



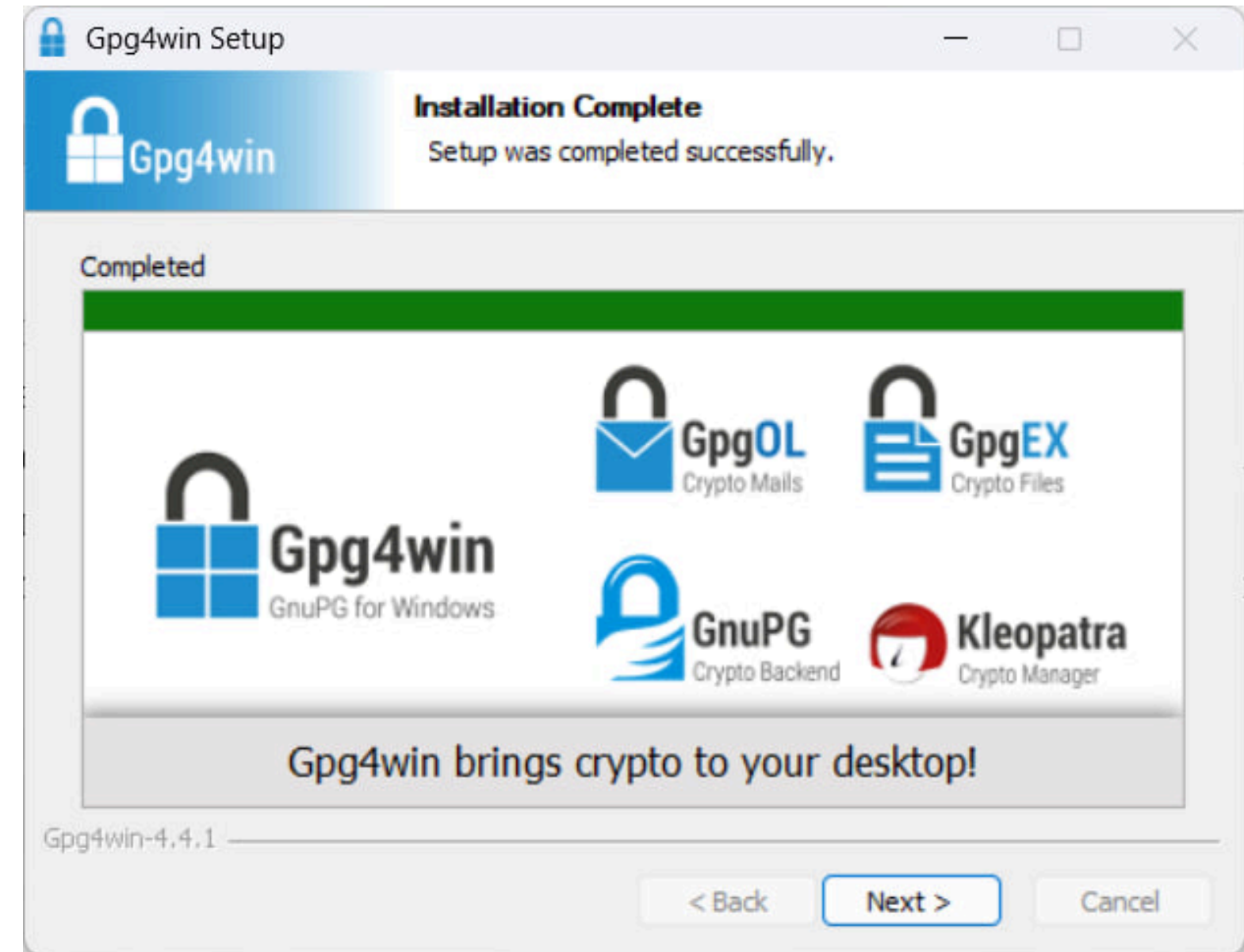
# ОСОБИСТИЙ ЦИФРОВИЙ ПІДПИС

Виконав Полулях Андрій



