Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет «Комп'ютерні технології і системи»
(назва факультету)

Кафедра «Електронні обчислювальні машини»
(повна назва кафедри)

Пояснювальна записка

до кваліфікаційної роботи
бакалавра
(ступінь вищої освіти)

на тему: Розробка комплексу засобів стеганографічного захисту інформації. Стеганографічний захист інформації з використанням графічних контейнерів

за освітньою програмою Кібербезпека

зі спеціальності: 125 Кібербезпека
(шифр і назва спеціальності)

Виконав: студент групи: КБ1811

(підпис студента) / Віра МАСЛЮК /
(Ім'я ПРІЗВИЩЕ)

Керівник: (підпис) / доцент, Денис ОСТАПЕЦЬ /
(посада, Ім'я ПРІЗВИЩЕ)

Нормоконтролер: (підпис) / ст. викладач, Володимир ДЗЮБА/
(посада, Ім'я ПРІЗВИЩЕ)

Консультанти:

(назва розділу) (підпис) / (посада, Ім'я ПРІЗВИЩЕ) /

(назва розділу) (підпис) / (посада, Ім'я ПРІЗВИЩЕ) /

(назва розділу) (підпис) / (посада, Ім'я ПРІЗВИЩЕ) /

Засвідчую, що у цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент (підпис)

Дніпро – 2022 рік

Ministry of Education and Science of Ukraine
Ukrainian State University of Science and Technologies

Faculty «Computer technologies and systems»
(faculty)

Department «Electronic computers»
(department)

Explanatory Note
to Bachelor's Thesis

(higher education degree)

on the topic: Development of a set of means of steganographic protection of information. Steganographic protection of information using graphic containers

according to educational curriculum Cybersecurity

in the Speciality: «125 Cybersecurity»
(speciality and its code)

Done by the student of the group: KB1811 / Vira Masliuk /
(name, surname)

Scientific Supervisor: / Associate Professor Denis Ostapets /
(position, name, surname)

Normative controller : / Senior lecturer Volodymyr Dziuba /
(position, name, surname)

Supervisors

(Chapter title heading)	/ / (position, name, surname)
(Chapter title heading)	/ / (position, name, surname)
(Chapter title heading)	/ / (position, name, surname)
(Chapter title heading)	/ / (position, name, surname)

Dnipro – 2022

Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет: Комп'ютерні технології і системи
Кафедра: Електронні обчислювальні машини
Рівень вищої освіти: Перший (бакалаврський)
Освітня програма: Кібербезпека
Спеціальність: 125 Кібербезпека
(шифр та назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
(підпис) Ігор ПЦуканицький
(Ім'я ПРІЗВИЩЕ)
Дата _____

ЗАВДАННЯ

на кваліфікаційну роботу бакалавра
(ступінь вищої освіти)

студенту Маслюк Вірі Олексіївні
(Прізвище. Ім'я По батькові)

1. Тема роботи: Розробка комплексу засобів стеганографічного захисту інформації. Стеганографічний захист інформації з використанням графічних контейнерів.
Керівник роботи: Остапець Денис Олександрович, к.т.н., доцент
(Прізвище, Ім'я, По батькові, науковий ступінь, вчене звання)

затверджені наказом від «07» 12 2021 р. № 67ст

2. Строк подання студентом роботи: 13.06.2022 р.
3. Вихідні дані до роботи: Методи стеганографічного захисту інформації; Формати графічних файлів-контейнерів
4. Зміст пояснювальної записки (перелік питань, які потрібно опрацювати):
4.1 Аналітична частина:
Аналіз методів стеганографії, що використовують графічні контейнери
4.2 Основна частина:
- Огляд методів та засобів стеганографічного захисту інформації
- Інформаційна структура та режими роботи комплексу
- Розробка програмного забезпечення комплексу
- Інструкція з використання комплексу
4.3 Охорона праці та захист навколишнього середовища: -

4.4 Економічна частина: -

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
- Порівняльний аналіз методів стеганографії;
- Структури даних;
- Склад та функції комплексу;
- Основні алгоритми програми;
- Приклади роботи комплексу.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Завдання видав: (підпис консультанта, дата)	Завдання прийняв: (підпис студента, дата)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Огляд методів та засобів стеганографічного захисту інформації	25.04.22	20%
2	Інформаційна структура та режими роботи комплексу	11.05.22	30%
3	Розробка та налагодження програмного забезпечення комплексу	01.06.22	40%
4	Інструкція з використання комплексу	06.06.22	5%
5	Реферат, вступ, висновки	13.06.22	5%
6	Подання кваліфікаційної роботи до кафедри	13.06.22	
7	Захист кваліфікаційної роботи на засіданні Екзаменаційної комісії	23.06.22	

Студент _____ Віра МАСЛЮК
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник роботи _____ Денис ОСТАПЕЦЬ
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалавра: 73 с., 36 рис., 6 табл., 13 додатків, 8 джерел.

Об’єкт розробки – засоби стеганографічного захисту інформації з використанням графічних контейнерів за методом заміни найменш значущих бітів.

Мета роботи – розробка програмного комплексу, який реалізує та демонструє методи стеганографічного захисту інформації з використанням графічних контейнерів.

Приведено опис та порівняльну характеристику методів стеганографії з використанням графічних контейнерів. Обґрунтовано вибір методу найменш значущих бітів. Описано структури даних та функціонування комплексу. Розроблені блок-схеми узагальнених алгоритмів роботи комплексу в режимах вбудовування та вилучення повідомлень. Написано та відлагоджено програмне забезпечення комплексу, перевірено на працездатність. Подана інструкція з використання комплексу.

Результати роботи можуть бути використані у навчальному процесі студентів відповідних спеціальностей при проведенні лабораторних і практичних робіт.

Ключові слова: СТЕГАНОГРАФІЯ, ЗАХИСТ ІНФОРМАЦІЇ, ГРАФІЧНИЙ КОНТЕЙНЕР, BMP-24, МЕТОД ЗАМІНИ НАЙМЕНШ ЗНАЧУЩИХ БІТІВ, LSB, C#, AES128.

ЗМІСТ

ВСТУП

В ході популяризації мультимедійних технологій, розвиток стеганографії вийшов на новий етап, названий комп'ютерною стеганографією. Проте наразі лише деякі її напрямки набувають широкого використання.

Стеганографія являє собою перспективну галузь в напрямку збереження та передачі інформації з обмеженим доступом, оскільки додаткове приховування самого факту існування такої інформації, значно підвищить її захист. Дана робота стосується стеганографічного приховування інформації у графічних контейнерах, тому її тема є актуальною.

Тема роботи затверджена наказом № 67 ст від 07.12.2021.

Мета роботи - розробка програмного комплексу, який реалізує та демонструє методи стеганографічного захисту інформації з використанням графічних контейнерів.

Основні положення цієї роботи були представлені та схвалені на XV Міжнародній конференції «Сучасні інформаційні та комунікаційні технології на транспорті, в промисловості та освіті», а також на 81 Всеукраїнській науково-технічній конференції молодих учених, магістрантів та студентів «Наука і сталий розвиток транспорту» у 2021 році (див. Додаток А та Додаток Б).

1 ОГЛЯД МЕТОДІВ ТА ЗАСОБІВ СТЕГАНОГРАФІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ

1.1 Загальні відомості

Незважаючи на те, що тема стеганографії була описана в численних публікаціях та обговорювалася на щорічних конференціях, доволі довго цей напрям не мав чіткої термінології. У 1996 р. на 1-й Міжнародній конференції з приховування даних [1] (Information Workhop on Information Hiding`96) були узгоджені основні поняття стеганографії. А вже у 2002 р. відбулася перша конференція, що була присвячена стеганографії.

За [1] стеганографія – це наука, що вивчає способи та методи приховання конфіденційних відомостей; задача стеганографії, на відміну від криптографії, це приховання самого факту існування, зберігання та передачі деякого секретного повідомлення.

Серед напрямів використання комп'ютерної стеганографії можна виділити два основних [1]:

– пов'язаний із цифровою обробкою сигналів (далі ЦОС) – в цьому напрямку використовуються методи стеганографії, які вбудовують секретне повідомлення у цифрові дані аналогової природи (тобто це зображення, аудіо- та відеозаписи);
– непов'язаний з ЦОС – в цьому напрямку використовуються методи, що розміщують приховане повідомлення в заголовках файлів або пакетів даних. Напрям не став популярним через легкість виявлення та знищення вбудованої інформації.

За [1] можна зробити висновок, що стеганографія активно застосовується при вирішенні наступних завдань:

– забезпечення захисту конфіденційних даних;
– встановлення та захист авторських прав;
– управління мережевими ресурсами;

– приховування деякого програмного забезпечення (зазвичай вірусного ПО), а також створення деяких прихованих каналів для передачі інформації.

Стеганографія охоплює ще ряд термінів, пояснення яких, за джерелом [2], подано нижче:

– повідомлення – послання, яке належить приховати;
– контейнер (стегоконтейнер) – будь-який об’єкт, який використовується для таємного вбудовуваня повідомлення;
– стегоканал – канал для передачі стегоконтейнера;
– ключ – ключ для отримання прихованого змісту з контейнера.

Стеганосистема або стеганографічна система – це сукупність засобів і методів, що служать для створення прихованого каналу для передачі деякої інформації. Така система [1] (рисунок 1.1) має виконувати як задачу вбудовування деякої таємної інформаційної послідовності, так і виокремлення цієї послідовності від іншої послідовності, у яку було вбудовано таємне повідомлення.

Існує декілька типів стеганосистем [3], які обумовленні типом/наявністю ключа:

– безключові системи;
– з секретним ключем;
– з відкритим ключем;
– системи змішаного типу.

В стеганосистемі з секретним ключем використовується лише один ключ, який створюється перед відправкою стеганофайлу і передається по захищеному каналу зв’язку. В стеганосистемі з відкритим ключем, для вбудовування та витягання повідомлення з контейнера застосовуються два ключі – секретний та відкритий.

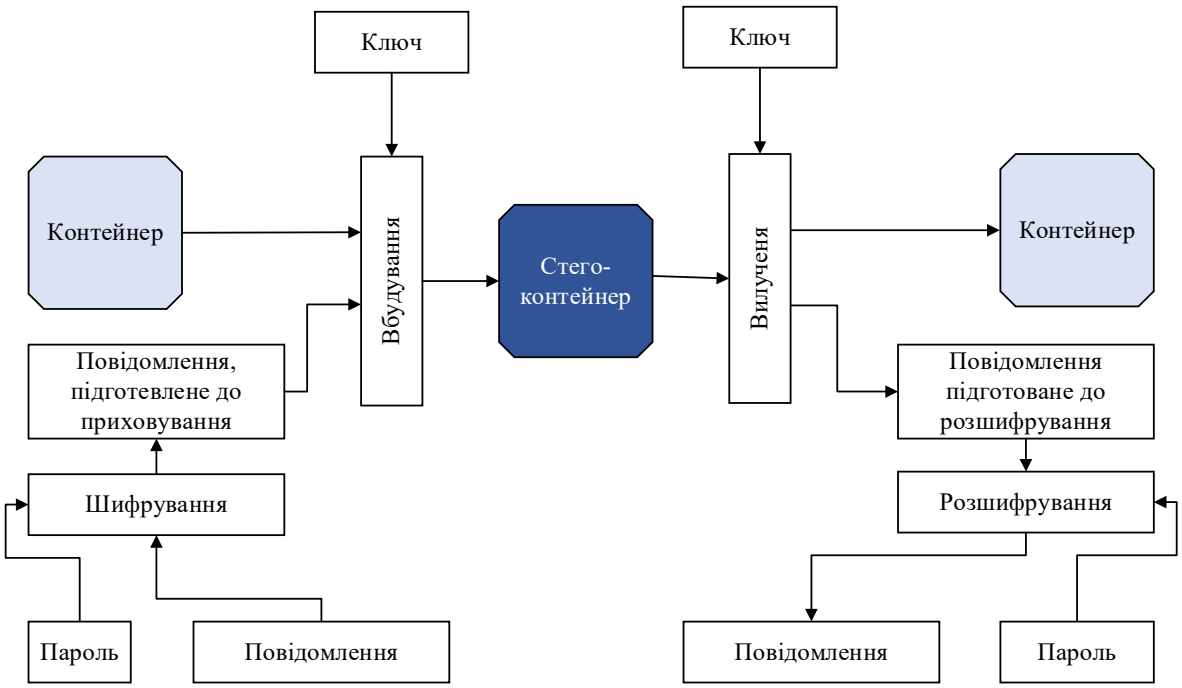

Рисунок 1.1 – Графічне представлення стеганографічної системи

При створенні стеганосистеми потрібно дотримуватися наступних вимог:

– у алгоритму вбудовування/видобування таємного повідомлення у контейнер має бути прийнятна обчислювальна складність;
– метод, за яким виконується приховування інформації, має не порушувати автентичність та цілісність вбудованого повідомлення;
– рекомендується наявність ключа, за допомогою якого тільки власник зможе видобути повідомлення з контейнеру, навіть якщо зловмисник встановив факт передачі.

Контейнери бувають [3] двох основних типів: потокові та фіксовані.

Потоковий контейнер це безперервно змінна послідовність бітів, що не ділиться на блоки або якісь структури з фіксованою довжиною. Основна проблема [3], що виникає при використанні такого типу контейнера, це визначення початку та кінця прихованого повідомлення. Розробка систем з використанням потокових контейнерів поки що не поширена, хоча має перспективи реалізації у застосуванні при мобільній розмові.

Фіксований контейнер, на відміну від потокового, має заздалегідь відомі характеристики, що надає значні переваги. У цій роботі буде розглядатися фіксований тип контейнера.

Контейнер називається пустим, якщо він не містить у собі ніякої прихованої інформації. Якщо ж у контейнер вже помістили деяке секретне повідомлення, то такий контейнер називається заповненим. Стеганофайлом називають файл, що представляє собою заповнений контейнер. Стеганофайлами можуть бути текстові файли, аудіо файли, відео файли, зображення та інші.

Порівняльна характеристика різних контейнерів, при використанні їх у стеганографії наведено в таблиці 1.1. В цій таблиці використано умовні позначення:

– H – високий рівень оцінки за параметром;
– M – середній рівень;
– L – низький рівень.

Таблиця 1.1 – Порівняльна характеристика контейнерів

Тип контейнера	Кількість методів приховування	Якість приховування	Розповсюдженість
Текстові файли	L	L	L
Аудіо файли	M	H	M
Зображення	H	H	H

Дана робота присвячена стеганографії на зображеннях.

1.2 Огляд методів стеганографії з використанням графічних контейнерів

Метод заміни найменш значущого біта (LSB - Least Significant Bit) – цей метод є доволі розповсюдженим і в своїй основі має просту для розуміння та реалізації ідею. За цим методом повідомлення вбудовується в молодші біти контейнера. Чим менше задіяно молодших бітів пікселя для приховування бітів повідомлення, тип менш помітним буде результат. Всього можна вбудовувати по

3 біти повідомлення на один піксель контейнера. Можливо використати і більше, але це буде збільшувати вірогідність виявлення наявності повідомлення.

Псевдовипадкові перестановки – за цим методом підмішуваннння бітів таємного повідомлення [4] відбувається за деяким псевдо випадковим законом, тобто порядок появи бітів повідомлення буде змінним, а не постійним. Такий підхід робить більш затратним дії по виявленню факту наявності та розшифровку деякої інформації у контейнері, особливо якщо використовується якісний алгоритм ГПЧ. Також потрібно уникати запис бітів у старші розряди пікселів, тому за аналогією до методу LSB, біти повідомлення записуються у молодші біти та можуть бути пошкоджені. В даному методі є деякі недоліки, пов'язані з використанням ГПЧ, оскільки генератор може видати декілька разів одну й ту ж адресу для запису біту повідомлення, що може спричинити перезапис біту. Розв'язати цю проблему можна [4] за використання додаткового файлу, в якому будуть записуватися усі сгенеровані адреси і при генерації нової адреси буде перевірятися наявність її у файлі. Для даного методу формат контейнеру не має особливого значення.

Метод з використанням патчів – один з методів статистичного кодування інформації шляхом зміни деяких статистичних якостей контейнера. Наприклад, це може бути [4] додавання деяких додаткових даних до початкового повідомлення, а потім «розмивання» цього повідомлення по зображенню з використанням гаусового розподілу. При витягуванні повідомлення використовується перевірка гіпотез. Також в методі використовується секретний ключ, що приміняється до деякої підмножини пікселів, яка обирається випадковим чином з зображення. Наступним кроком ця підмножина поділяється на два, так званих, патчі. В підмножині першого патча яскравість пікселів змінюється на позитивне число, а в підмножині другого патча – на негативне число.

