НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

з лабораторної роботи № 6

із дисципліни «Криптографічні методи захисту інформації»

на тему

*Хеш-функції та їх застосування в електронних цифрових підписах*

|  |  |
| --- | --- |
| Виконала: | Керівник: |
| студент групи КМ-01 | *ст. викладач Бай Ю. П.* |
| *Боженко А. О.* |  |

Київ — 2022

ЗМІСТ

[Постановка завдань і хід виконання лабораторної роботи 3](#_Toc105577740)

[Відповіді 5](#_Toc105577741)

[Список літератури 10](#_Toc105577742)

[Додаток 1 11](#_Toc105577743)

[Посилання на файли з кодом (розміщені в Replit) 12](#_Toc105577744)

***Мета роботи:*** засвоїти *поняття хеш-функції, властивості хеш-функцій, електронний цифровий підпис, алгоритми створення та перевірки ЕЦП*. Отримати практичні навички роботи з найбільш використовуваними на практиці хеш-функціями, отримати практичний досвід створення електронного цифрового підпису з використанням алгоритму RSA, опанувати бібліотеку *rsa* для Python.

# Постановка завдань і хід виконання лабораторної роботи

1. Дати визначення хеш-функцій і навести властивості ідеальної хеш-функції. Дати визначення колізії хеш-функції.

2. Обчислити значення хеш-функцій від заданих вхідних даних, заповнити *Таблицю 1*.

3. Створити файл *zayava.docx* від власного імені, записати Прізвище Ім’я По-батькові. Обчислити *sha1* від вмісту даного файлу, записати в *Таблицю 1.*

4. Навести теоретичні відомості з електронного цифрового підпису. Навести схему створення та верифікації ЕЦП за допомогою асиметричних алгоритмів. Створити електронний цифровий підпис до власного файлу *zayava.docx,* використовуючи алгоритм RSA та бібліотеку *rsa* для Python:

4.1. Встановити бібліотеку *rsa*:

python -m pip install rsa

pip install --upgrade rsa

4.2. Згенерувати секретний (private) і відкритий (public) ключі для алгоритму RSA – для цього необхідно один раз виконати файл ***genkeys.py.***

4.3. Записати секретний і відкритий ключі(згенеровані ключі знаходяться у файлах *private\_key\_1024.pem, public\_key\_1024.pem*) ***display\_key.py***

4.4. Приєднати **метадані** до хеш-коду від вхідного файлу. Зашифрувати одержаний результат, користуючись функцією *rsa.encrypt().* Налаштувати і запустити на виконання файл ***hash\_sign.py****.* Цифровий підпис буде записано в файл ***zayava.docx.sgn****.* Заповнити *Таблицю 2.*

4.5. Передати одержувачу документ *zayava.docx,*  цифровий підпис до нього − *zayava.docx.sgn* тапублічний ключ.

5. Імітувати дії одержувача, верифікувати цифровий підпис (див. файл ***verify\_signature.py***).

6. Створити і верифікувати цифровий підпис до документу *zayava.docx* **без метаданих**, користуючись вбудованими функціями бібліотеки rsa: *rsa.sign(), rsa.verify().* Якщо ЕЦП до файлу вірний, то функція *rsa.verify()* повертає тип хеш-функції, яка використовувалась при хешуванні вхідного файлу.

# Відповіді

1. Хеш-функція – одностороння функція, що на вході приймає дані

довільного розміру, а повертає, закодований за допомогою спеціального алгоритму, об’єкт фікcованого розміру, що називається хеш, хешсума, дайджест повідомлення.

Властивості ідеальної хеш-функції:

* вхідні дані можуть бути довільного розміру, результат хеш-функції завжди має фіксований розмір;
* простота обчислення хеш-функції як для апаратної, так і для програмної реалізацій;
* на одні й ті ж самі вхідні дані функція повертає однакове значення хешу;
* для двох різних даних значення хеш-функцій відрізняється (відсутність колізій першого та другого родів);
* односторонність хеш-функції - обчислити вхідні дані з отриманого хешу неможливо(дуже важко);
* лавиновий ефект – при малій зміні вхідних даних хеш значно змінюється;

Колізія хеш-функцій – випадок, коли для різних даних хеш однаковий: х, у, х ≠ у, hash(x) = hash(y). Розрізняють колізію першого та другого родів. Колізія першого роду – існування таких даних (y), різних за вмістом до відомих даних (х) за умови, що їхні хеші однакові: hash(x) = hash(y). Колізія другого роду – існування пари різних даних: х, у, для яких hash(x) = hash(y).