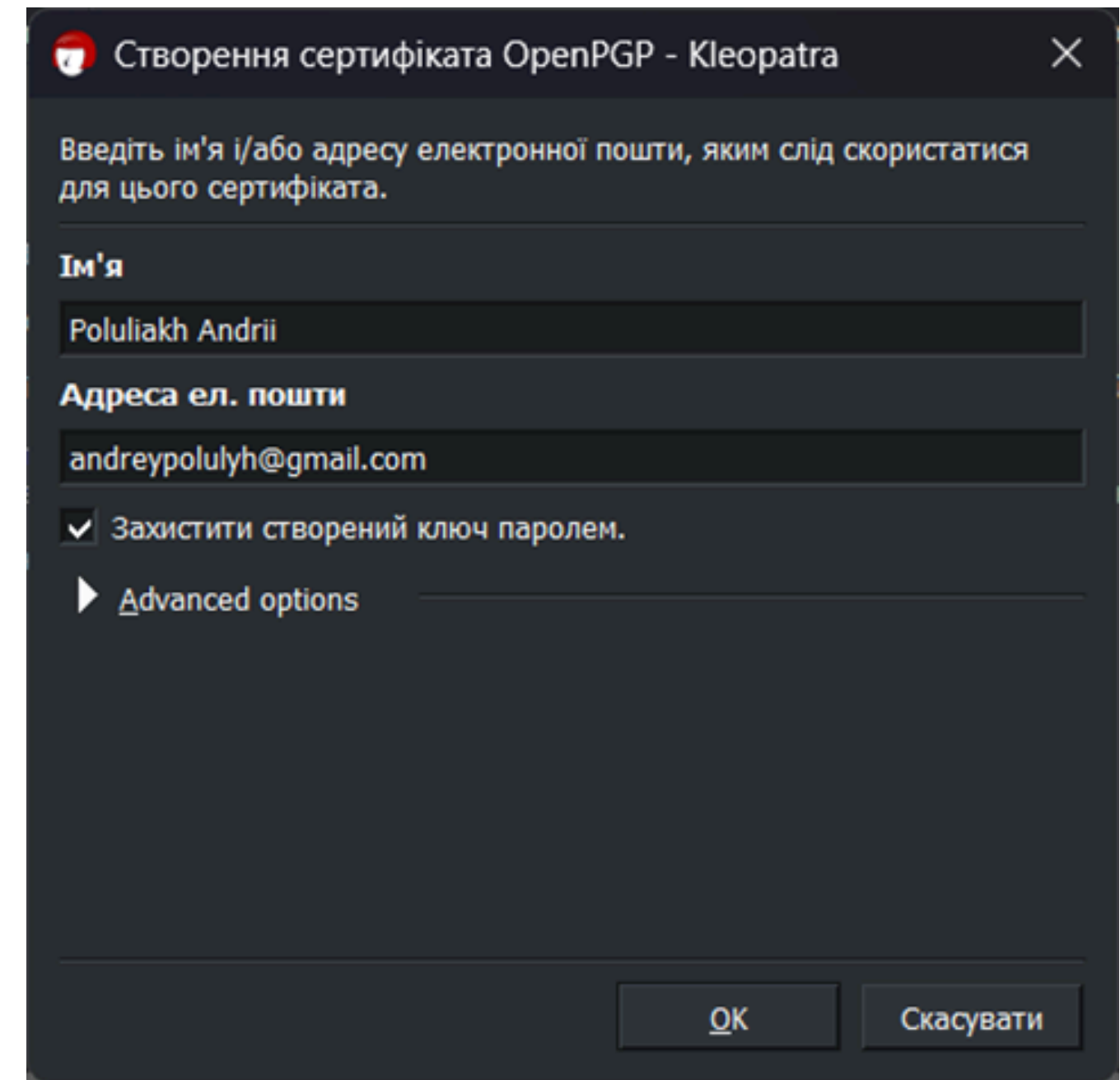
# КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