Метод приховування інформації у молодших бітах палітри – метод схожий на LSB [4] і по суті є його варіацією. Відмінність від LSB класичного полягає у тому, що повідомлення приховується не у біти контейнера, а у найменш значущі

біти палітри. Цей метод не підходить для вбудовування таємної інформації великих розмірів, оскільки очевидна мала ємність контейнера.

Метод приховування інформації у полях формату – метод простий але і не надійний, оскільки деякі програми для перегляду зображень дозволяють переглядати службові поля заголовку. Суть методу полягає у записі повідомлення до поля формату контейнера. Також цей метод, як і попередній, не використовується для зберігання великих за розміром повідомлень.

GLM (англ. Grey Level Modification — Зміна рівня сірого) – метод, за яким потрібно змінювати кратність (парність/непарність) значення яскравості чорно білого зображення [4]. Таємне повідомлення записується до кожного пікселю зображення по 1 біту на піксель.

Вбудовування відбувається наступними етапами [4]:

– значення яскравості усіх пікселів роблять парними, шляхом зміни всіх непарних значень на 1;
– парність отриманих значень порівнюється з парністю бітів даних: якщо перший біт даних парний (рівний 0), то перший піксель не змінюється; якщо ж він не парний (рівний 1), то значення яскравості змінюється на непарне.

Витягання відбувається наступними етапами [4]:

– для кожного пікселя, що має в собі вбудоване повідомлення, визначається значення яскравості;
– якщо значення парне – то відповідний біт повідомлення рівний 0, якщо не парне – 1.

1.3 Порівняльний аналіз методів стеганографії з використанням графічних контейнерів

У таблиці 1.2 представлена порівняльна характеристика розглянутих методів приховування повідомлення у зображеннях. Використані умовні позначення: H – високий рівень оцінки за параметром, M – середній, L – низький рівень.

Таблиця 1.2 – Порівняльна характеристика методів стеганографії для зображень

Назва методу	Рівень приховування для зору людини	Складність методів розпізнавання наявності повідомлення	Ємність контейнера	Складність реалізації методу
Метод заміни найменш значущого біта	H	L	H	L
Псевдовипадкові перестановки	H	H	L	M
Метод з використанням патчів	H	H	L	H
Метод приховування інформації у молодших бітах палітри	H	L	L	L
Метод приховування інформації у полях формату	L	L	L	L
Зміна рівня сірого	H	L	M	L

Проаналізувавши різні методи комп’ютерної стеганографії для зображень, можна сказати, що найлегшим по усім параметрам є метод приховування інформації у полях формату, а найскладнішим – метод з використанням патчів. Найоптимальнішими для реалізації в рамках даної роботи є методи LSB і GLM, але метод GLM обмежує своє використання тим, що працює з чорно-білими зображеннями. Тому для реалізації обрано метод заміни найменш значущого біта.

Метод LSB може бути реалізований на графічних контейнерах різних типів, але найбільш зручно та доречно в даній роботі використати контейнер типу BMP, оскільки цей тип має просту та зрозумілу побудову файлу.

1.4 Огляд популярних програмних засобів стеганографії з використанням графічних контейнерів

В даному пункті розглянемо наступне програмне забезпечення для стеганографії з використанням графічних контейнерів: Anubis, Hallucinate, JHide та OpenStego. Їх опис та скріншоти інтерфейсу наведені у Додатках В – Е . При досліджені функціоналу цих програм, було визначено, що більшість працює в рамках різних варіацій методу запису LSB для різних типів графічних контейнерів.

Можна виділити основний функціонал, що зазвичай присутній у стеганографічному ПЗ:

– вибір контейнера для вбудовування або вилучення повідомлення;
– вибір файлу повідомлення для вбудовування або шляху для зберігання вилученого файлу;
– попереднє шифрування файлу користувацьким паролем;
– введення шляху та імені файлу для збереження заповненого контейнера.

1.5 Висновки за розділом

Розглянуті основні визначення, що охоплюються темою стеганографії. Подано короткий опис популярних стеганографічних методів, що використовують графічний контейнер. Складена та проаналізована порівняльна характеристика описаних методів. Обрано для реалізації метод заміни найменш значущого біта. Розглянуті готові програмні рішення з використанням стеганографії для графічних контейнерів та описано їх загальний функціонал.

2 ІНФОРМАЦІЙНА СТРУКТУРА ТА РЕЖИМИ РОБОТИ КОМПЛЕКСУ

2.1 Інформаційна структура

Для реалізації функцій стеганографічного вбудовування секретного повідомлення, обрано в якості контейнера зображення типу BMP. Для обраного контейнера є важливим параметр глибини кольору, оскільки за різних значень цього параметра присутні відмінності у структурі файлу. Оптимальними значеннями глибини кольору для вбудовування повідомлення є 24, 48 та вище бітів на піксель. Від значення глибини кольору залежить розмір контейнера, а великі файли в першу чергу викликають підозру. Тому обрано BMP з глибиною кольору 24 біти/піксель. Загальна структура такого файлу [1] подана на рисунку 2.1, де значення Size відповідає за кількість байт, що займають пікселі зображення.

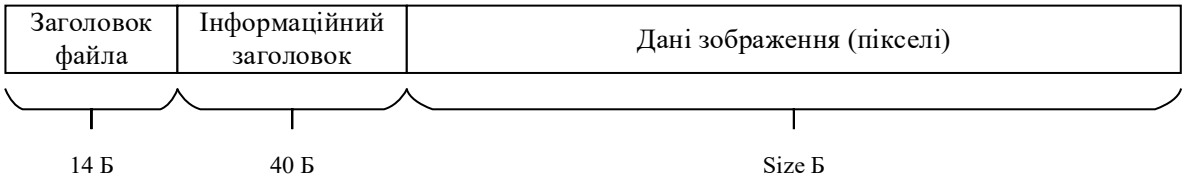

Рисунок 2.1 – Графічне представлення структури файлу типу BMP-24

Детальна інформація [1], щодо структури BMP-24, міститься у таблиці 2.1.

Таблиця 2.1 – Структура файлу типу BMP-24

Ім'я	**Довжина в байтах**	**Опис**
1	**2**	**3**
Заголовок файла (BitMapHeader)		
Type	2	Сигнатура формату, для файлу BMP має значення «BM» (при перегляді вмісту файлу текстом виглядає як пара символів ASCII)
Size	4	Розмір файла в байтах
Reserved 1	2	Зарезервовано
Reserved 2	2	Зарезервовано
OffsetBits	4	Зміщення, де можна знайти масив пікселів (дані про растр)

Продовження таблиці 2.1

1	2	3
Інформаційний заголовок (BitMapInfoHeader)		
Size	4	Довжина інформаційного заголовка
Width	4	Ширина зображення в пікселях
Height	4	Висота забраження в пікселях
Planes	2	Кількість площин (у форматі BMP може бути тільки значення «1»)
BitCount	2	Глибина кольору, бітів на точку
Compression	4	Тип компресії (0 – незжате зображення)
SizeImage	4	Розмір зображення, байт
XpelsPerMeter	4	Горизонтальна роздільна здатність, піксель на метр
YpelsPerMeter	4	Вертикальна роздільна здатність, піксель на метр
ColorsUsed	4	Кількість використаних кольорів (в даному випадку поле заповнено значенням «0» тому, що палітра не використовується)
ColorsImportant	4	Кількість основних кольорів (тут також значення «0», бо всі кольори важливі)
Дані зображення (BitMap Array)		
Image	Size	Зображення, записане рядками зліва направо і знизу

Проаналізувавши методи стеганографії для зображень та обраний тип контейнера, вирішено реалізувати метод вбудовування LSB. Оскільки таким методом можна відносно непомітно вписати у піксель декілька бітів повідомлення, то у створеній програмі надана можливість вибору між одним або трьома бітами на піксель. Візуалізація таких варіантів вбудовування бітів повідомлення у піксель представлена на рисунках 2.2 – 2.3.

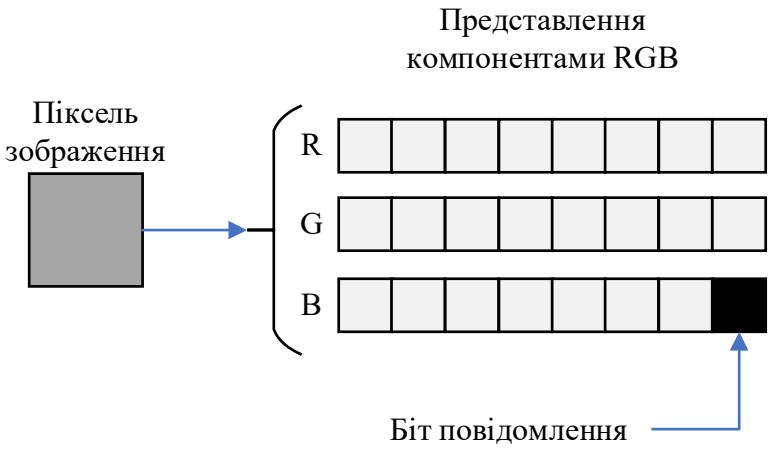

Рисунок 2.2 – Вбудовування одного біта на піксель

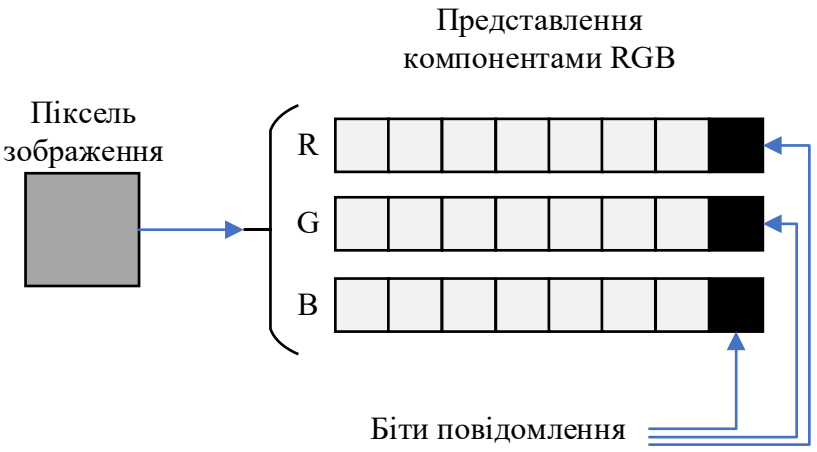

Рисунок 2.3 – Вбудовування трьох бітів на піксель

На прикладі трьох пікселів продемонстровано вбудовування числа 181 (в двійковому вигляді це 1011 0101) для трьох бітів на піксель (рисунок 2.4). Щоб сховати число 181 по одному біту, нам знадобиться 8 пікселів.

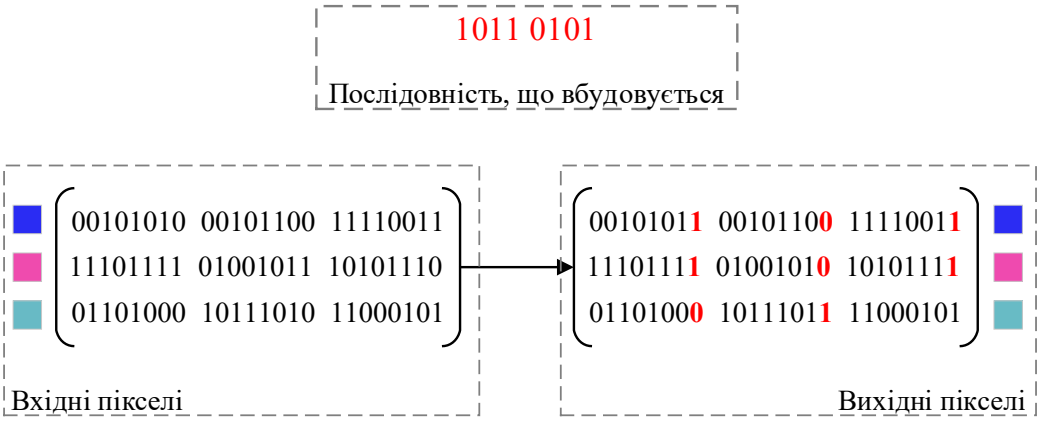

Рисунок 2.4 – Приклад вбудовування

Як видно з рис. 2.4, зміна кольору непомітна для людського ока, навіть при заміні трьох бітів для кожного пікселя. Але таке заповнення можуть з легкістю розпізнати найпростіші методи стеганоаналізу. Тому прийняте рішення про створення можливості записувати повідомлення до контейнера в пікселі не тільки підряд, а й у рівномірному розподілі по зображенню. Тобто відстань між пікселями з бітами повідомлення буде розраховуватись в залежності від розмірів зображення.

Також надано можливість зашифрувати повідомлення криптографічними методами перед його вбудуванням. Для реалізації цієї можливості обрано алгоритм AES 128. AES – це симетричний алгоритм блочного шифрування [5], який був прийнятий як стандарт у 2002 році урядом США. Розмір ключа для цього алгоритму може бути 128/196/256 біт, розмір блоку для шифрування складає 128 біт. У роботі вирішено використати ключ рівний 256 біт.

Для того, щоб передати дані про налаштування, обрані при вбудовуванні повідомлення у контейнер, до контейнера також записана деяка технічна інформація, що представлена набором даних сталого розміру. Далі ці дані будемо називати преамбулою. Структура преамбули показана на рисунку 2.5.

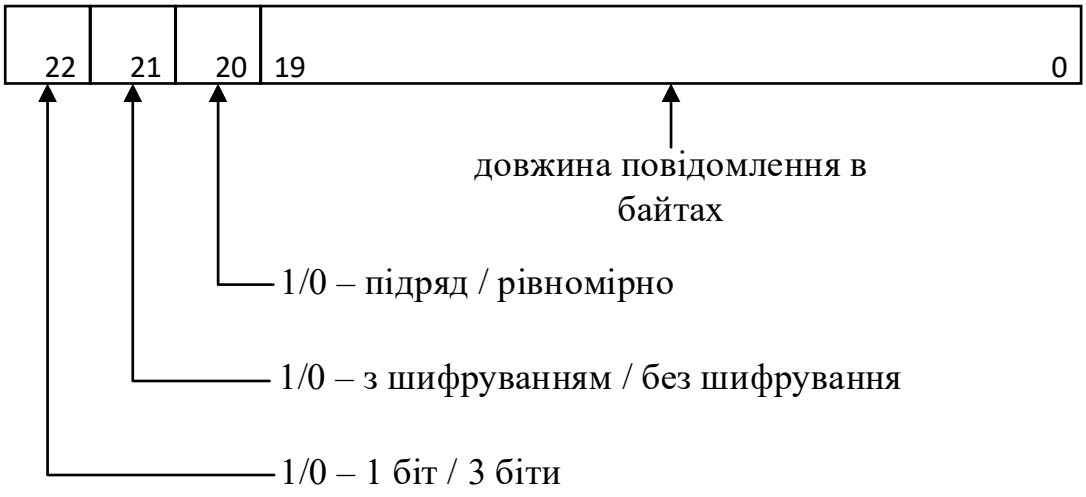

Рисунок 2.5 – Структура преамбули

Наприклад, преамбула «11000000 00001000 0110010» читається як:

– біти повідомлення записувались по одному на піксель;

– повідомлення було попередньо зашифроване;
– приховування виконано по 3 біти на піксель;
– розмір записаного повідомлення складає 1074 Б.

Записується преамбула завжди в перші 23 пікселя зображення методом по одному біту в кожен піксель. Саме ж повідомлення вирішено починати вбудовувати з другого за преамбулою пікселя.

Технічне завдання для розробки представлено у Додатку П.

2.2 Режими роботи комплексу

Реалізована програма матиме змогу як вбудовувати в контейнер, так і вилучати з контейнера повідомлення. Також можна переглядати значення преамбули, деяку кількість даних контейнерів (пустого і заповненого) та вбудовані дані в шістнадцятирічному вигляді. Режими роботи комплексу подані на рисунку 2.6.