2. Визначення дайджестів (хеш-кодів) від рядків та вмісту файлу.

*Таблиця 1*

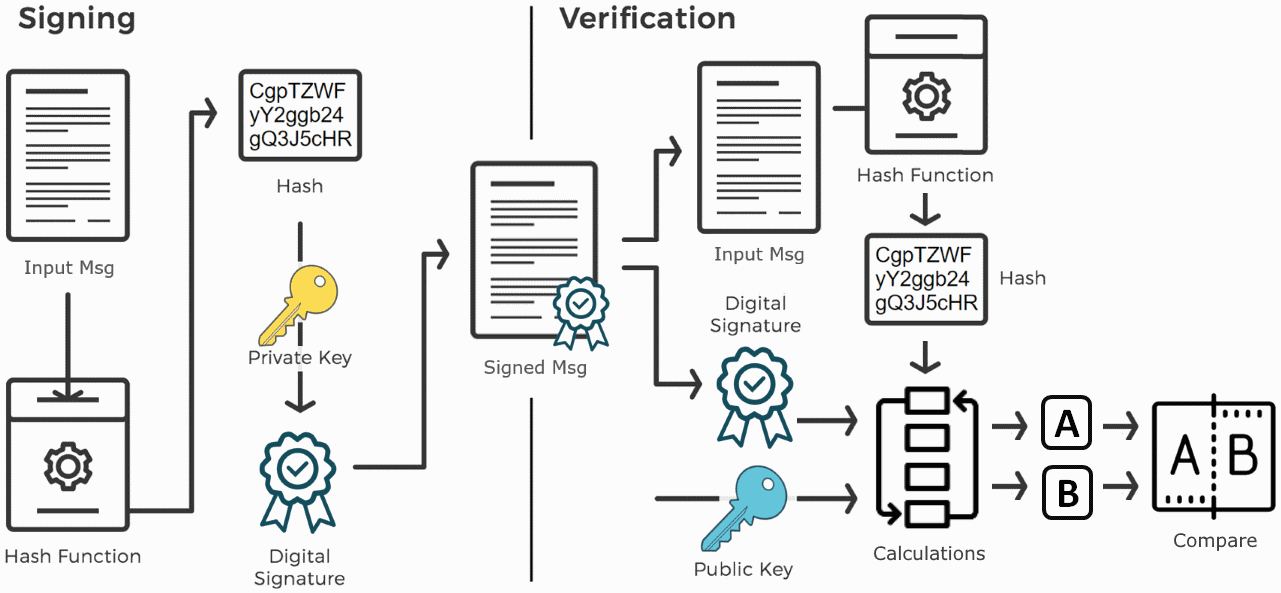
|  |  |  |  |
| --- | --- | --- | --- |
| № | Вхідні дані, *M* | *hash* | Результат,  *h(M)* |
| 1 | *‘Hello, world!’* | sha1 | 943a702d06f34599aee1f8da8ef9f7296031d699 |
| 2 | *‘12345’* | sha1 | 8cb2237d0679ca88db6464eac60da96345513964 |
| 3 | *‘12346’* | sha1 | 7c4a8d09ca3762af61e59520943dc26494f8941b |
| 4 | *‘’* | sha1 | da39a3ee5e6b4b0d3255bfef95601890afd80709 |
| 5 | *‘Боженко А. О.’* | md5 | 7f6646c639f5d8501355af2181053045 |
| 6 | *‘Боженко А. О.’* | sha1 | a35ab8ee5274f5c5e62cbd3f3dbcf3ebaebd94ad |
| 7 | *‘Боженко А. О.’* | sha224 | 63822ab0c5e1dd2a34c9e5b159841f4d436e16c21a04bf3b5bce27f1 |
| 8 | *‘Боженко А. О.’* | sha256 | 32be1596faa575098f1fe6c2ecba3bf7f09d6f651ba7492eddbfec37c318d5e6 |
| 9 | *‘Боженко А. О.’* | sha512 | bf04f990f912010411ab3b111fa8ba3e3d193c8b3c24298cbca43b6393d05aa2ee668d88425b55afe7c307d99d781b26798c07c0b0667fcc02ad8353eeb360a4 |
| 10 | *zayava.docx* | sha1 | 9ba09c08462d787a8d61fd6a5d3595ad60041ecd |

3. Див. рядок 10 в *Таблиці 1*.