Завантажуємо та встановлюємо GPG4Win



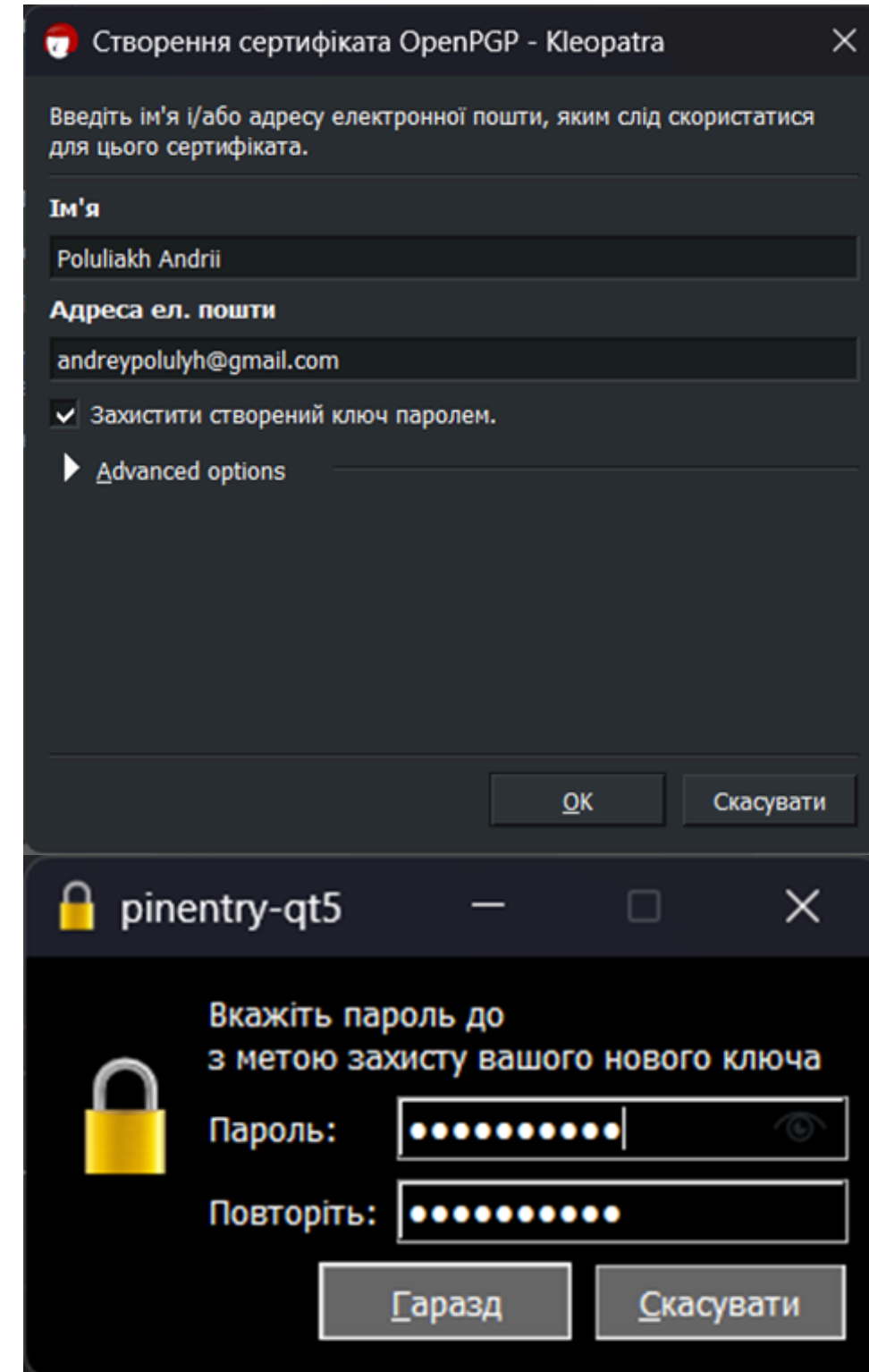
# КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

Для генерації тестової пари ключів запускаємо програму Kleopatra.