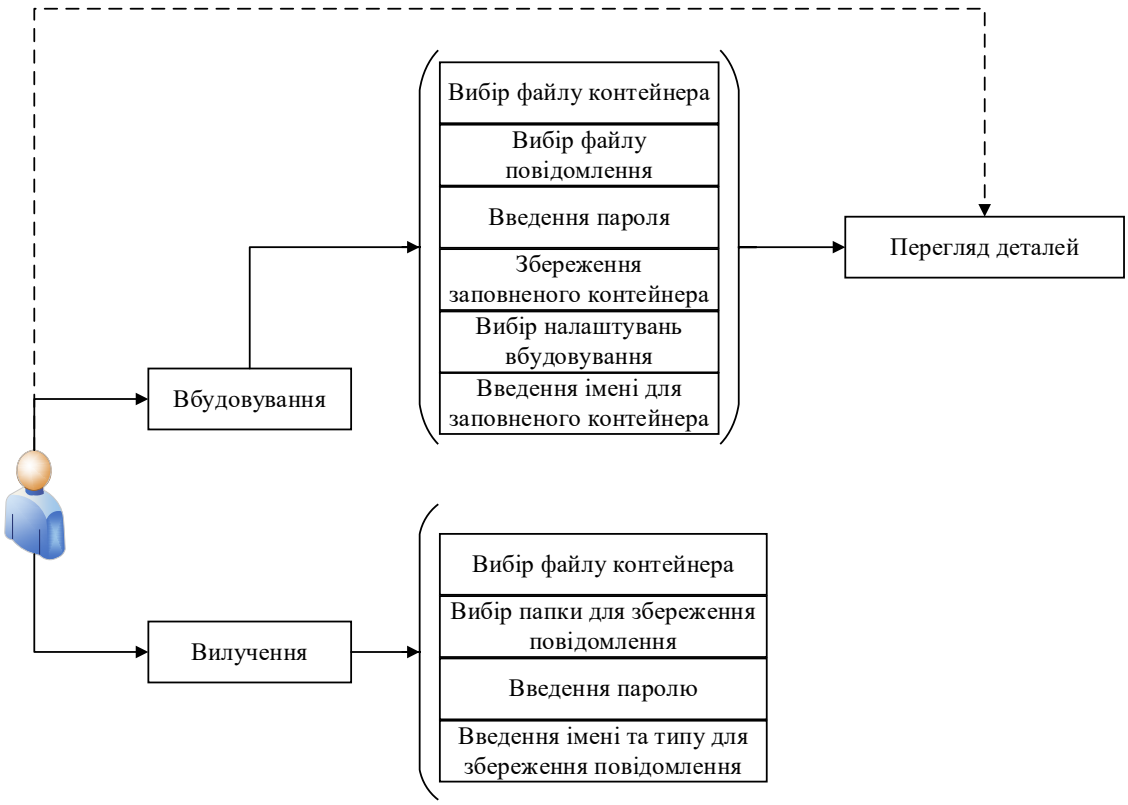

Рисунок 2.6 – Режими роботи комплексу

В режим перегляду деталей можна зайти тільки після вбудовування.

2.3 Висновки за розділом

Подана структура файлу BMP-24. Сформавано набір налаштувань для вбудовування повідомлення. Обрано алгоритм для шифрування повідомлення. Розроблена структура преамбули, що включає в себе обрані налаштування вбудовування та розмір прихованого повідомлення.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМПЛЕКСУ

3.1 Вибір середовища та засобів розробки

Серед аналогів готового програмного забезпечення, що виконує стеганографічні процеси, переважна більшість написана на мовах JAVA або C#.

Порівнюючи можливості цих мов, можливо виділити певні переваги мови C#. Ця мова є простішою в розумінні, оскільки до неї написано багато детальної документації. Також мова C# має велику кількість бібліотек, які не потребують додаткової установки. Тому для реалізації комплексу обрано мову C#.

Використані наступні бібліотеки:

1) *System.Drawing* – надає можливість використовувати основні графічні функції [6];
2) *System.Security.Cryptography* – надає велику кількість різноманітних служб [7] для шифрування та дешифрування, хешування та ін.

Також варто відмітити середовище розробки Visual Studio, що дозволяє писати програми на мові C# [8]. Завдяки інтерфейсу програмування додатків Windows forms, це середовище стає потужним інструментом розробки програм для ЕОМ сімейства ОС Windows.

3.2 Розробка програмного забезпечення комплексу в режимі вбудовування повідомлення

Блок-схема узагальненого алгоритму вбудовування повідомлення подана на рисунку 3.1. Вихідний код мовою C# наведено у Додатку Ж (див. процедуру buttonRun_Click). Код додаткових класів наведено у Додатках И - Л.

Блок 1 – початок процесу вбудовування повідомлення до контейнера.

Блок 2 – зчитування обраних налаштувань для вбудовування повідомлення.

Блок 3 – виділення пам’яті для збереження значення преамбули.

Блок 4 – відкриття файлу повідомлення, створення масиву bytes для збереження байтів файлу.

Блок 5 – запис до масиву bytes байтів повідомлення.

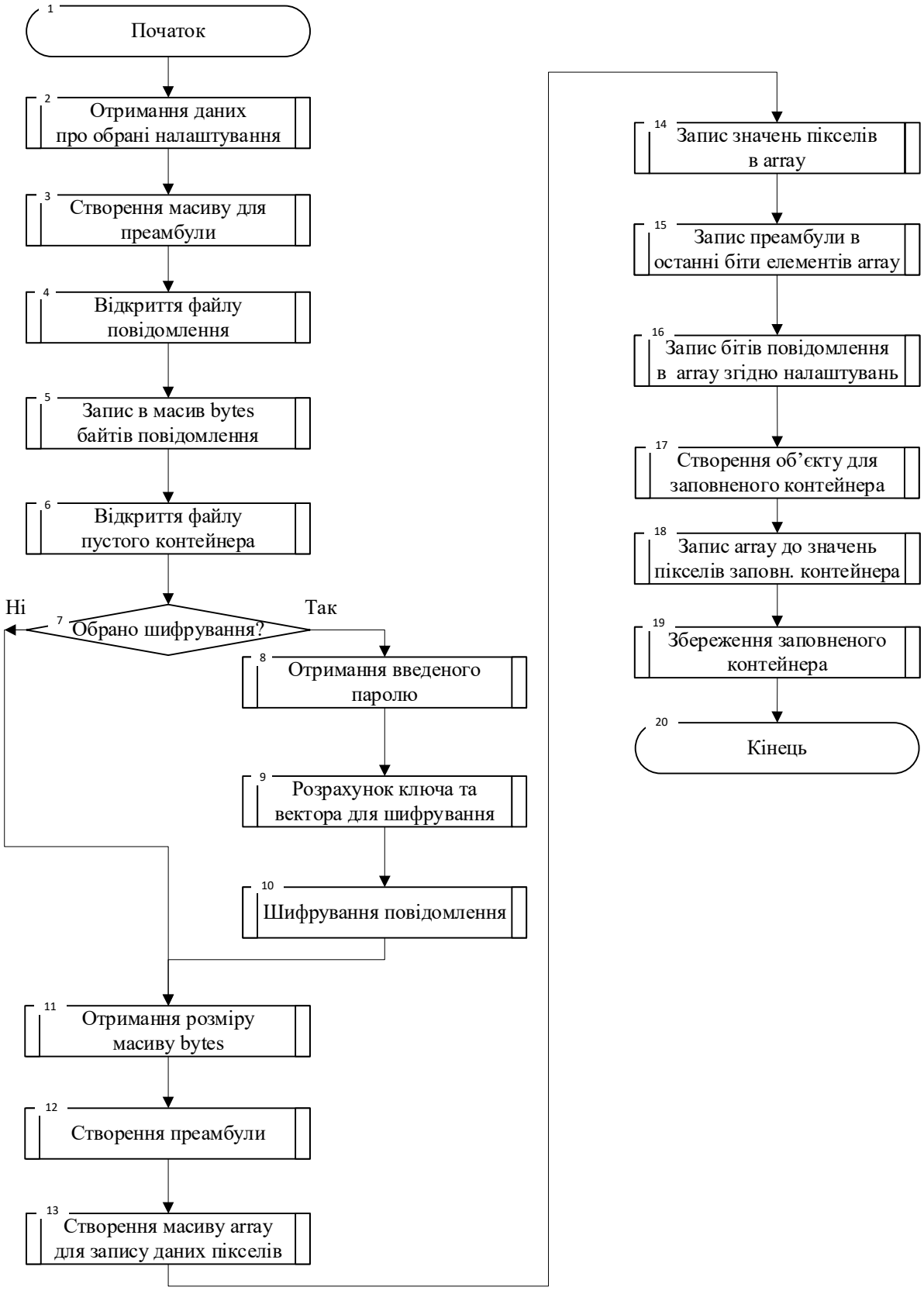

Рисунок 3.1 – Блок-схема узагальненого алгоритму вбудовування повідомлення

Блок 6 – відкриття файлу обраного пустого контейнера.

Блоки 7 – 10 – попереднє шифрування файлу повідомлення (якщо було обрано відповідне налаштування).

Блок 11 – отримання розміру масиву bytes.

Блок 12 – формування преамбули відповідно отриманих даних про налаштування та розмір вбудовуваного повідомлення.

Блок 13 – виділення пам’яті для збереження масиву array пікселів обраного пустого контейнера.

Блок 14 – заповнення масиву array значеннями пікселів пустого контейнера.

Блок 15 – вбудовування преамбули в значення елементів масиву array.

Блок 16 – вбудовування бітів повідомлення до елементів масиву array.

Блок 17 – створення графічного об’єкту для запису нових значень пікселів.

Блок 18 – запис масиву array в якості значень пікселів нового зображення.

Блок 19 – збереження заповненого контейнера.

Блок 20 – закінчення процесу вбудовування повідомлення.

3.3 Розробка програмного забезпечення комплексу в режимі вилучення повідомлення

Блок-схема узагальненого алгоритму вилучення повідомлення подана на рисунку 3.2. Вихідний код мовою C# наведено у Додатку Ж (див. процедуру buttonRun2_Click). Код додаткових класів наведено у Додатках И – К, М.

Блок 1 – початок процесу вилучення повідомлення з заповненого контейнера.

Блок 2 – виділення пам’яті для збереження значення преамбули.

Блок 3 – відкриття обраного файлу заповненого контейнера.

Блок 4 – створення масиву array для пікселів заповненого контейнера.

Блок 5 – запис в масив array значень пікселів заповненого контейнера.

Блок 6 – читання бітів преамбули з елементів array.

Блок 7 – створення масиву bytes для збереження вилученого повідомлення.

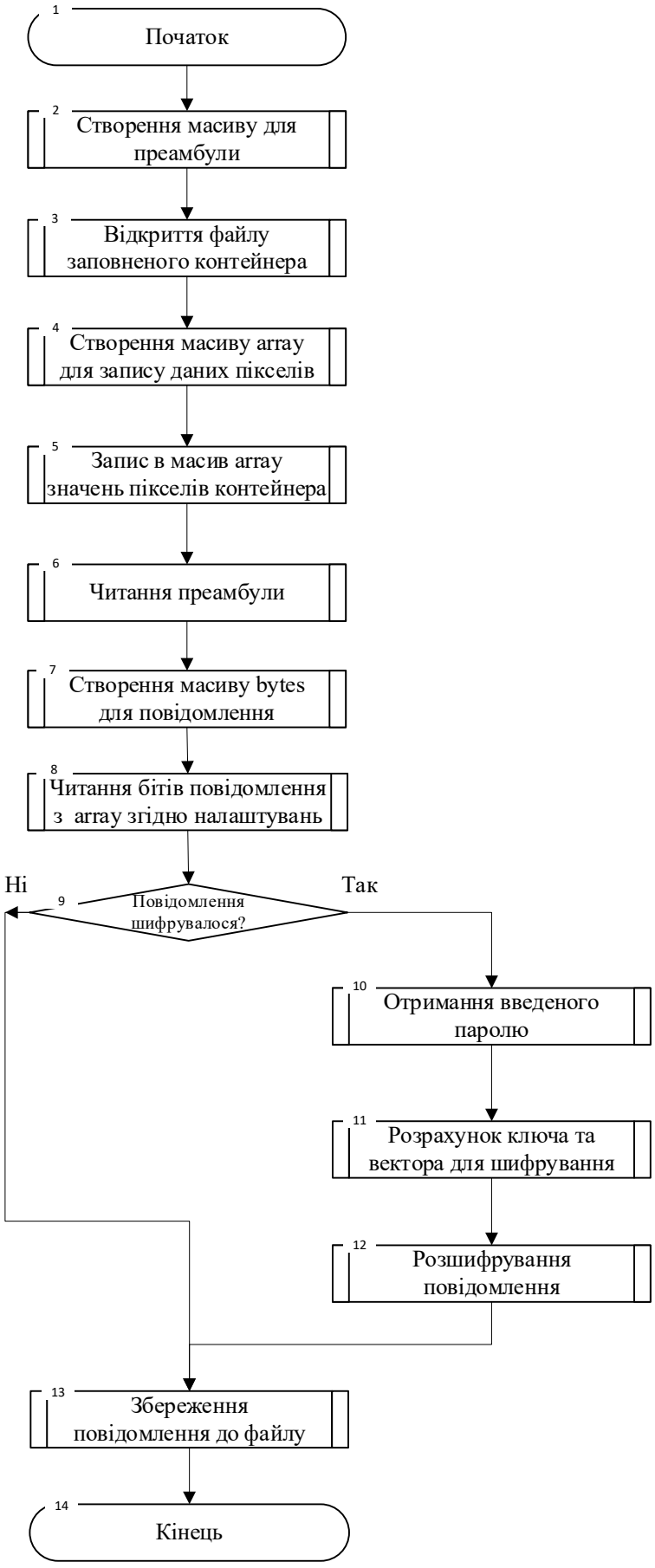

Рисунок 3.2 – Блок-схема узагальненого алгоритму вилучення повідомлення

Блок 8 – читання з елементів array бітів вбудованого повідомлення в bytes згідно налаштувань.

Блоки 9 – 12 – розшифровування повідомлення за допомогою пароля, якщо воно було зашифроване.

Блок 13 – збереження байтів масиву bytes в якості файлу повідомлення.

Блок 14 – закінчення процесу вилучення повідомлення.

3.4 Перевірка працездатності програми

Для перевірки працездатності обрано стегоконтейнери типу BMP-24 різних розмірів та повідомлення, що підлягають прихованню, різних розмірів та форматів. Програма не передбачає ніяких обмежень до типу повідомлення, тому перевіряється на деякому популярному наборі файлів. Короткий опис використаних файлів подано в таблиці 3.1.

Таблиця 3.1 – Опис файлів для перевірки програми

№	Назва файлу	Тип	Розмір
Контейнери			
1	001	Файл "BMP" (.bmp)	1 157 814 байт
2	002	Файл "BMP" (.bmp)	6 220 854 байт
3	003	Файл "BMP" (.bmp)	454 байт
Повідомлення			
1	1_text	Microsoft Word 97 - 2003 Document (.doc)	26 624 байт
2	2_foto	Файл "JPG" (.jpg)	28 579 байт
3	3_text	Текстовий документ (.txt)	291 байт
4	4_doc	Файл "PDF" (.pdf)	235 295 байт
5	5_program	Додаток (.exe)	82 944 байт
6	6_struct	Microsoft Visio Drawing (.vsdx)	82 816 байт
7	7_art	Файл "GIF" (.gif)	36 973 байт

Щоб перевірити працездатність програми, потрібно відтворити можливі ситуації, що можуть виникнути при роботі з нею.

Оскільки довжина обраного повідомлення записується до преамбули, яка має сталий розмір, то програма недозволить обрати файл розміру більшого, ніж вмістить в себе преамбула. Обробка ситуації вибору завеликого повідомлення

показана на рисунку 3.3. Як тільки вікно повідомлення про помилку буде закрите, шлях до обраного файлу буде очищенно і можна обрати новий.