4. Основні відомості з електронного цифрового підпису.

ЕЦП – це дані в електронній формі, отримані за результатами криптографічного перетворення, які додаються до інших даних або документів і забезпечують їх цілісність та ідентифікацію автора. ЕЦП має таку ж юридичну силу як і звичайний підпис. ЕЦП може мати як юридична, так і фізична особа. Електронний цифровий підпис накладається за допомогою закритого ключа та перевіряється за допомогою відкритого ключа. Обидва ключі належать автору ЕЦП. Закритий ключ – рядок символів, згенерований для конкретного користувача, що має намір створити ЕЦП, цей ключ зберігається користувачем в таємниці. Відкритий ключ – рядок символів, пов’язаний з закритим ключем математично, цей ключ автор ЕЦП передає стороні, що підтверджує ЕЦП документу. Результат перевірки ЕЦП – твердження(так/ні) про автентичність документу (документ не був змінений після підписання), належність авторові ЕЦП (власнику секретного ключа, а отже й підпису).

Схема створення ЕЦП з використанням асиметричних алгоритмів.



мал. 1 – схема створення та перевірки ЕЦП

1. Створення ключів

Ключі надаються ПІДПИСУВАЧУ (автору документа) кваліфікованим надавачом електронних довірчих послуг – акредитованим Центром сертифікації ключів. В Україні - див. <https://czo.gov.ua/ca-registry>

1. Підписання документу (дії ПІДПИСУВАЧА)

Обчислення хешу документу. Створення текту (М), що складається з хешу документу та метаданих (може бути дата створення документу, ім’я автора тощо). Шифрування тексту-М закритим ключем, що в результаті формує цифровий підпис. Передача одержувачу документу з ЕЦП та відкритого ключа.

1. Варифікація(первірка) ЕЦП (дії ОДЕРЖУВАЧА)

Розшифровування ЕЦП відкритим ключем, що повертає текст М - зашифрований хеш документу та метадані. Обчислення хеш-коду документу та порівняння його з тим, що обчислений відкритим ключем, якщо хеші однакові, то позитивна відповідь (так) на підтвердження ЕЦП, інакше - ні.

4.3. Генерація ключів.

а) ключі довжиною 16 біт

*Private Key:* **n** *= 39271 ;* **e** *= 65537 ;* **d** *= 4369 ;* **p** *= 227 ;* **q** *= 173 .*

*Public Key:* **n** *= 39271 ;* **e** *= 65537 .*

б) ключі довжиною 1024 біт

*Private Key:* **n***=101388131499222586817241082770745709194332477846614982823294275152346805910301597795738009187134254147625788823125444758020532890417331667714038257586178532379559037744680720624059441481400900876818709651592760131134917467719964180367327504710744442715019644730636828986632489564459536102946941568101270027943 ;*

**e** *= 65537 ;*

**d***=33012212003325003892352525523672775508458744598759740581141592344415040226435842277846916673699709313154198509188303793145858848415634534039548062156995854360881250202336796798506285055329630082415837410943227041505237737687466390002336536623763329492451447645951087515741187825351094420103186768153205184593 ;* **p**=*48223835530962010205120157755282746504745154750059890979743180569522707901368546469369652513120160331037831563619703298652797554357133273480663325207409031269217481 ;* **q***=2102448517064275287189822327455416565542650614591784775806056503161547415274905557632591090378822141286807235954891937043067425166783632697835503 .*

*Public Key:*

**n***=101388131499222586817241082770745709194332477846614982823294275152346805910301597795738009187134254147625788823125444758020532890417331667714038257586178532379559037744680720624059441481400900876818709651592760131134917467719964180367327504710744442715019644730636828986632489564459536102946941568101270027943 ;*