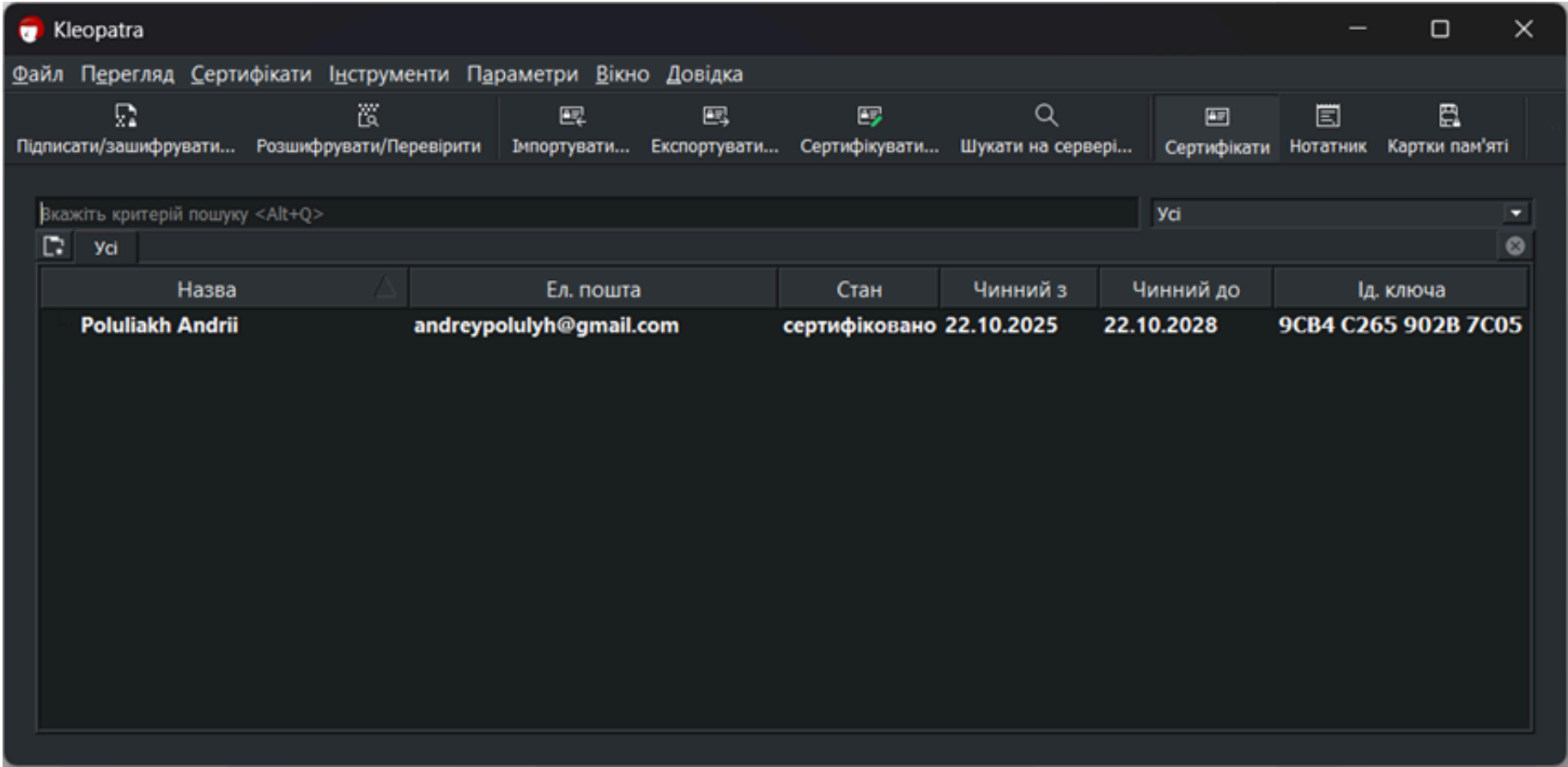
# КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

Для генерації тестової пари ключів запускаємо програму Kleopatra.



# КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

На головній сторінці відобразився створений ключ



## КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

Для створення публічного ключа обираємо щойно  
створений приватний ключ та на панелі інструментів  
обираємо функцію «Експортувати позначений сертифікат»