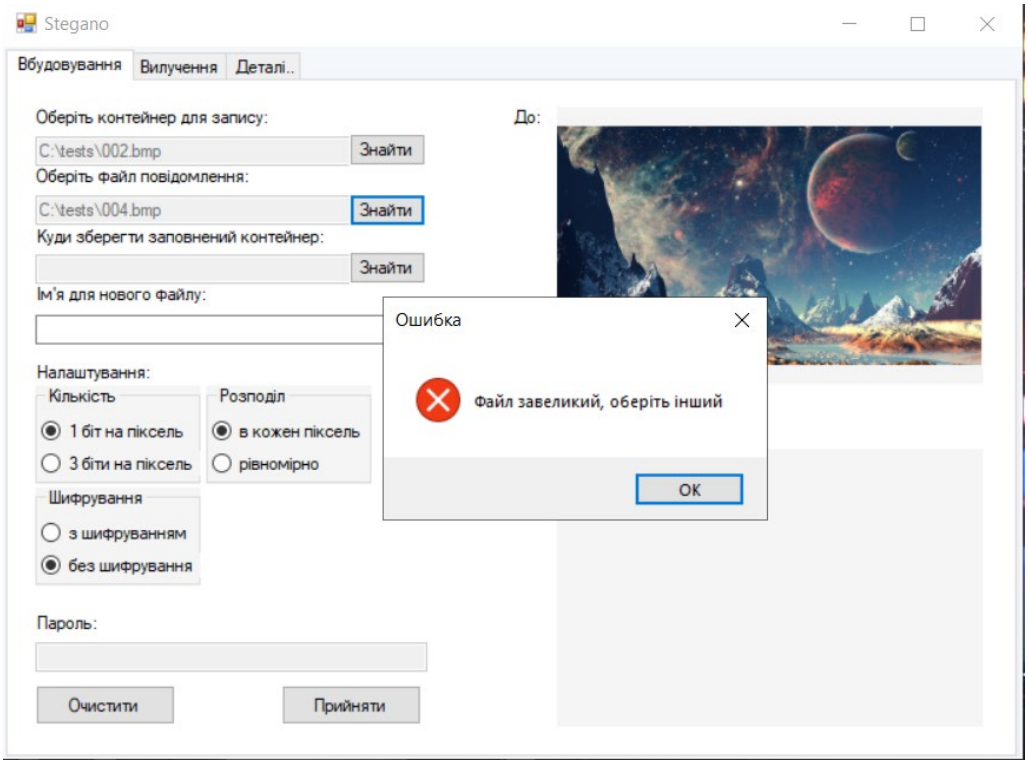

Рисунок 3.3 – Обробка ситуації вибору завеликого повідомлення

Програма перевіряє, чи поміститься до вхідного пустого контейнера обране повідомлення, то можлива ситуація, де повідомлення не зможе записатись повністю до одного контейнера. Реакція програми на таке виключення представлена на рисунку 3.4.

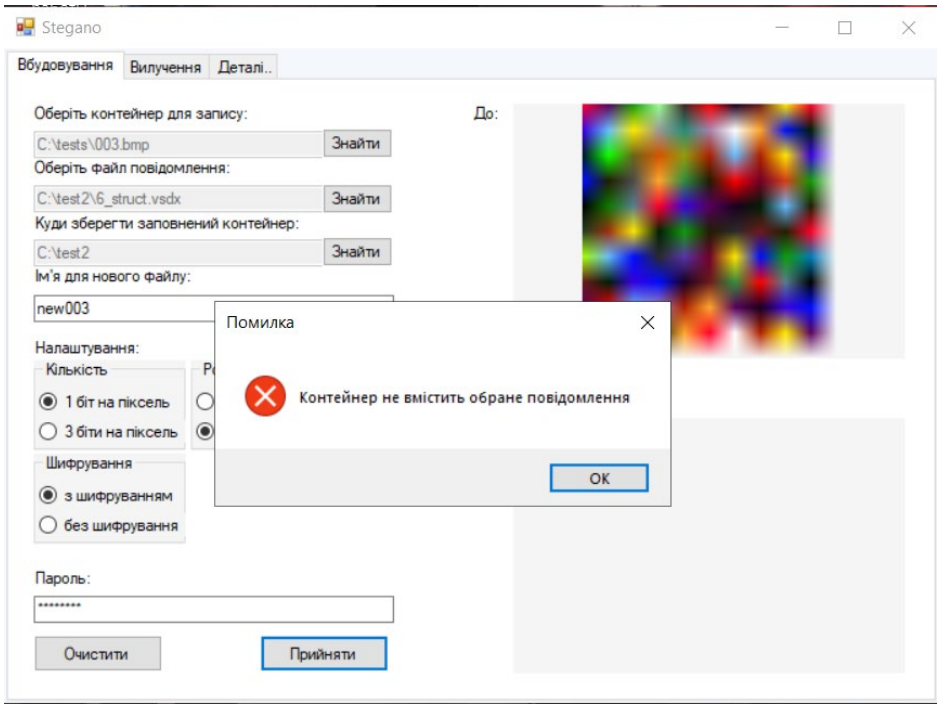

Рисунок 3.4 – Перевірка вміщення контейнером обраного повідомлення

Також можливий варіант введення недопустимих назв для імені нових файлів. Оскільки введення імені присутнє як на вкладці вбудовування, так і на вкладці вилучення, то обробку виключення показано на рисунках 3.5 та 3.6 відповідно.

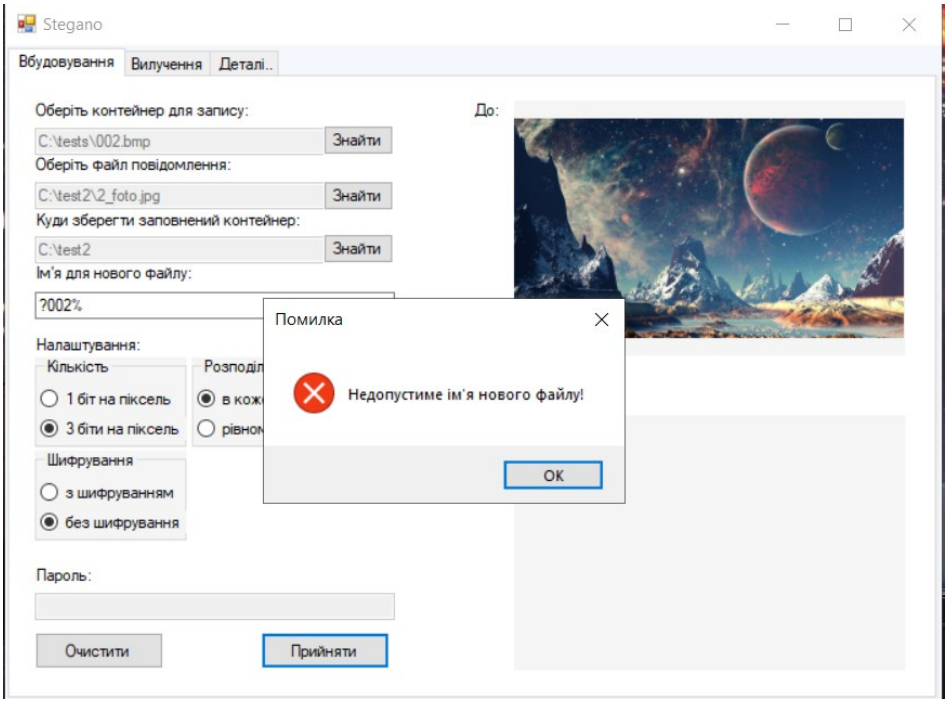

Рисунок 3.5 – Введення недопустимої назви нового файлу при вбудовуванні

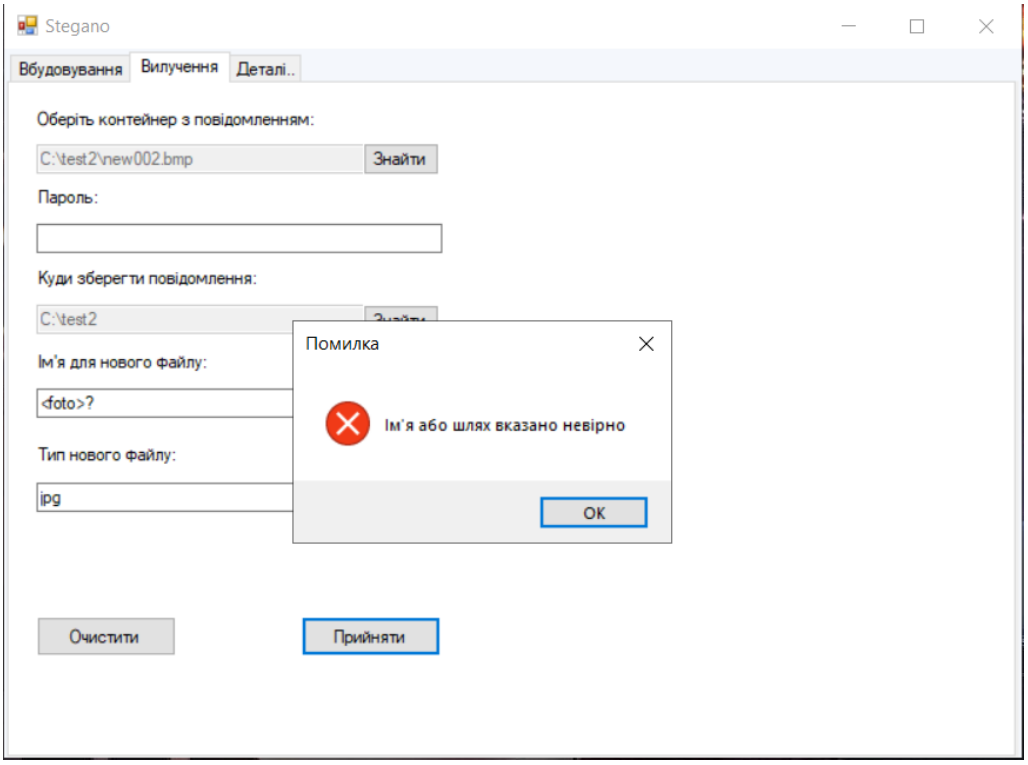

Рисунок 3.6 – Введення недопустимої назви нового файлу при вилученні

Якщо не обрати хоч якийсь з файлів або не ввести імена нових файлів, програма повідомить про це (рисунок 3.7).

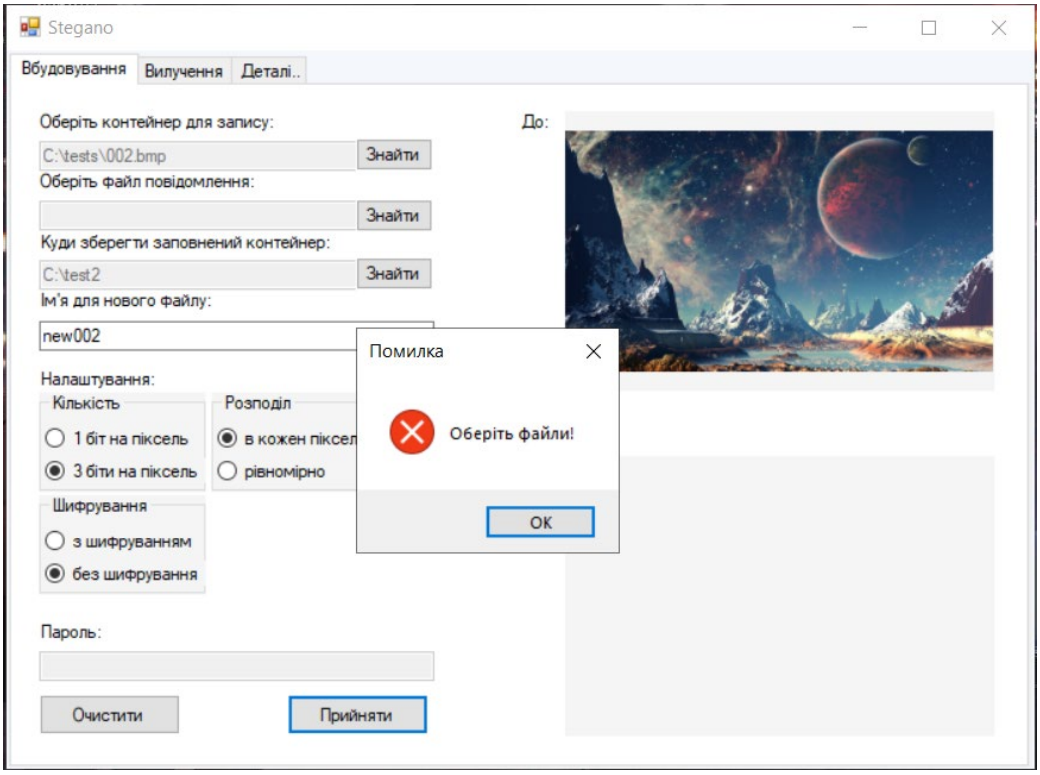

Рисунок 3.7 – Реакція програми на незаповнені обов’язкові поля

Якщо під час вилучення повідомлення не вказати шлях для нового файлу, то файл з вилученим повідомленням буде збережено до системного диску (рисунок 3.8).

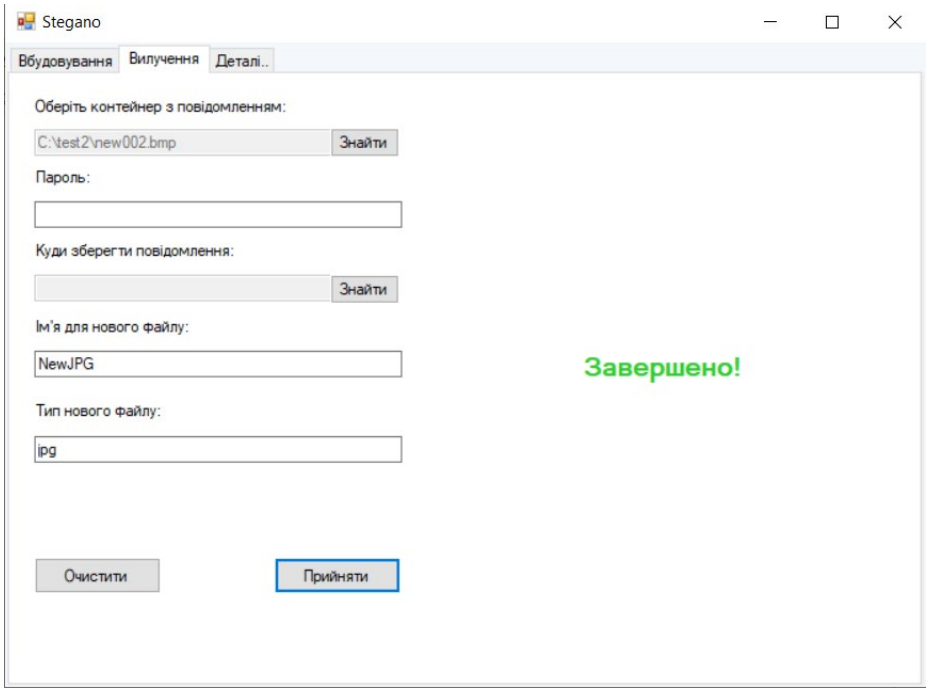

Рисунок 3.8 - Виконання вилучення без шляху для нового файлу

Якщо під час вилучення не задати тип створюваного файлу для повідомлення, то вилучення відбудеться у файл без типу (рисунки 3.9 – 3.10).