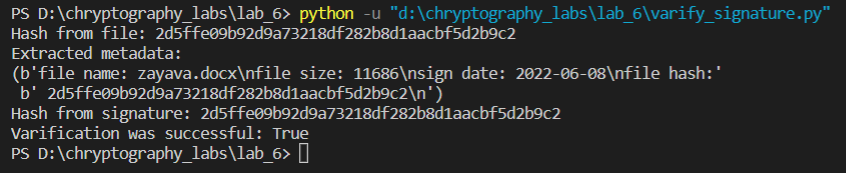
**e** *=65537 .*

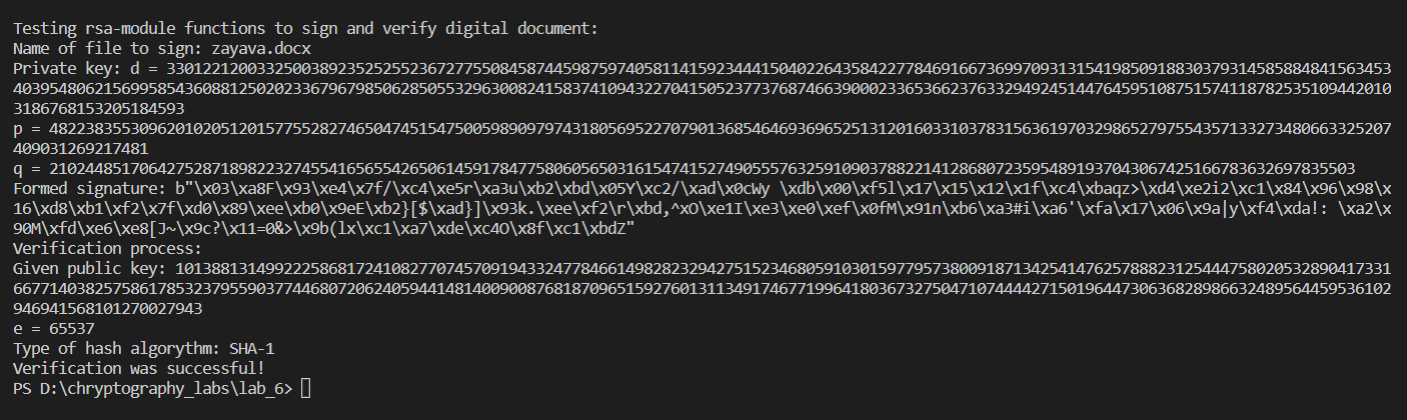
4.4. Метадані до файлу *zayava.docx* та цифровий підпис:

*Таблиця 2*

|  |  |  |
| --- | --- | --- |
| 1 | Ім’я файлу | ‘zayava.docx’ |
| 2 | Розмір файлу (в байтах) | 11686 |
| 3 | Дата підписання | 2022-06-08 |
| 4 | Хеш-код вмісту файлу | 2d5ffe09b92d9a73218df282b8d1aacbf5d2b9c2 |
| 5 | Цифровий підпис (hex) | 67d234f843a14286f3d92987ef0e8c6988561e2249  c71f739c643b49fe19752d9080b8c3b18913babf769  09b38958b62d51ec294dba0500a360748500c5bef0e  327582c61e181f58120f63b5685ca24b2d792b152a76  589468a0b4897e271fe7bcd97e2450dc54afb3e65a1e1  086e214bc72eb1ca678927b165cc28dbb1b3987 |

5. Скріншот перевірки цифрового підпису (дії одержувача)



6. Генерація та варифікація цифрового підпису за допомогою вбудованих в бібліотеку rsa функцій sign(), varify(). Ключі беремо згенеровані раніше (пункт 4.3, б). Файл, що підписувати – ‘zayava.docx’: 