Poluliakh Andrii\_0x902B7C05\_public

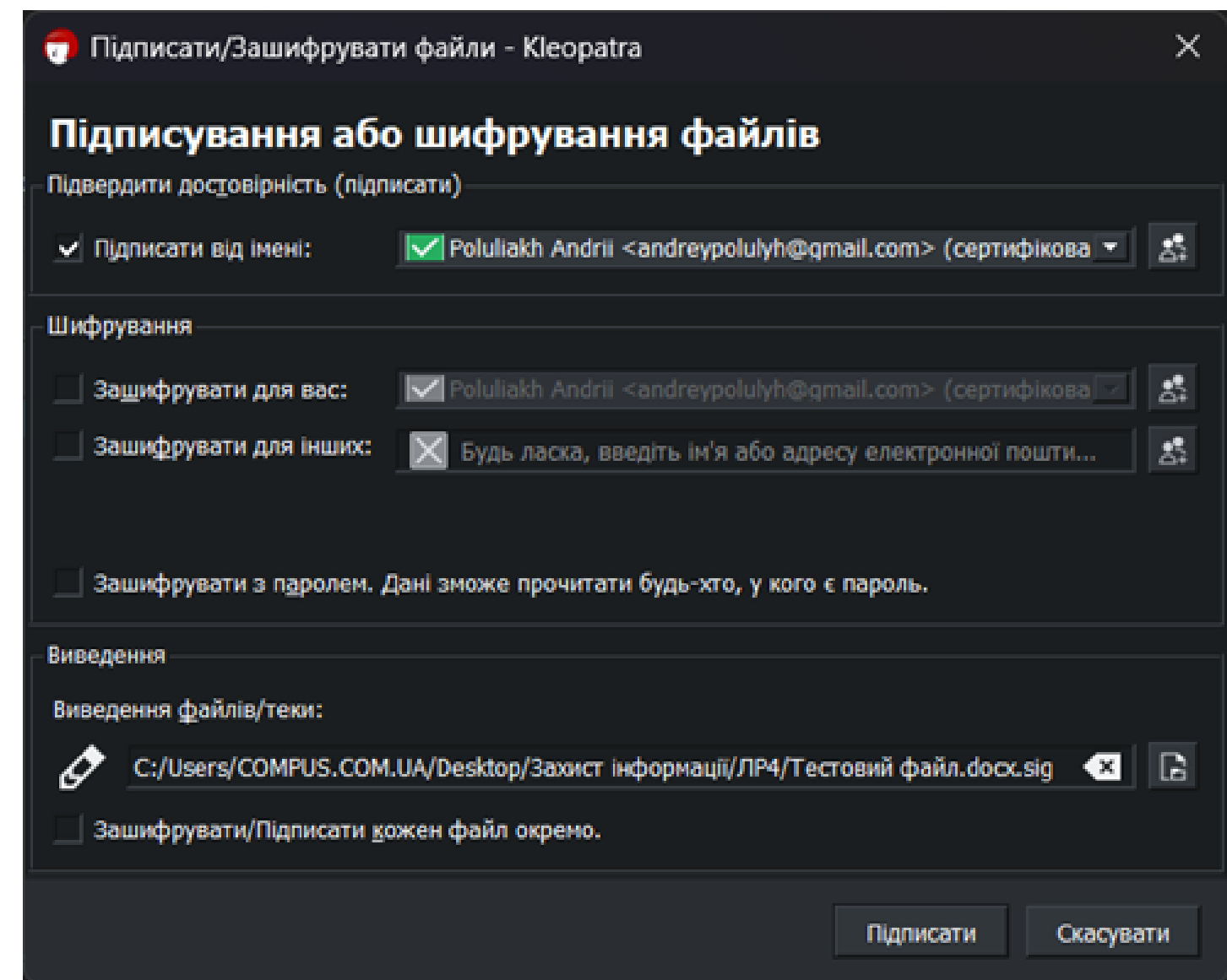
22.10.2025 22:33

OpenPGP Text File

1 КБ

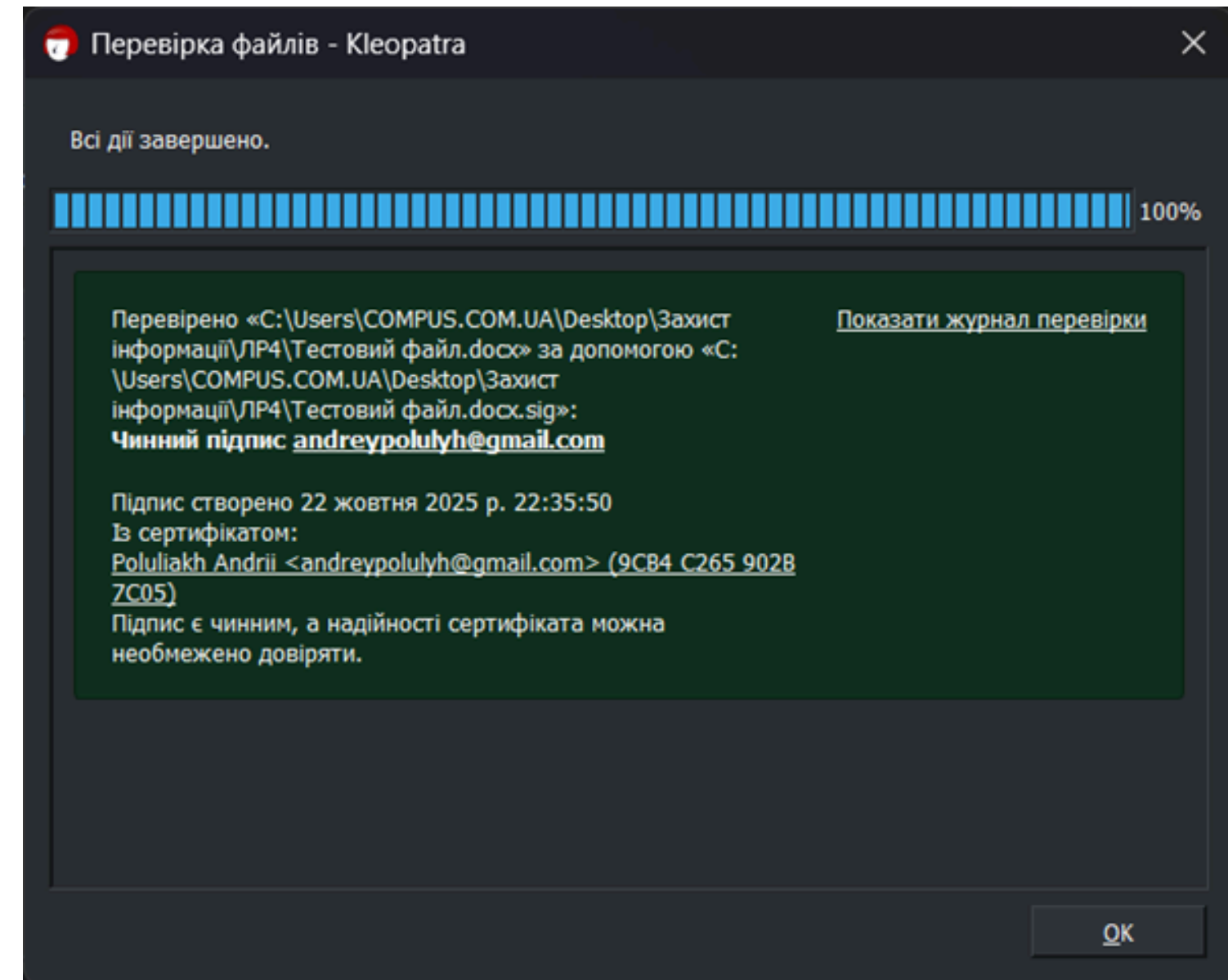
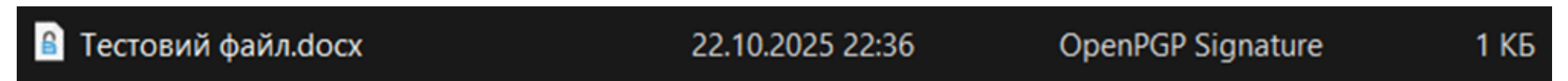
# КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

Щоб підписати тестовий файл, на панелі інструментів обираю функцію «Підписати/зашифрувати» та обираю файл. Відкривається вікно підписання файлу



# КРОК 1. ДОСЛІДЖЕННЯ ГОТОВИХ ІНСТРУМЕНТІВ ЦИФРОВОГО ПІДПИСУ

Після підписання з'являється файл з розширенням .sig, тобто файл підпису. Якщо його відкрити, бачимо інформацію про те, що підпис є чинним, а надійності сертифіката можна необмежено довіряти





## КРОК 2. СТВОРЕННЯ ВЛАСНОЇ ПАРИ КЛЮЧІВ

Для виконання завдання розробляємо просту консольну Python-програму. Програма отримує три значення від користувача: ім'я, дату народження та секретне слово

```
Введіть ваше ім'я: Andrii  
Введіть дату народження (ДД.ММ.РРРР): 09.02.2005  
Введіть секретне слово: poluliakh
```

## КРОК 2. СТВОРЕННЯ ВЛАСНОЇ ПАРИ КЛЮЧІВ

Потім вона об'єднує ці рядки в один та застосовує функцію хешування SHA-256 до нього. SHA-256 повертає 256-бітне число, яке представляється у вигляді 64 символів у шістнадцятковому форматі, тобто hex. Програма перетворює цей hex-рядок у звичайне десяткове число для подальших математичних операцій

```
=====
Створення приватного ключа
=====

Об'єднані дані: Andrii09.02.2005poluliakh

Приватний ключ (SHA-256 хеш):
d440f24c630dc31660f107d6a0dd9c7374f4d45300766c9dd9d45ef6261615ce

Приватний ключ (як число):
96005074393801739210513559987865596847159573780296367196171846152073195689422
```