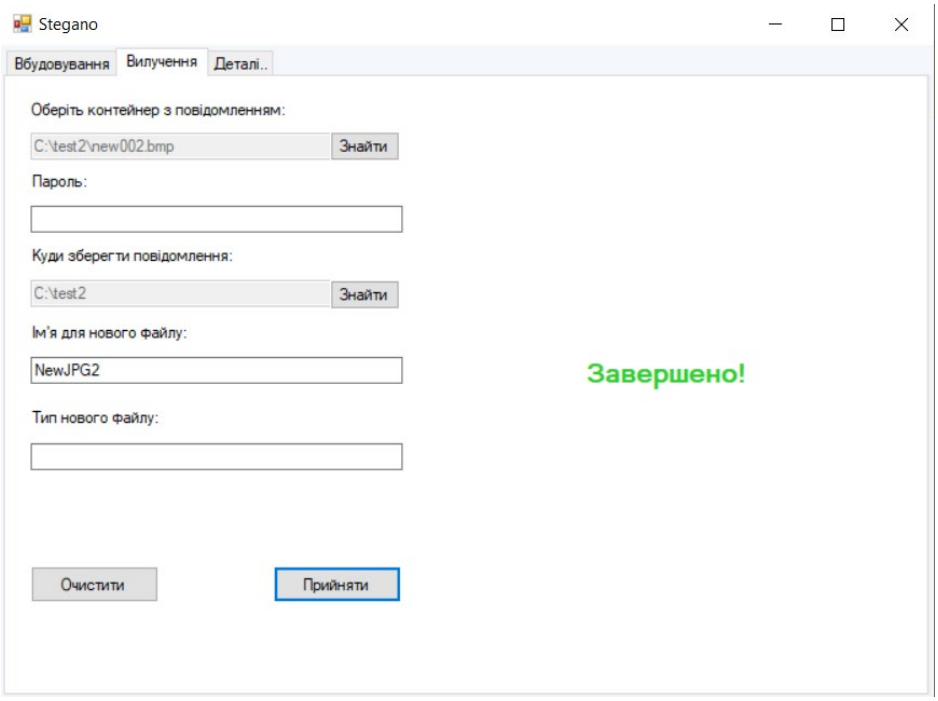

Рисунок 3.9 – Виконання вилучення без вказання типу для нового файлу

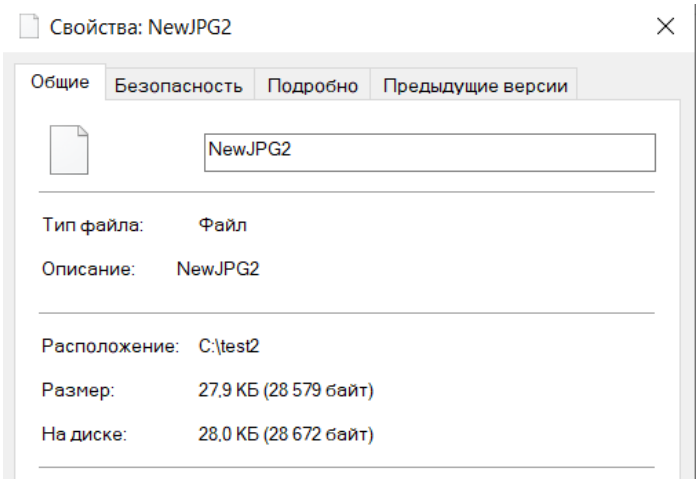

Рисунок 3.10 – Вікно властивостей створеного файлу з повідомленням

Переглянути режим «Демонстрації» на вкладці «Деталі» можливо тільки після вбудовування повідомлення у деякий контейнер, тому існує загроза появи виключення, обробка якого показана на рисунку 3.11.

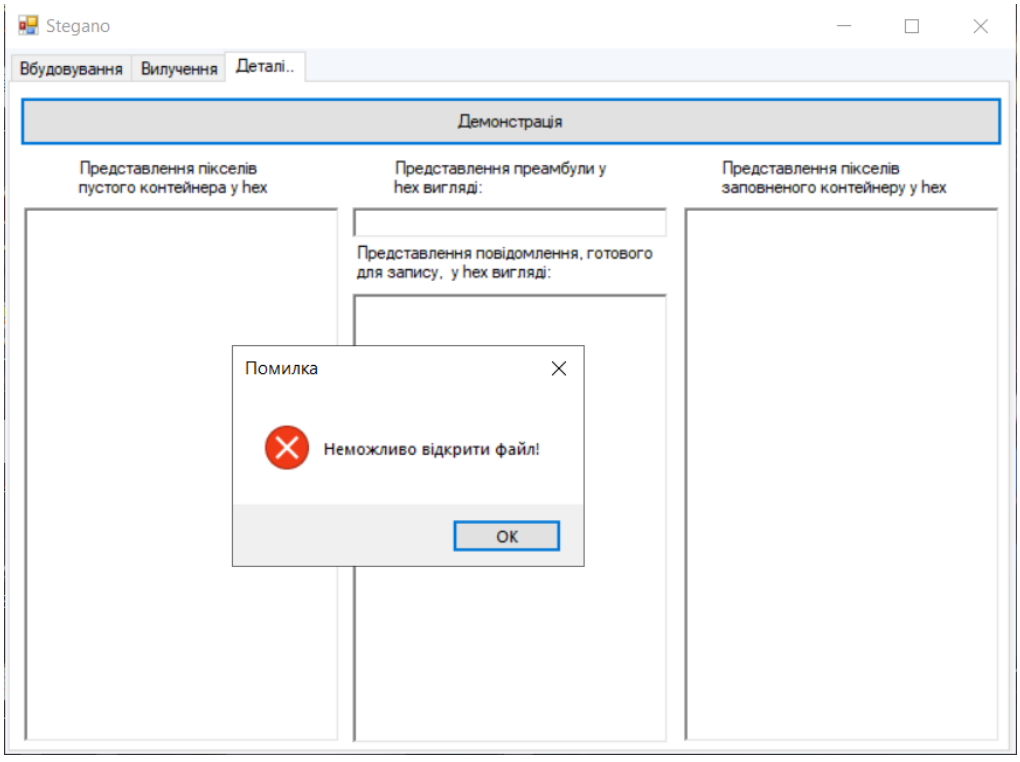

Рисунок 3.11 – Спроба перегляду «Демонстрації» без виконання процесу вбудовування

В таблиці 3.2 представлені реакції програми при вбудовуванні повідомлень у контейнер при різних налаштуваннях. Протестовані усі варіанти вибору налаштувань.

Таблиця 3.2 – Опис даних для тестування вбудовування різних повідомлень

№	**Вхідні дані**			
	Контейнер	Повідомлення	Налаштування	Ім’я заповненого контейнера
1	002	1_text	1, Підряд, З шифруванням	002_1_text_cont
2	002	4_doc	1, Рівномірно, Без шифрування	002_4_doc_cont
3	002	7_art	3, Підряд, Без шифрування	002_7_art_cont
4	002	5_program	3, Рівномірно, З шифруванням	002_5_program_cont

Значення налаштувань:

– 1 біт / 3 біти – по одному або по три біти повідомлення записується до пікселя;
– Підряд / Рівномірно – пікселі з повідомленням обираються підряд (в кожен піксель) або рівномірно розподіляються по забраженню;
– З шифруванням / Без шифрування – чи здійснюється попереднє шифрування файлу чи ні.

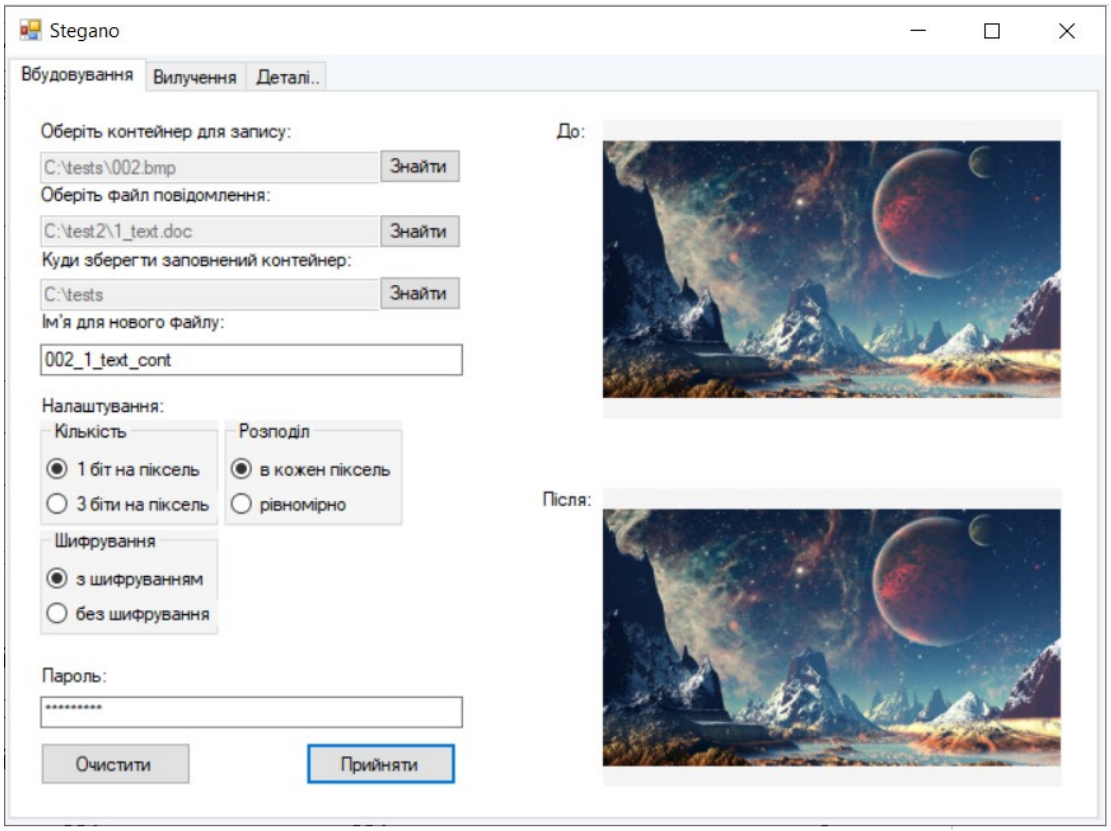

Рисунок 3.12 – Обробка програмою ситуації №1 вбудовування

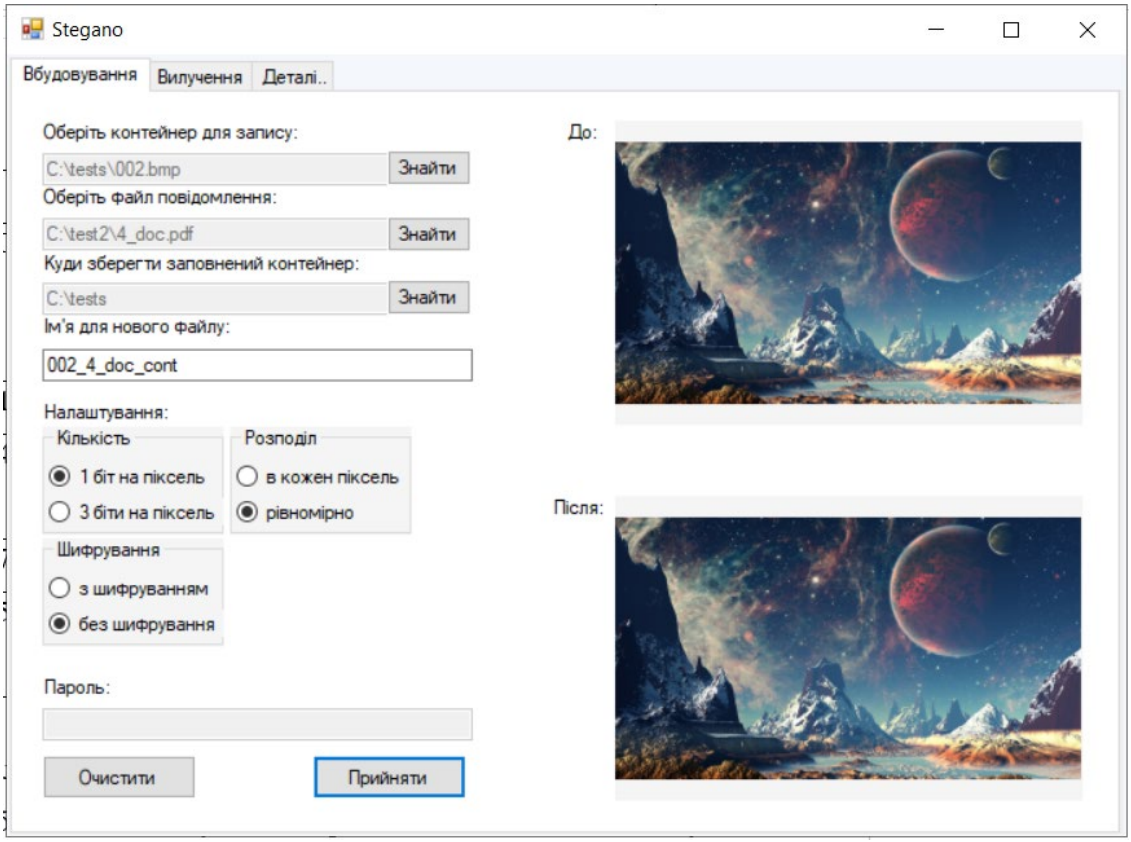

Рисунок 3.13 – Обробка програмою ситуації №2 вбудовування

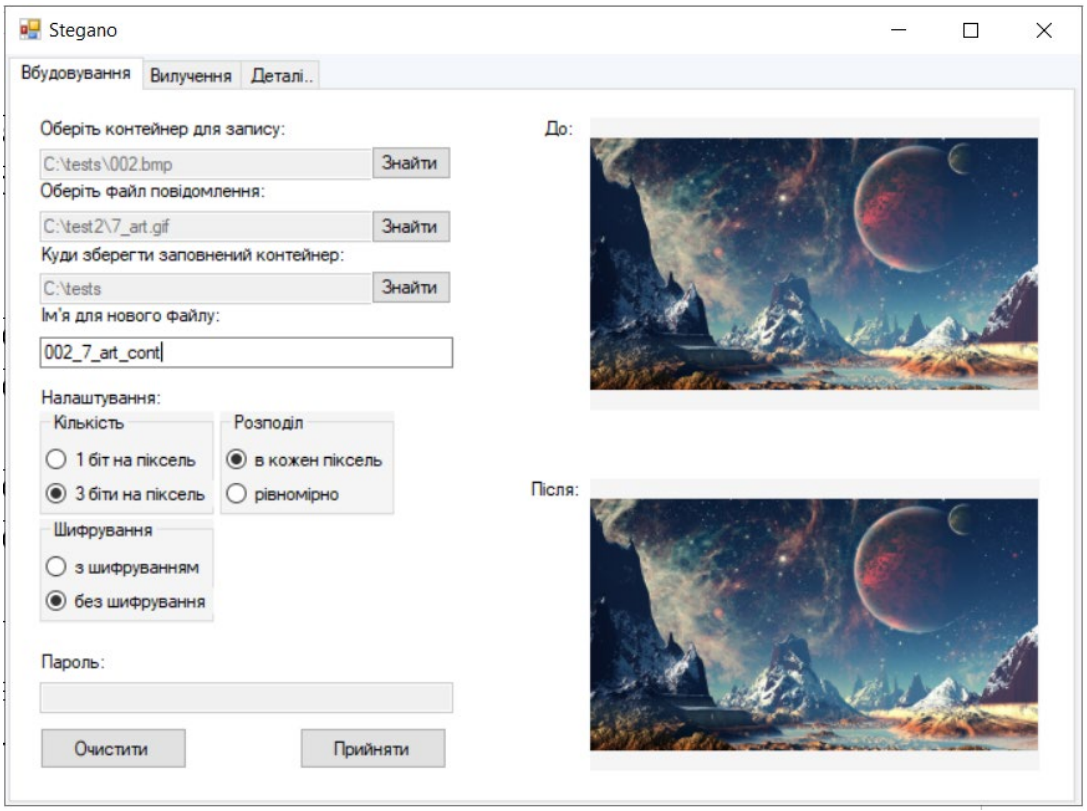

Рисунок 3.14 – Обробка програмою ситуації №3 вбудовування

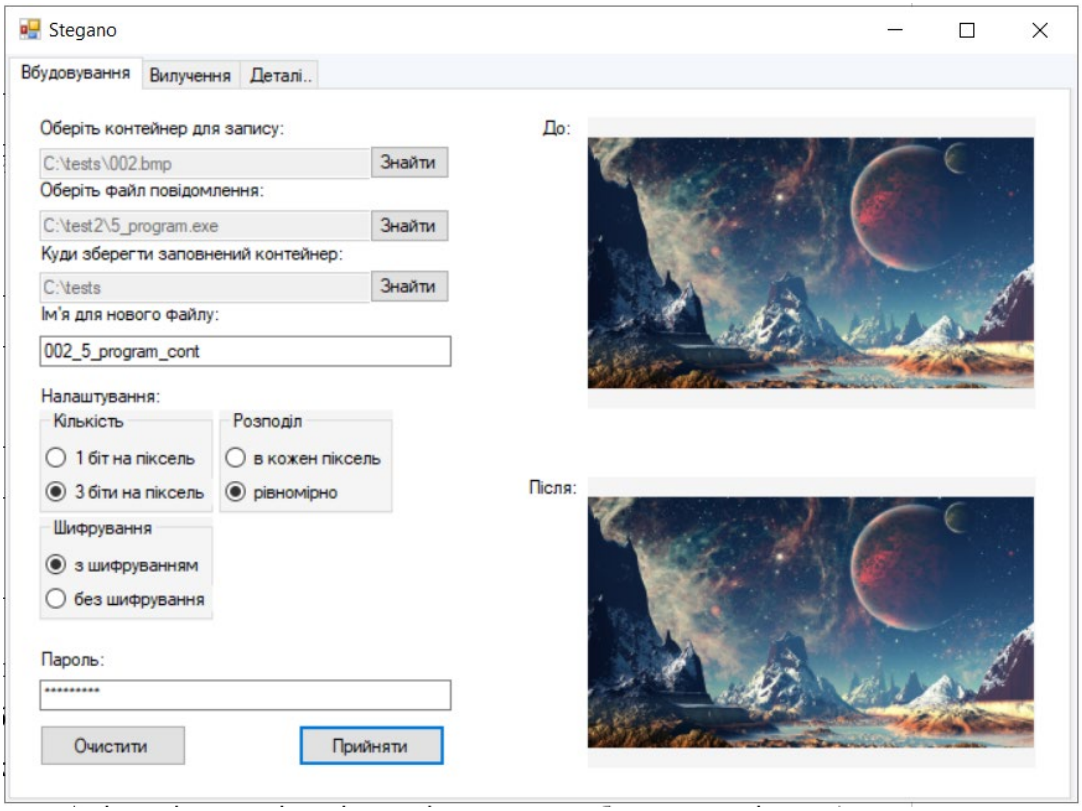

Рисунок 3.15 – Обробка програмою ситуації №4 вбудовування

В таблиці 3.3 зібрані результати вилучення попередньо вбудованих повідомлень (див. табл. 3.2). Процес тестування вилучення подано на рисунках 3.16 – 3.19.

