# Міністерство освіти і науки України Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського" Фізико-технічний інститут

Криптографія Комп'ютерний практикум №3

> Підготували: студенти групи ФБ-02 Єсаф'єв Євгеній Сапегін Валентин

# Мета роботи:

Ознайомлення з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практичне ознайомлення з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.

#### Постановка задачі:

- Написати функцію пошуку випадкового простого числа з заданого інтервалу або заданої довжини, використовуючи датчик випадкових чисел та тести перевірки на простоту. В якості датчика випадкових чисел використовуйте вбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту перевірки на простоту рекомендовано використовувати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч, використання готових реалізацій тестів не дозволяється.
- За допомогою цієї функції згенерувати дві пари простих чисел p, q i p1, q1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq ≤ p1q1; p i q прості числа для побудови ключів абонента A, p1, q1 абонента B.
- Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ (d, p, q) та відкритий ключ (n, e). За допомогою цієї функції побудувати схеми RSA для абонентів A і B тобто, створити та зберегти для подальшого використання відкриті ключі (e, n), (e1, n1) та секретні d i d1.
- Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів *A* і *B*. Кожна з операцій (шифрування, розшифрування, створення цифрового підпису, перевірка цифрового підпису) повинна бути реалізована окремою процедурою, на вхід до якої повинні подаватись лише ті ключові дані, які необхідні для її виконання За допомогою датчика випадкових чисел вибрати відкрите повідомлення *M* і знайти криптограму для абонентів *A* и *B*, перевірити правильність розшифрування. Скласти для *A* і *B* повідомлення з цифровим підписом і перевірити його.
- За допомогою раніше написаних на попередніх етапах програм організувати роботу протоколу конфіденційного розсилання ключів з підтвердженням справжності по
  - відкритому каналу за допомогою алгоритму RSA. Протоколи роботи кожного учасника
  - (відправника та приймаючого) повинні бути реалізовані у вигляді окремих процедур, на
  - вхід до яких повинні подаватись лише ті ключові дані, які необхідні для виконання. Перевірити роботу програм для випадково обраного ключа 0 < k < n.

Кожну операцію рекомендується перевіряти шляхом взаємодії із тестовим середовищем, розташованим за адресою

http://asymcryptwebservice.appspot.com/?section=rsa

# Хід роботи:

Генеруємо p, q, p1, q1 (при чому pq  $\neq$  p1q1):

```
q: 75377582710393124700210720757628726523704117994094646238195983202757295736371
p: 73663677631378435941766617475144322848313903124345113882745543388735929177389
q1: 84744505678031382420452358536678624978186159377862123837156406986108288771311
p1: 82318779325839488747096458804955644683490189815989735038807809648674072920169
```

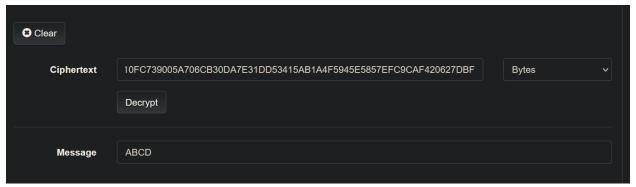
## Для юзера Lil та Big:

## Пересилання:

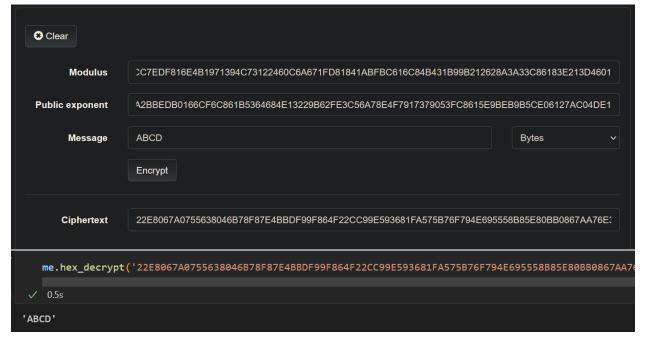
```
Generated key: 1050099265
Success
```

