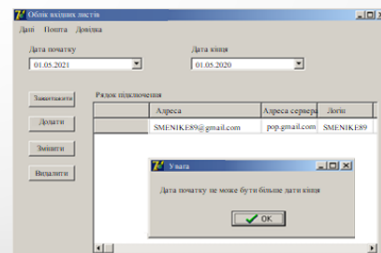


Рисунок 13 – Вид екранної форми «Редагування» при тестуванні функції Видалити

Рисунок Б.10 – Слайд №10

ТЕСТУВАННЯ ПРОГРАМИ

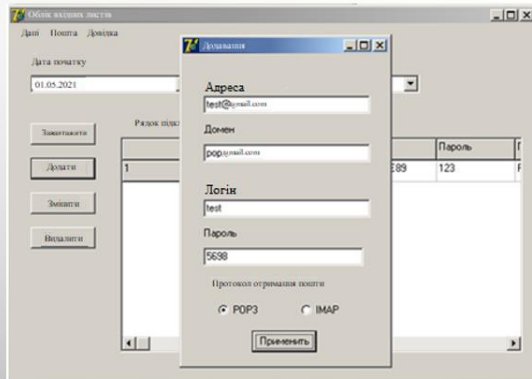


Рисунок 14 – Вид екранної форми «Додавання»

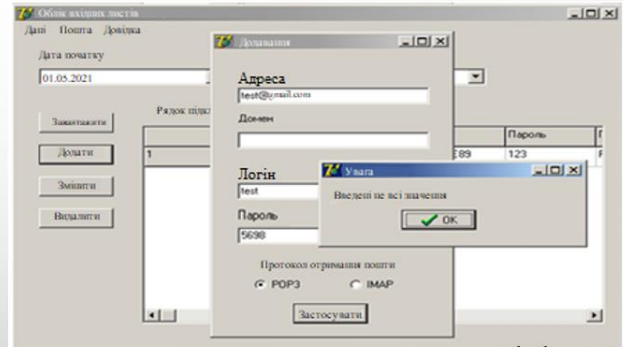


Рисунок 15 – Вид екранної форми повідомлення про помилку

11

Рисунок Б.11 – Слайд №11

ТЕСТУВАННЯ ПРОГРАМИ

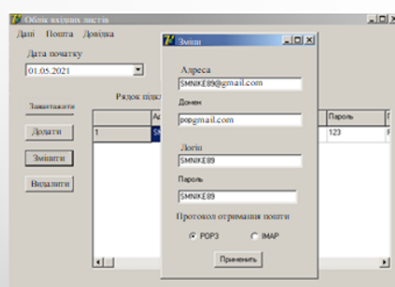


Рисунок 16 – Вид екранної форми «Зміна»

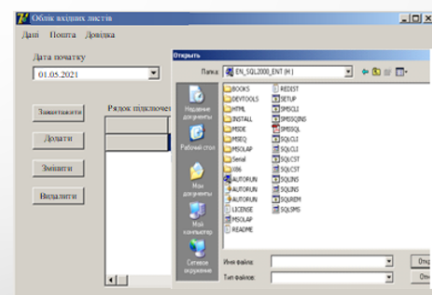


Рисунок 17 – Вид екранної форми «Відкрити»

12

Рисунок Б.12 – Слайд №12

ТЕСТУВАННЯ ПРОГРАМИ

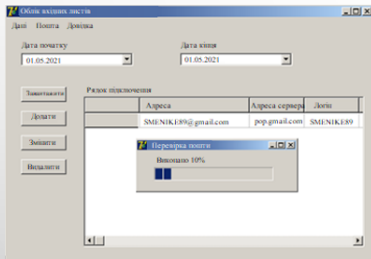


Рисунок 18 – Вид екранної форми «Перевірка пошти»

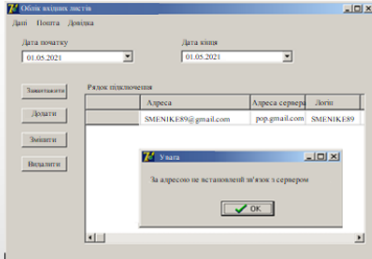


Рисунок 19 – Вид екранної форми повідомлення користувачу про помилку

13

Рисунок Б.13 – Слайд №13

ТЕСТУВАННЯ ПРОГРАМИ

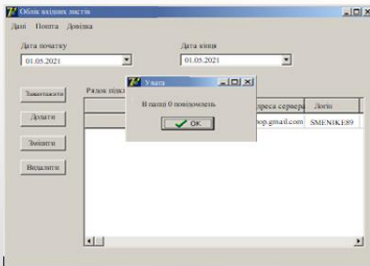


Рисунок 20 – Вид екранної форми повідомлення користувачу про помилку

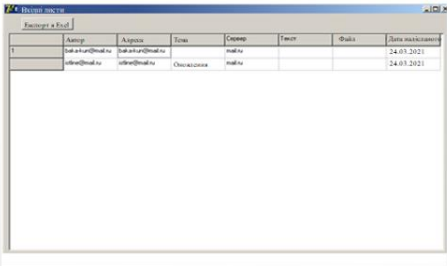


Рисунок 21 – Вид екранної форми «Вхідні електронні листи»

14

Рисунок Б.14 – Слайд №14

ВИСНОВОК

В даному дипломному проєкті була спроектована і реалізована програма обліку вхідних електронних листів на малому підприємстві. Дана програма забезпечує наступні основні функції: з'єднання з сервером по рядках підключень; перевірку наявності вхідних повідомлень; перевірку дати, зазначеної в листі, на входження в зазначений діапазон дат; формування результату: звіту входять електронних листів за період.

Реалізація системи проводилася відповідно до вимог замовника. Вибір середовища програмування був обумовлений також основними характеристиками більшості використовуваного програмного забезпечення на малих підприємствах. При розробці програми використовувалися Borland Delphi 9.0. В процесі написання програми велика увага була приділена зручності роботи користувача і побудови дружнього інтерфейсу.

Введення програми забезпечить автоматизований безпомилковий облік всіх електронних листів, що містять відомості про оновлення, дасть можливість формування достовірної та повної інформації про всіх клієнтів, що здійснили поновлення для ПЗ.

В ході виконання дипломного проєкту були виконані наступні завдання:

- проведено аналіз способів та методів оновлення програмного забезпечення;
- проведено аналіз існуючих підходів для інформаційної підтримки сервіс-інженера;
- проведено обґрунтування вибору методу рішення та інструментів;
- побудовано структурно-функціональну схему;
- побудовано алгоритми роботи програми;
- розроблено програму для інформаційної підтримки сервіс-інженера з питань обслуговування програмного забезпечення.

15

Рисунок Б.15 – Слайд №15