Таблиця 3.3 – Опис даних для тестування вилучення різних повідомлень

№	Вхідні дані			
	Ім’я заповненого контейнера	Ім’я нового файлу з повідомленням	Розмір вилученого повідомлення	Відповідність оригінальному повідомленню
1	002_1_text_cont	1_text_2	26 624 байт	Повна
2	002_4_doc_cont	4_doc_2	235 295 байт	Повна
3	002_7_art_cont	7_art_2	36 973 байт	Повна
4	002_5_program_cont	5_program_2	82 944 байт	Повна

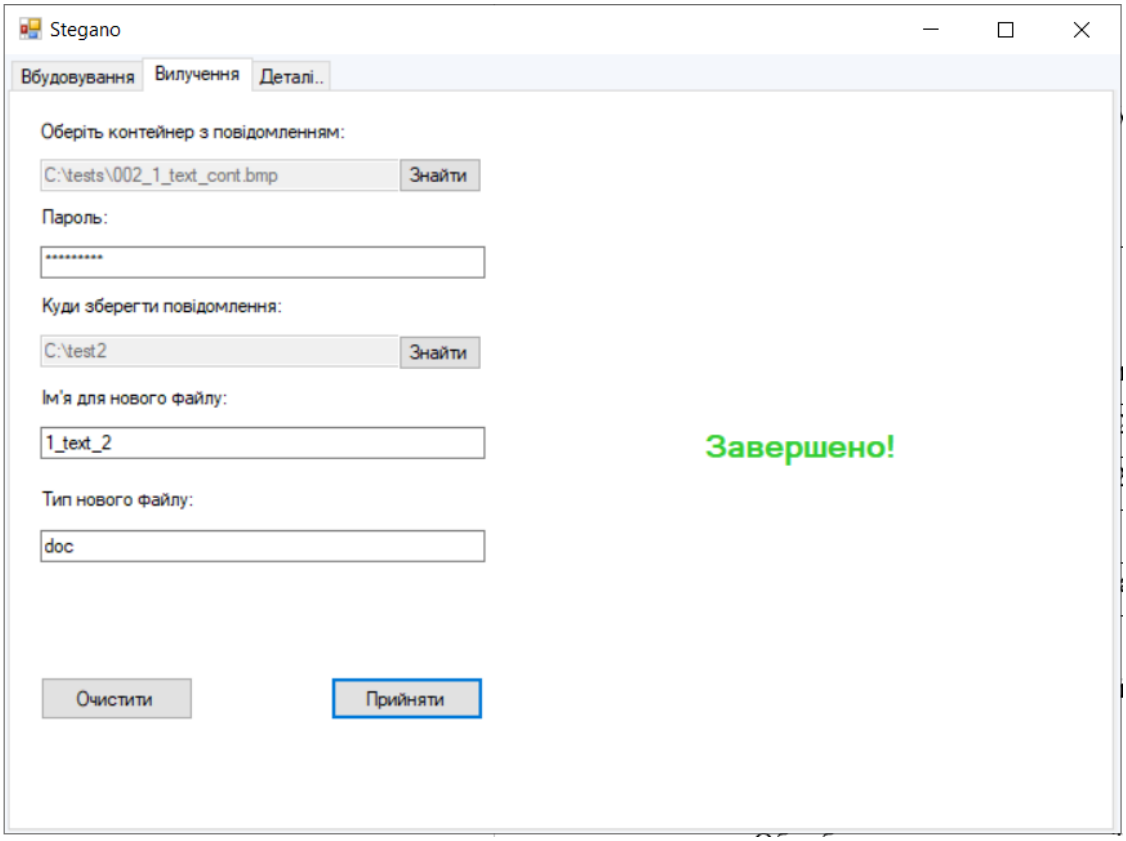

Рисунок 3.16 – Обробка програмою ситуації №1 вилучення

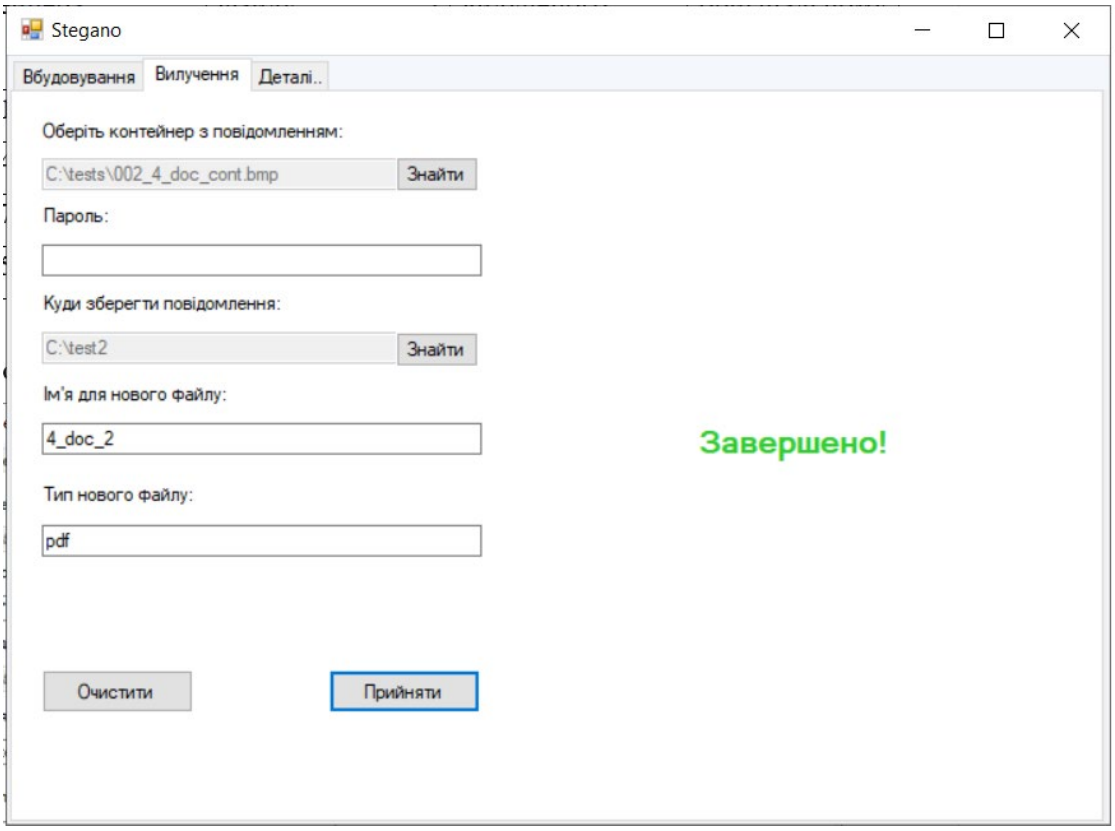

Рисунок 3.17 – Обробка програмою ситуації №2 вилучення

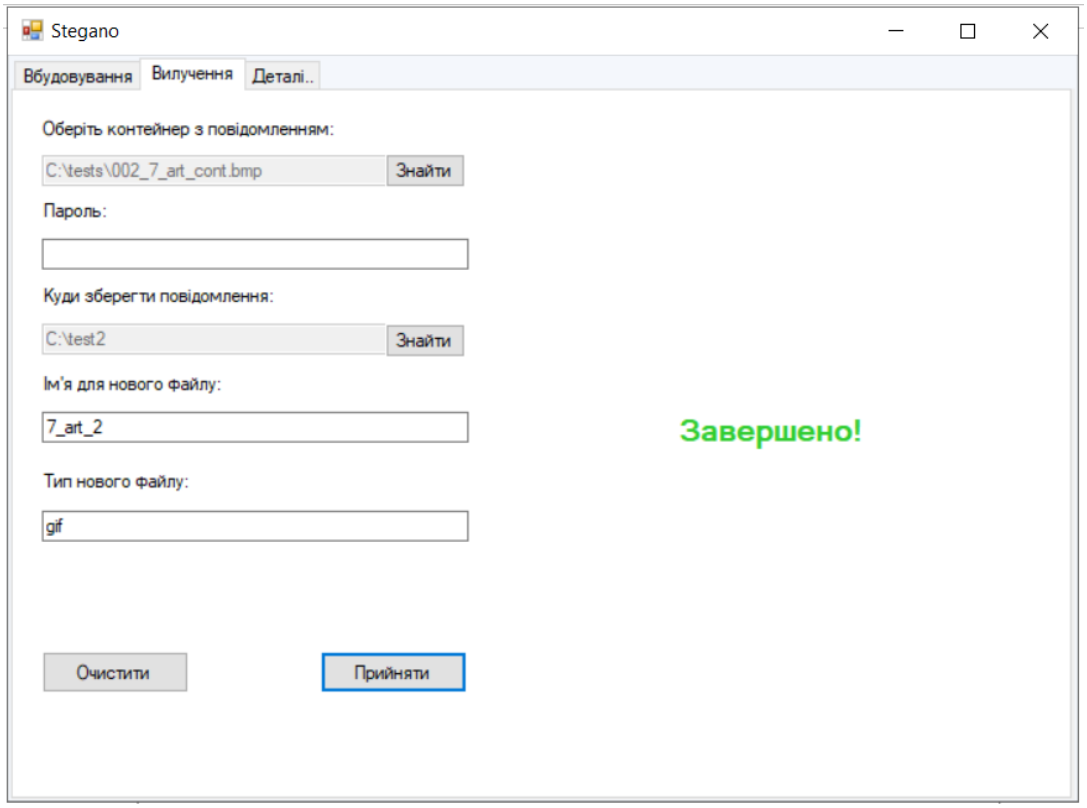

Рисунок 3.18 – Обробка програмою ситуації №3 вилучення

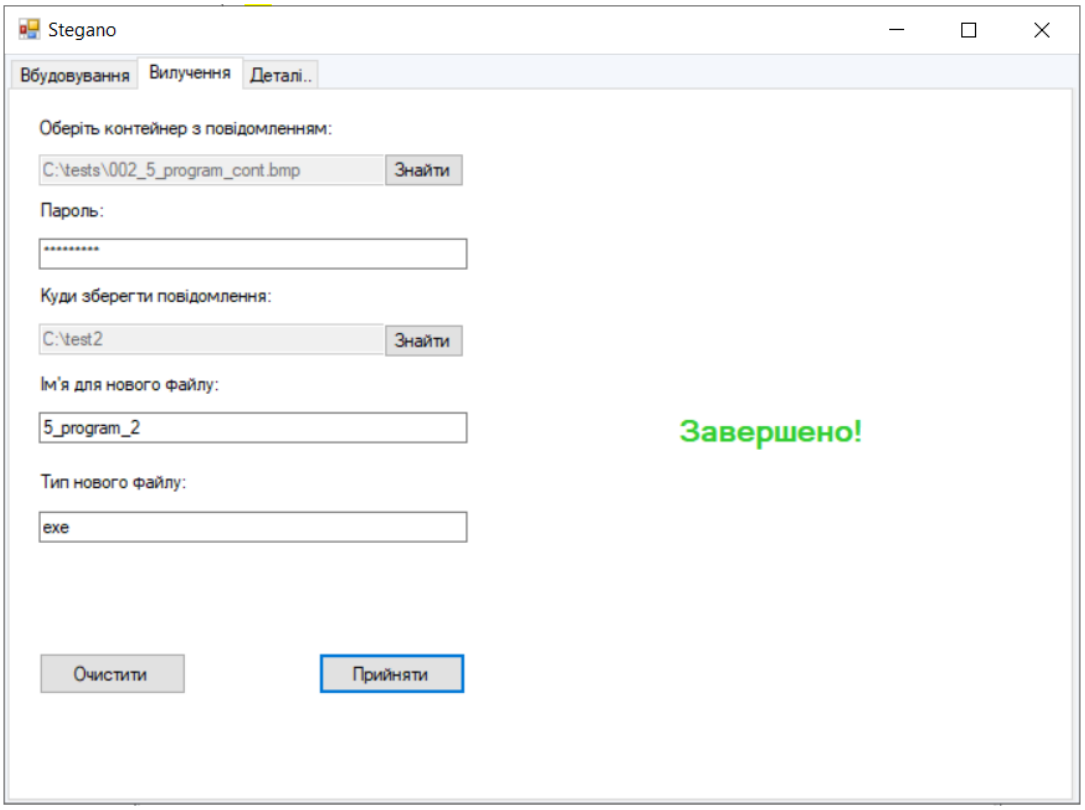

Рисунок 3.19 – Обробка програмою ситуації №4 вилучення

При вилученні повідомлень, що були попередньо зашифровані перед вбудуванням, без пароля (рисунок 3.20) або з невірним паролем (рисунок 3.21), програма виконає вилучення повідомлення, яке неможливо прочитати (рисунки 3.22 – 3.23).