## КРОК 2. СТВОРЕННЯ ВЛАСНОЇ ПАРИ КЛЮЧІВ

Публічний ключ створюється за допомогою формули  $\text{public\_key} = g^{(\text{private\_key})} \% \text{modulus}$ , де  $g = 65537$  і є генератором, який ми підносимо до степеню;  $\text{modulus} = 2^{256} - 189$  і числом, за яким береться залишок від ділення

```
=====
Створення публічного ключа
=====

Генератор (g): 65537
Модуль: 115792089237316195423570985008687907853269984665640564039457584007913129639747

Публічний ключ =  $g^{(\text{приватний\_ключ})} \bmod \text{модуль}$ 

Публічний ключ (як число):
86886954411789399870196978731480681792527145932567508960379331529951361507249

Публічний ключ (hex):
c01846012fe3b474785a5cda8b8f23aa49cb6affc0390326fe9bf093f45a1fb1
```



## КРОК 2. СТВОРЕННЯ ВЛАСНОЇ ПАРИ КЛЮЧІВ

Після створення ключі зберігаються в окремі файли

```
private_key.json > ...
1  {
2    "type": "private_key",
3    "owner": "Andrii",
4    "generated_from": "hash(name + birth_date + secret_word)",
5    "algorithm": "SHA-256",
6    "key_hex": "d440f24c630dc31660f107d6a0dd9c7374f4d45300766c9dd9d45ef6261615ce",
7    "key_number": "96005074393801739210513559987865596847159573780296367196171846152073195689422"
8  }
```

```
public_key.json > ...
1  {
2    "type": "public_key",
3    "owner": "Andrii",
4    "algorithm": "g^(private_key) mod modulus",
5    "generator_g": 65537,
6    "modulus": "115792089237316195423570985008687907853269984665640564039457584007913129639747",
7    "key_hex": "c01846012fe3b474785a5cda8b8f23aa49cb6affc0390326fe9bf093f45a1fb1",
8    "key_number": "86886954411789399870196978731480681792527145932567508960379331529951361507249"
9  }
```

# КРОК 3. ПІДПИСАННЯ ВЛАСНИХ ДОКУМЕНТІВ

 ЛРЗ_122.1_Полулях_А.П.	10.10.2025 21:53	Документ Microso...	1 783 КБ
 ЛРЗ_122.1_Полулях_А.П..docx	23.10.2025 0:57	OpenPGP Signature	1 КБ

Підписати/Зашифрувати файли - Kleopatra

Підписування або шифрування файлів

Підтвердити достовірність (підписати)

☒ Підписати від імені:

☒ Poluliakh Andrii <andreypolulyh@gmail.com> (сертифікова

Шифрування

☐ Зашифрувати для вас:

☒ Poluliakh Andrii <andreypolulyh@gmail.com> (сертифікова


☐ Зашифрувати для інших:

☐ Будь ласка, введіть ім'я або адресу електронної пошти...

☐ Зашифрувати з паролем. Дані зможе прочитати будь-хто, у кого є пароль.

Виведення

Виведення файлів/теки:

 C:\Users\COMPUS.COM.UA\Desktop\Захист інформації\ЛР4\krok3\ЛРЗ\_122.1\_Полулях\_А.П..docx.sig

☐ Зашифрувати/Підписати кожен файл окремо.

Підписати

Скасувати

Перевірка файлів - Kleopatra

Всі дії завершено.