Все працює, переконаємось в цьому перевіркою на сайті: Зашифровуємо нашим кодом:

Розшифровуємо на сайті:



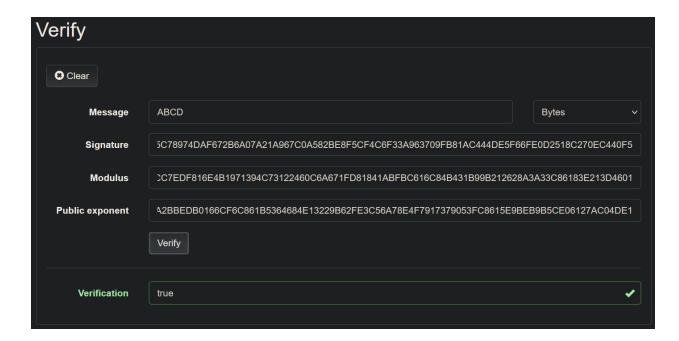
#### Робимо навпаки:



# Отримуємо підпис:



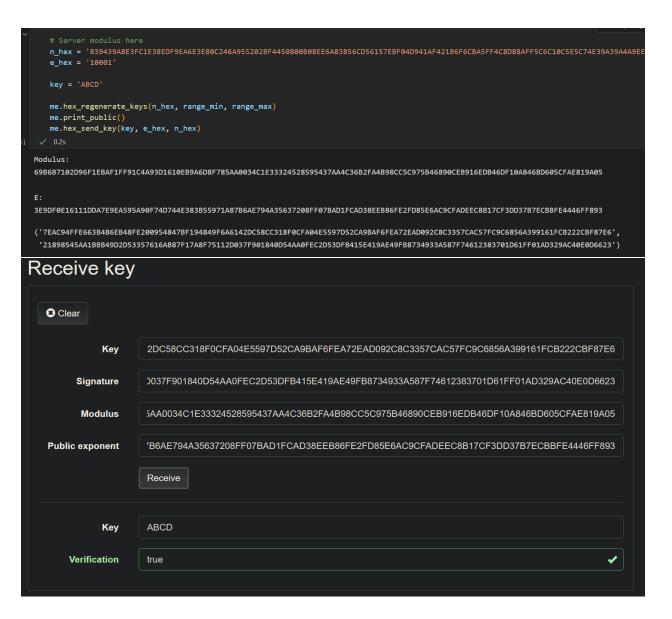
Та перевіряємо:



# Та навпаки:



Отримуємо згенерований з код ключ на сайті:



І навпаки:

Send key			
	<b>⊗</b> Clear		
	Modulus	ECD8360F3D5948DCD49FEBA0A089FF365601A5AFAED2C12069391517E6E456C4965780898432F01E5FEDB	
	Public exponent	1492A4DBFDD4030E283DF89857C85C1797BBD60BCE4746361AF2772AF1D0B4AABA7AE14CF1B8A1AD043F	
		Send	
	Кеу	4AE352FA394198BFA9AFBA4A6069AC8ED5CAA9690776758A16611394D2A60BBF25FB7A744D3CA85F30D3C	
	Signature	4E1CDF2EFD8B1146E93AF5460896F4DF279E664FB289A4034FD3C7538F7ADE2D1DD44F1A0E4EEDB2DC44	
	me.hex_r	me.hex_receive_key('4AE352FA394198BFA9AFBA4A6069AC8ED5CAA9690776758A1661139	
[45]	✓ 0.7s		
•••	Success		

Все збігається.

# Висновок:

Під час виконання практикума ми розібрали перевірку простоти чисел за допомогою тесту Міллера-Рабіна. Реалізували принцип роботи криптосистеми типу RSA, навчились шифрувати, розшифровувати, використовувати цифрові підписи, яки генерували з використанням даної системи, організували захищений зв'язок.