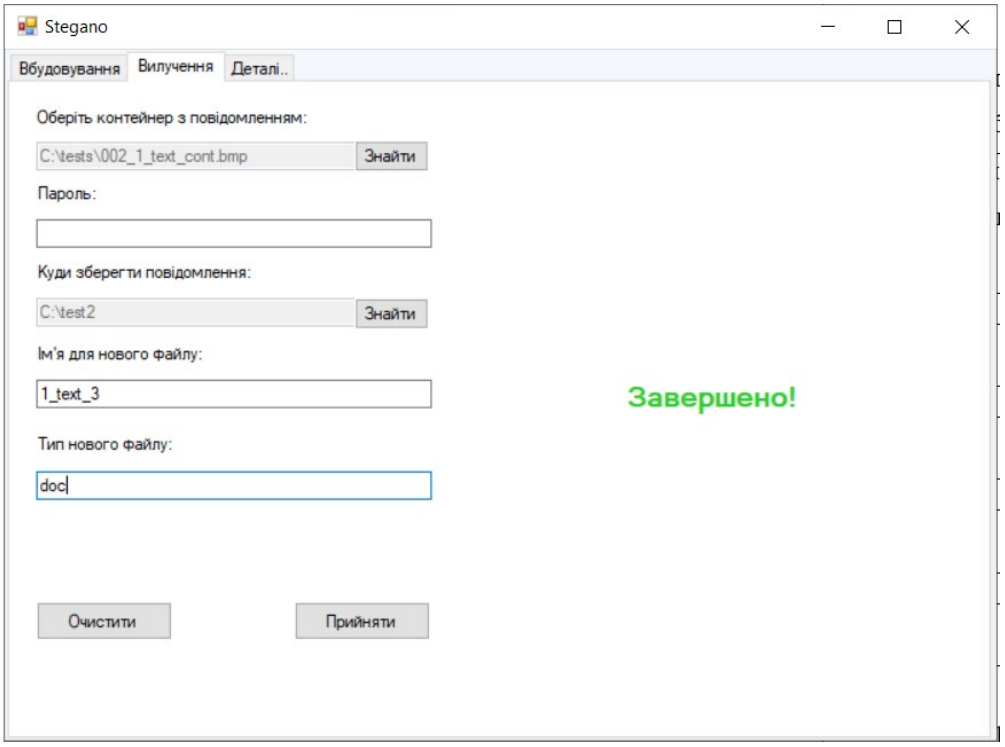

Рисунок 3.20 – Вилучення зашифрованого повідомлення без пароля

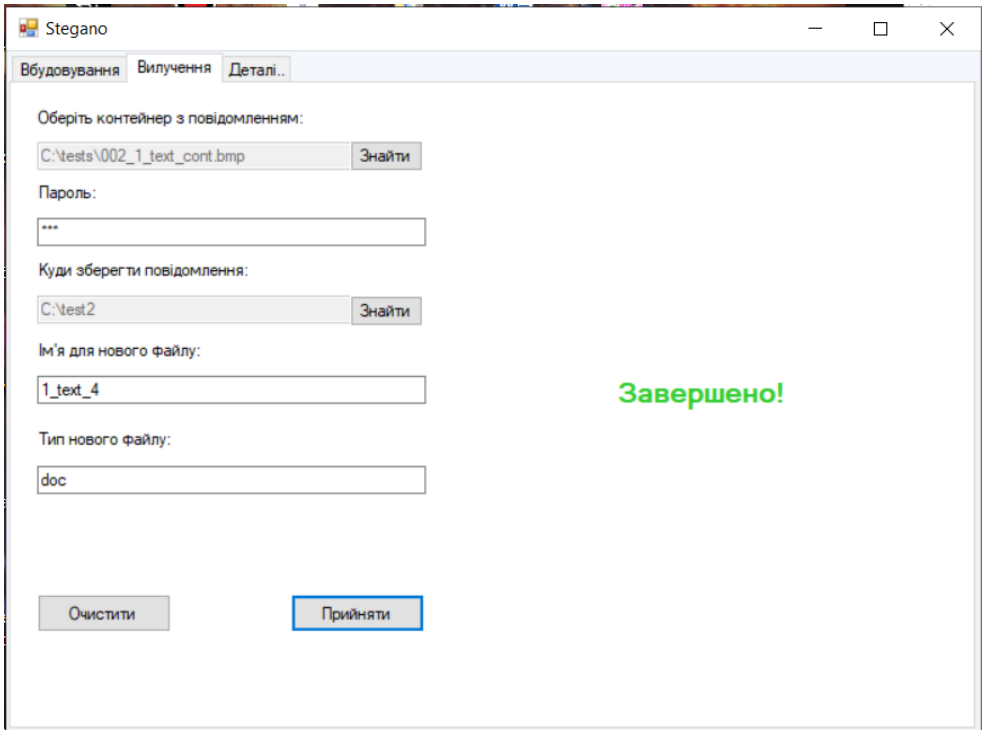

Рисунок 3.21 – Вилучення зашифрованого повідомлення з невірним паролем

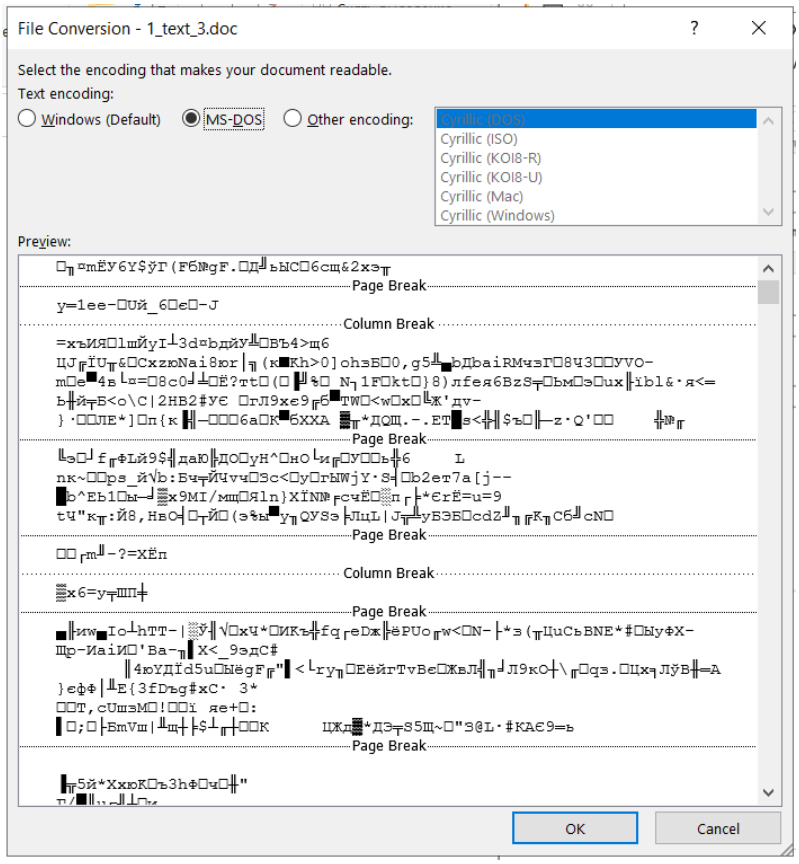

Рисунок 3.22 – Вигляд отриманого повідомлення 1_text_3.doc

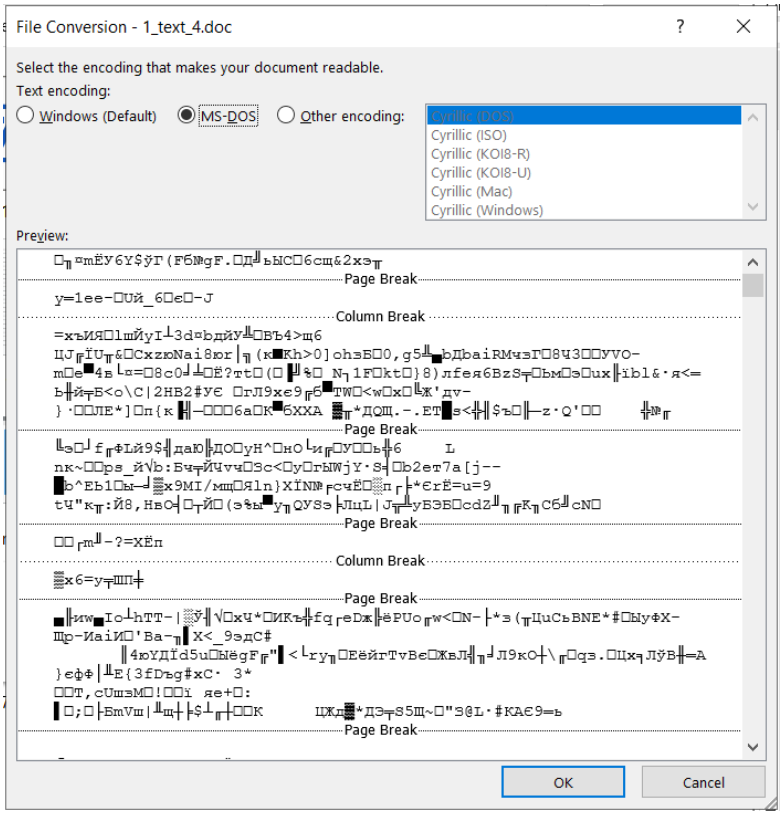

Рисунок 3.23 – Вигляд отриманого повідомлення 1_text_4.doc

При вилученні повідомлення, що не було попередньо зашифровано, програма ігнорує заповненість поля «Пароль» (рисунок 3.24).

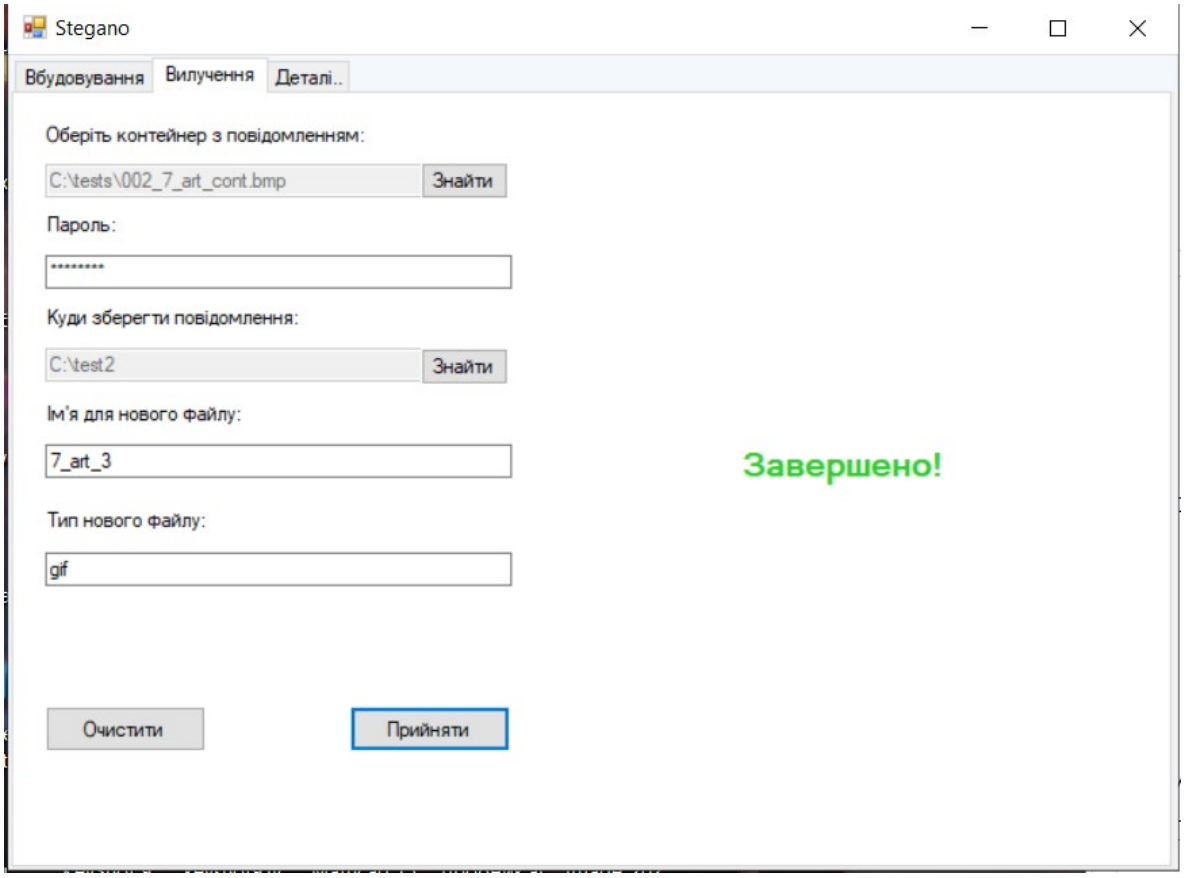

Рисунок 3.24 – Спроба ввести пароль при вилученні не зашифрованого повідомлення

3.5 Висновки за розділом

Обрано середовище розробки, мова програмування та потрібні бібліотеки. Подано узагальнені алгоритми, що виконують вбудування та вилучення повідомлення. Розроблена відповідна програма.. протестована написана програма за допомогою різних вхідних даних.

4 ІНСТРУКЦІЯ З ВИКОРИСТАННЯ КОМПЛЕКСУ

4.1 Вбудовування повідомлення

Щоб виконати вбудовування повідомлення потрібно відкрити програму та перейти на вкладку «Вбудовування». Вигляд вкладки подано на рисунку 4.1.

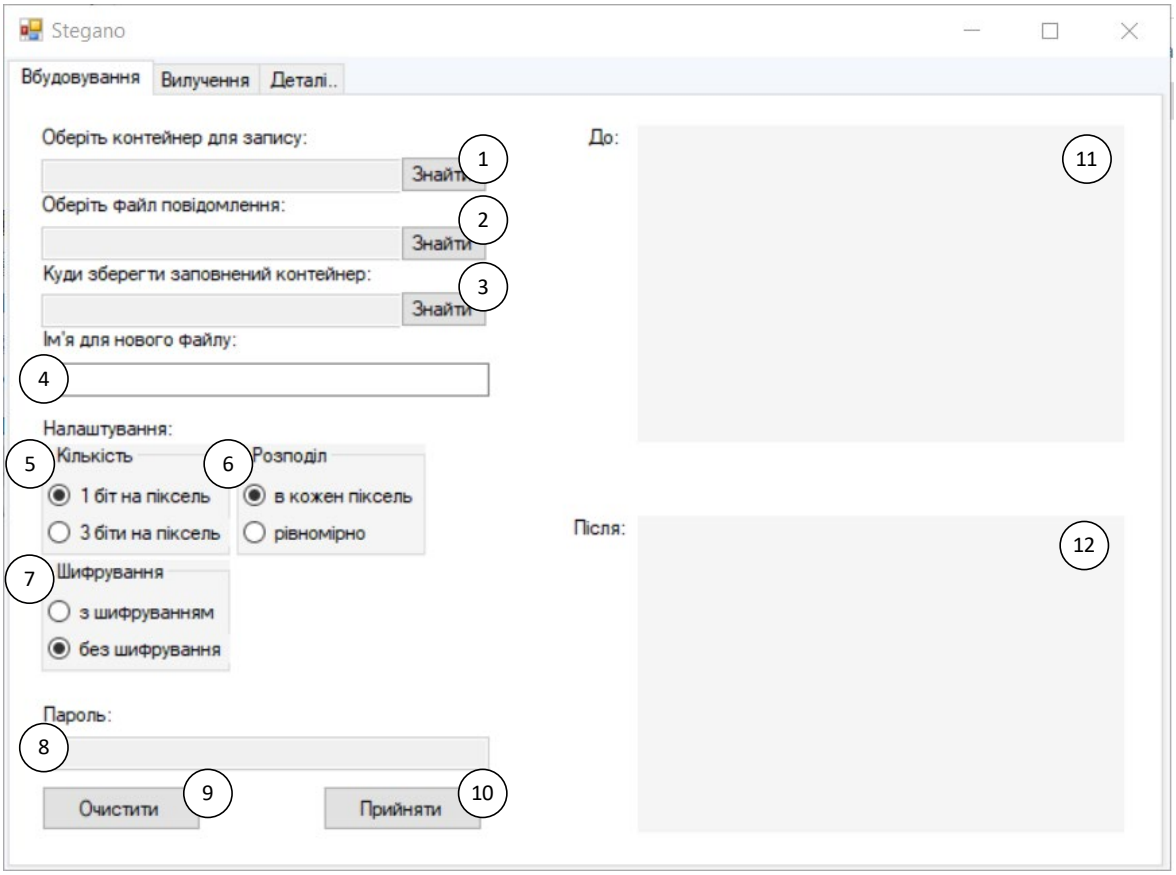

Рисунок 4.1 – Вкладка «Вбудовування»

Порядок роботи з вкладкою:

1) обрати файл пустого контейнера типу BMP-24 кнопкою (1) (після вибору зображення, воно відобразиться у полі (11)), обрати файл повідомлення кнопкою (2) та папку для збереження заповненого контейнера кнопкою (3) (введеня цих даних вручну не дозволяється);
2) введіть ім'я для заповнего контейнера у поле (4);
3) оберіть налаштування для вбудовування (5)-(7);
4) якщо обрано налаштування «з шифруванням» по введіть в поле (8) пароль для шифрування;

5) якщо потрібно очистити заповнені поля – натисніть кнопку (9);
6) щоб почати вбудовування написніть кнопку (10);
7) у полі (12) буде відображено готовий контейнер.

4.2 Вилучення повідомлення

Щоб виконати вилучення повідомлення потрібно відкрити програму та перейти на вкладку «Вилучення». Вигляд вкладки подано на рисунку 4.2.