100%

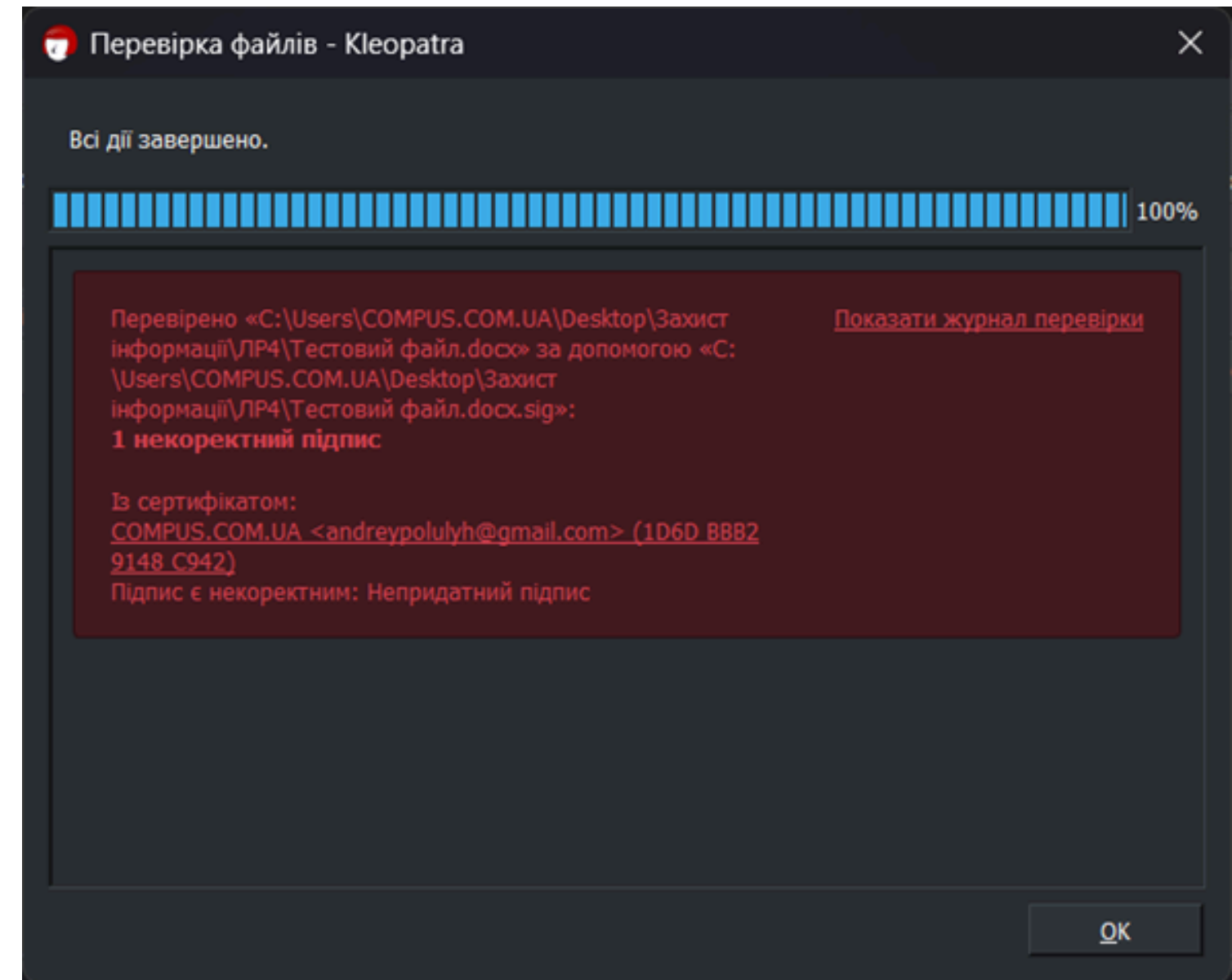
Перевірено «C:\Users\COMPUS.COM.UA\Desktop\Захист інформації\ЛР4\krok3\ЛРЗ\_122.1\_Полулях\_А.П..docx» за допомогою «C:\Users\COMPUS.COM.UA\Desktop\Захист інформації\ЛР4\krok3\ЛРЗ\_122.1\_Полулях\_А.П..docx.sig»:  
**Чинний підпис andreypolulyh@gmail.com**  
  
Підпис створено 23 жовтня 2025 р. 0:57:13  
Із сертифікатом:  
Poluliakh Andrii <andreypolulyh@gmail.com> (9CB4 C265 902B 7C05)  
Підпис є чинним, а надійності сертифіката можна необмежено довіряти.

Показати журнал перевірки

ОК

## КРОК 4. ТЕСТУВАННЯ ПІДРОБКИ ПІДПИСІВ

Якщо змінити підписаний тестовий файл та запустити файл підпису, то програма наддасть інформацію про його некоректність





## КРОК 4. ТЕСТУВАННЯ ПІДРОБКИ ПІДПИСІВ

Навіть ручне повернення файлу до оригінального стану не відновлює валідність підпису. Процес підписання працює наступним чином: спочатку обчислюється унікальний цифровий відбиток (хеш) документа за допомогою алгоритму хешування, після чого цей хеш підписується приватним ключем. Якщо змінити вміст документа, його хеш-значення зміниться, і він не буде відповідати збереженому в підписі, що призведе до помилки перевірки.





**ДЯКУЮ ЗА  
УВАГУ**