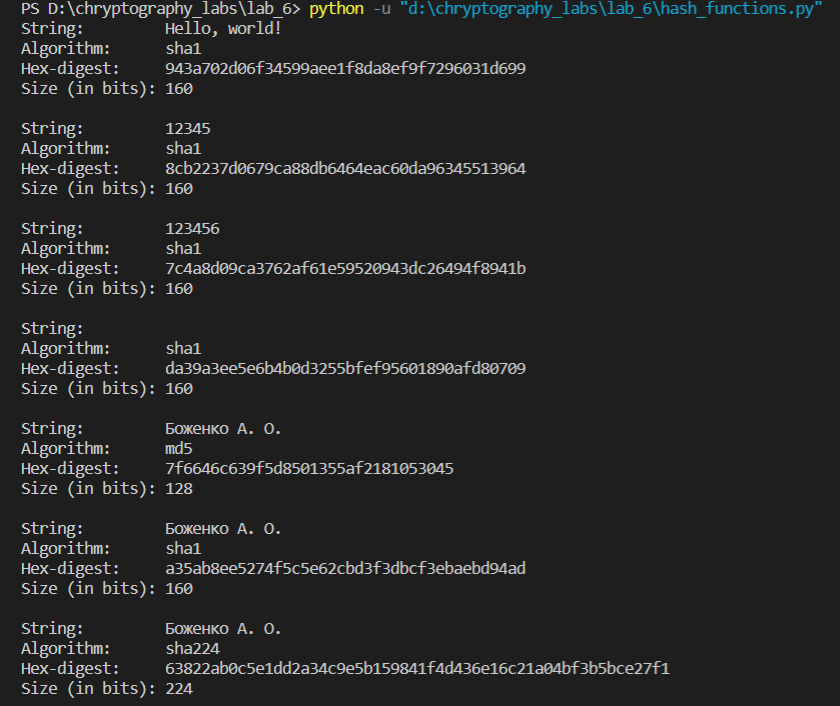
# Список літератури

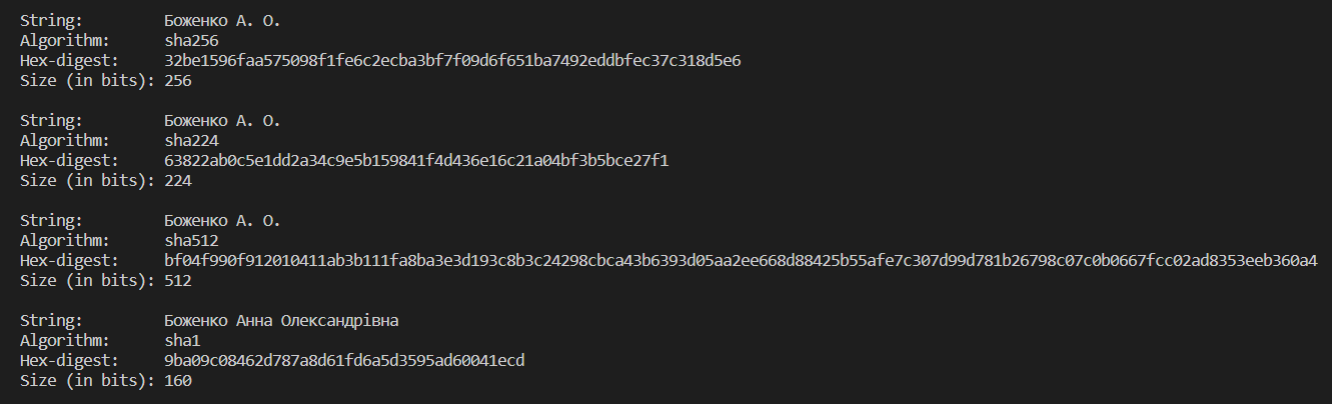
1. Тарнавський Ю.А. Технології захисту інформації [Електронний ресурс] / Ю. А. Тарнавський. – Київ: КПІ ім. Ігоря Сікорського, 2018. – 162 с.
2. Шнайер Б. Прикладная криптография: Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – М.: Диалектика, 2003. – 610 с.
3. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии. – М.: Гелиос АРВ, 2001. – 480 с.
4. Столлингс В. Криптография и защита сетей: принципы и практика, 2-е изд.: Пер. с англ. – М.: «Вильямс», 2001. – 672 с.
5. Menezes A.J., Van Oorschot P.C., Vanstone S.A. Handbook of Applied Cryptography. – CRC Press, Inc., 1997. – 795 p.
6. [Тлумачення\_ЕЦП\_з\_сайту\_урядового\_порталу](https://www.kmu.gov.ua/usi-pitannya-po-e-poslugam/sho-tak-elektronnij-cifrovij-pidpis-ecp)