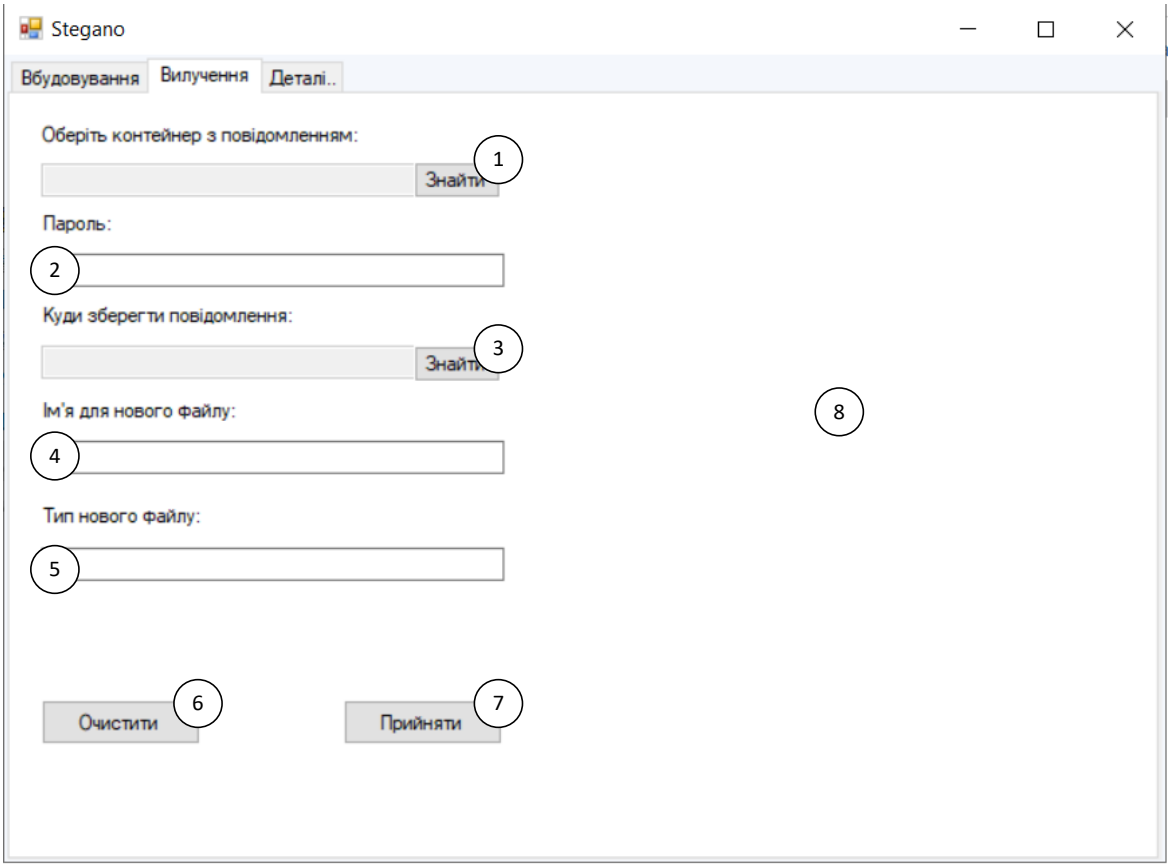

Рисунок 4.2 – Вкладка «Вилучення»

Порядок роботи з вкладкою:

1) обрати файл заповненого контейнера кнопкою (1);
2) якщо повідомлення було зашифроване, то введіть пароль в поле (2); якщо повідомлення не було зашифроване, а поле пароля запонене, то програма це поле проігнорує;
3) обрати кнопкою (3) папку для збереження вилученого повідомлення;
4) задайте ім'я для файлу повідомлення у полі (4);

5) у поле (5) введіть розширення файлу повідомлення без крапки (наприклад, «txt» або «pdf»);
6) якщо потрібно очистити заповнені поля – натисніть кнопку (6);
7) щоб почати вилученя натисніть кнопку (7);
8) коли вилучення буде завершено на місці (8) з’явиться повідомлення про завершення вилучення.

4.3 Перегляд деталей

Вигляд вкладки «Деталі..» подано на рисунку 4.3.

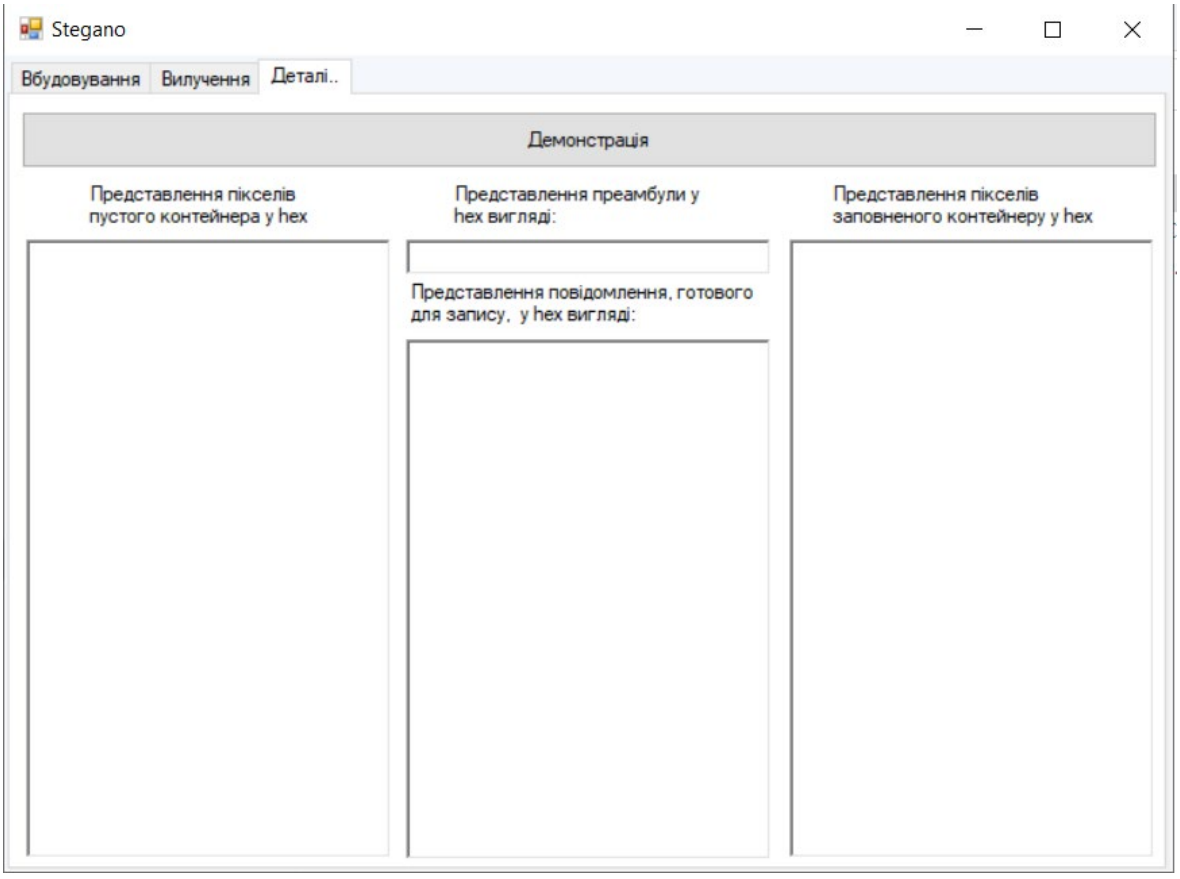

Рисунок 4.3 – Вкладка «Деталі..»

Щоб переглянути результат вбудування повідомлення в пікселі, потрібно виконати вбудування (див. пункт 4.1), перейти на вкладку «Деталі..» та клікнути на кнопку «Демонстрація». Приклад вбудування подано на рисунку 4.4, перегляд деталей для цього прикладу показано на рисунку 4.5.

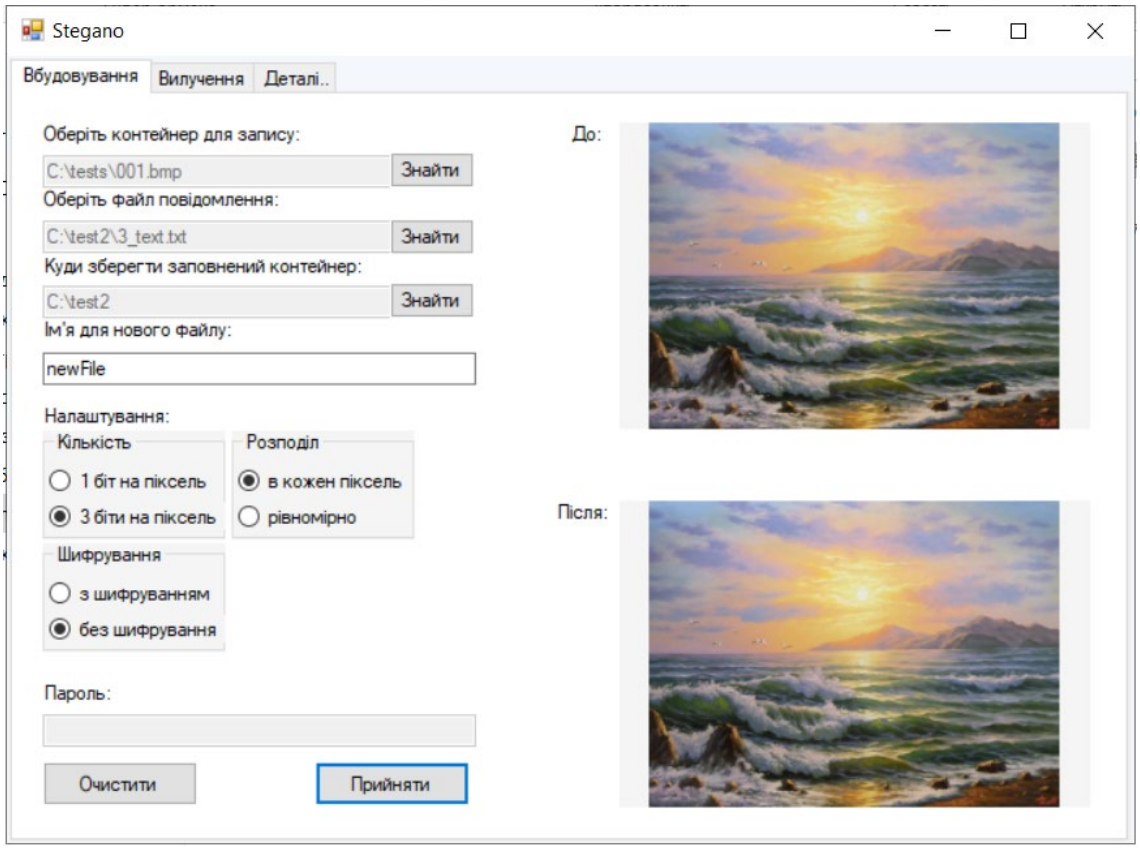

Рисунок 4.4 – Приклад вбудування повідомлення для перегляду деталей

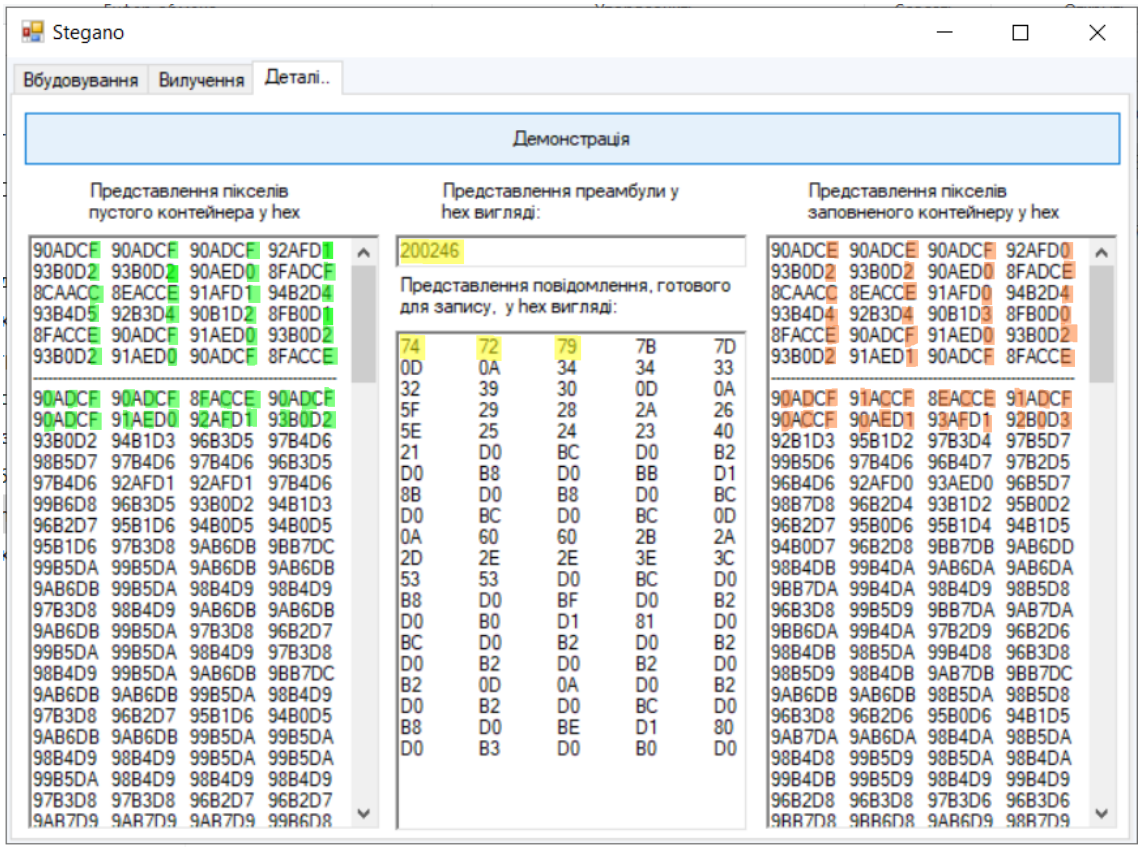

Рисунок 4.5 – Перегляд деталей вбудування

На рис. 4.5 зеленим показані значення, що будуть нести в собі біти повідомлення; жовтим значення в шістнадцятиричному вигляді байтів преамбули та повідомлення, що будуть вбудовані; червоним показані значення, що вже містять в собі біти повідомлення та преамбули. В представленнях пікселів пустого та заповненого контейнерів, пунктирною лінією розділені пікселі які виділені для запису преамбули та пікселі для запису повідомлення.

4.4 Висновки за розділом

Надана інструкція для роботи з різними режимами програми. Також представлено приклад роботи з вкладкою для перегляду деталей. Перегляд в шістнадцятеричному вигляді опрацьованих алгоритмом вбудовування даних можна використовувати для демонстрації стеганографічного приховування даних.

ВИСНОВКИ

У кваліфікаційній роботі розроблено програмний комплекс, який реалізує та демонструє стеганографічне приховування та вилучення даних, з використанням графічних контейнерів.

Наведено загальні поняття, що охоплює галузь стеганографії. Проведено порівняльний аналіз методів стеганографії для графічних контейнерів. За результатами цього аналізу, для реалізації у програмному комплексі обрано метод заміни найменш значущого біта.

Описані режими функціонування створеного комплексу: вбудовування повідомлення, вилучення повідомлення та перегляд деталей вбудовування.

Реалізована можливість додаткового захисту повідомлення шляхом шифрування за алгоритмом AES128. Наведені структури контейнера та преамбули.

Обрано середовище та мова розробки. Розроблено узагальнені алгоритми, що виконують вбудування та вилучення повідомлення. Розроблено програмне забезпечення згідно з функціоналом, що обрано для реалізації. Виконана перевірка працездатності комплексу на різних вхідних даних. Створена інструкція для роботи в різних режимах програми.

Розроблений комплекс може використовуватись під час навчального процесу студентів відповідних спеціальностей при проведенні лабораторних і практичних робіт.

ПЕРЕЛІК ПОСИЛАНЬ

1. О.О. Кузнецов. Стеганографія: навчальний посібник / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Х. : Вид. ХНЕУ, 2011. – 232 с.
2. Shulmin A., Krylova E. Steganography in contemporary cyberattacks [Електронний ресурс]: Electronic publication – August 3, 2017. – Режим доступу до ресурсу: https://securelist.com/steganography-in-contemporary-cyberattacks/79276/
3. Г.Ф. Конахович, Д.О. Прогонов, О.Ю. Пузиренко. Комп’ютерна стеганографічна обробка й аналіз мультимедійних даних. Підручник. Київ: «Центр учбової літератури», 2018. 558с.
4. С.А. Сейеди, Р.Х. Садыхов. Сравнение методов стеганографии в изображениях. Информатика №1. Белорусский государственный университет информатики и радиоэлектроники, Минск, 2013, С.66-75.
5. Advanced encryption standard (AES) [Електронний ресурс]: Federal Information Processing Standards Publication 197. – November 26, 2001. – Режим доступу до ресурсу: https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.197.pdf
6. System.Drawing Namespace [Електронний ресурс]: Microsoft Documentation. – 2022. – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/api/system.drawing?view=net-6.0
7. System.Security.Cryptography Namespace [Електронний ресурс]: Microsoft Documentation. – 2022. – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/api/system.security.cryptography?view=net-6.0
8. Visual Studio IDE documentation [Електронний ресурс]: Microsoft Documentation. – 2022. – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/visualstudio/ide/?view=vs-2022