# Додаток 1

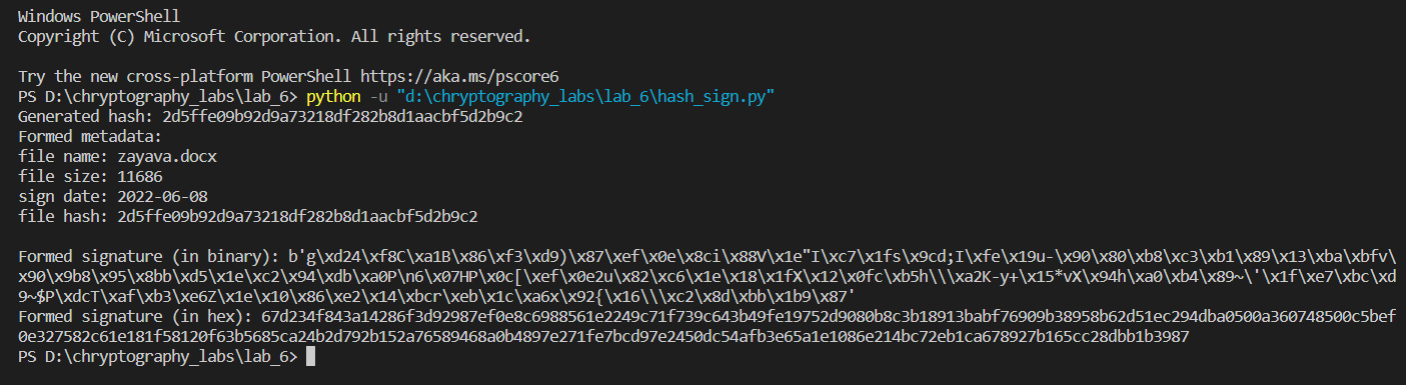
**Скріншоти виконання файлів**

Результат виконання файлу hash.py:

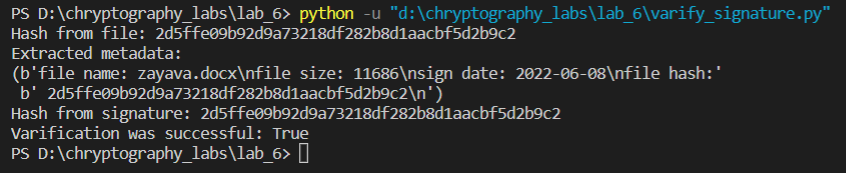




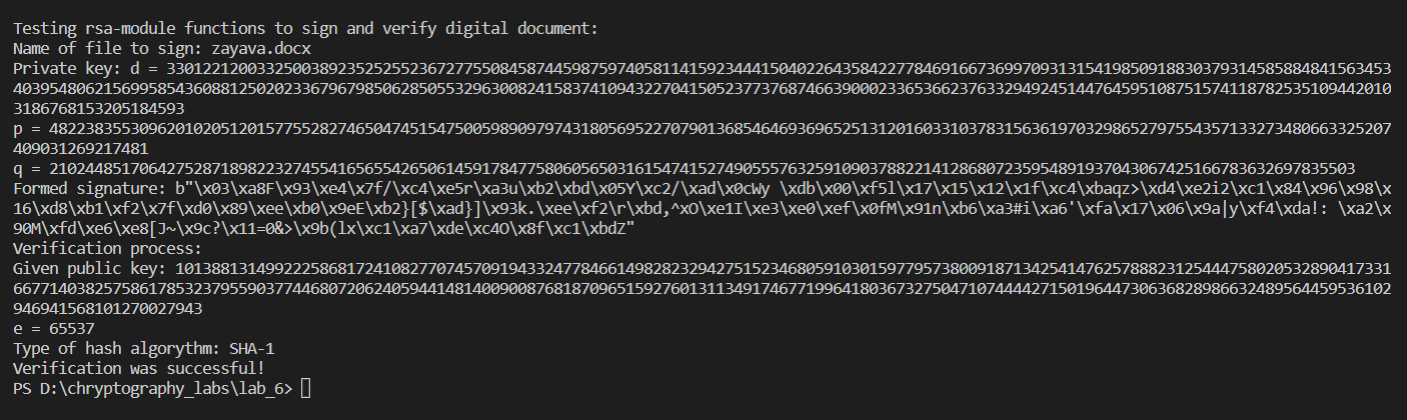
Результат виконання файлу hash\_sign.py:



Результат перевірки підпису:



Результата генерації та перевірки підпису вбудованими функціями:



# Посилання на файли з кодом (розміщені в Replit)

1. [display\_keys.py](https://replit.com/@AnnAnnAnn/lab6chryptodigitalsignature#lab_6/display_keys.py)
2. [hash.py](https://replit.com/@AnnAnnAnn/lab6chryptodigitalsignature#lab_6/hash_functions.py)
3. [hash\_sign.py](https://replit.com/@AnnAnnAnn/lab6chryptodigitalsignature#lab_6/hash_sign.py)
4. [varify\_signature.py](https://replit.com/@AnnAnnAnn/lab6chryptodigitalsignature#lab_6/varify_signature.py)
5. [gen\_keys.py](https://replit.com/@AnnAnnAnn/lab6chryptodigitalsignature#lab_6/gen_keys.py)