HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN

BÁO CÁO BÀI THỰC HÀNH
HỌC PHẦN: THỰC TẬP CƠ SỞ
MÃ HỌC PHẦN: INT13147

BÀI THỰC HÀNH 4.2
LẬP TRÌNH THUẬT TOÁN
MẬT MÃ HỌC

Sinh viên thực hiện:

B22DCAT018 – Nguyễn Hoàng Anh

Giảng viên hướng dẫn: ThS. Ninh Thị Thu Trang

HỌC KỲ 2 NĂM HỌC 2024-2025

MỤC LỤC

MỤC LỤC 2
DANH MỤC CÁC HÌNH VẼ 3
DANH MỤC BẢNG 4
CHƯƠNG 1. TÌM HIỂU LÝ THUYẾT 5
1.1 Mục đích 5
1.2 Tìm hiểu lý thuyết 5
1.2.1 Tìm hiểu về lập trình số lớn với các phép toán cơ bản 5
1.2.2 Tìm hiểu về giải thuật mật mã khóa công khai RSA 6
1.2.2.1 Giới thiệu về thuật toán mã hoá khoá công khai RSA 6
1.2.2.2 Giải thuật sinh khoá 7
1.2.2.3 Mã hoá và giải mã 7
CHƯƠNG 2. NỘI DUNG THỰC HÀNH 9
2.1 Chuẩn bị môi trường 9
2.2 Các bước thực hiện 9
2.2.1 Lập trình thư viện số lớn 9
2.2.2 Lập trình mã hoá và giải mã 12
TÀI LIỆU THAM KHẢO 17

DANH MỤC CÁC HÌNH VẼ

Hình 1 - cơ chế hoạt động của RSA 6
Hình 2 - tính gcd 9
Hình 3 - thuật toán euclid mở rộng 9
Hình 4 - tính nghịch đảo modulo 10
Hình 5 - tính luỹ thừa modulo 10
Hình 6 - tạo thư viện để sử dụng các hàm tính toán 11
Hình 7 - ví dụ thử nghiệm các hàm tính toán số lớn 11
Hình 8 - tạo số nguyên lớn 12
Hình 9 - sinh khoá rsa 12
Hình 10 - mã hoá văn bản 13
Hình 11 - giải mã 13
Hình 12 - giao diện demo 13
Hình 13 - thử nghiệm với input theo đề bài 14
Hình 14 - code demo RSA (1) 15
Hình 15 - code demo RSA (2) 16
Hình 16 - code demo RSA (3) 16

DANH MỤC BẢNG

Bảng 1 - hai phương pháp mã hoá và giải mã RSA 8

CHƯƠNG 1. TÌM HIỂU LÝ THUYẾT

1.1 Mục đích

Hiểu một giải thuật mã hóa phổ biến và lập trình được chương trình mã hóa và giải mã sử dụng ngôn ngữ lập trình phổ biến như C/C++/Python/Java, đáp ứng chạy được với số lớn.

1.2 Tìm hiểu lý thuyết

1.2.1 Tìm hiểu về lập trình số lớn với các phép toán cơ bản

Khái niệm số lớn (Big Integer): Trong lập trình, kiểu dữ liệu số nguyên chuẩn (như int, long, long long trong C/C++ hoặc int trong Java) có giới hạn về kích thước do bộ nhớ bị giới hạn. Khi cần xử lý các số có hàng trăm, hàng ngàn chữ số (ví dụ trong mật mã học, tính toán khoa học, hoặc xử lý số nguyên lớn trong toán học), ta cần sử dụng kỹ thuật lập trình số lớn (Big Integer).

Biểu diễn số lớn: Do không thể lưu trữ số lớn bằng kiểu dữ liệu nguyên thủy, ta phải biểu diễn số lớn dưới dạng mảng các chữ số hoặc xâu ký tự. Ví dụ: Số 12345678901234567890 có thể được lưu dưới dạng chuỗi "12345678901234567890" hoặc mảng int[] = {1,2,3,...,0}.

Các phép toán cơ bản với số lớn:

- Cộng hai số lớn: Duyệt từ phải sang trái (tức từ hàng đơn vị đến hàng cao nhất), cộng từng cặp chữ số cùng vị trí. Giữ biến nhớ để cộng dồn nếu tổng lớn hơn 9. Kết quả được ghép lại thành chuỗi/mảng mới.
- Trừ hai số lớn: Tương tự phép cộng, nhưng xử lý trường hợp mượn khi chữ số bị trừ nhỏ hơn chữ số trừ. Cần đảm bảo số bị trừ lớn hơn hoặc bằng số trừ, hoặc xử lý dấu âm.
- Nhân hai số lớn: Sử dụng thuật toán nhân truyền thống (nhân từng chữ số và cộng dồn các kết quả trung gian). Có thể tối ưu bằng các thuật toán nhanh hơn như Karatsuba, nhưng với cấp độ cơ bản, nhân thủ công là đủ.
- Chia số lớn: Có thể mô phỏng phép chia tay như chia giấy nháp. Thường sẽ so sánh từng đoạn của số bị chia với số chia để tìm thương và phần dư.

Ngôn ngữ hỗ trợ:

- C/C++: Không hỗ trợ sẵn, cần lập trình thủ công hoặc dùng thư viện như BigInteger của GMP.
- Java: Có lớp BigInteger trong gói java.math, hỗ trợ sẵn cộng, trừ, nhân, chia, so sánh, mod, pow, gcd...

- Python: Hỗ trợ số lớn mặc định, có thể tính toán số rất lớn trực tiếp với kiểu int.

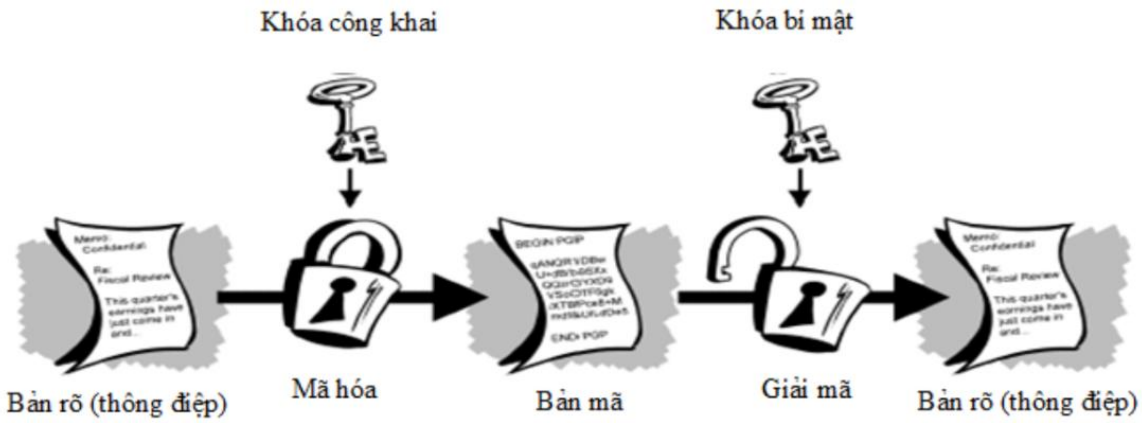
Ứng dụng:

- Mật mã học (RSA, Diffie-Hellman...)
- Tính toán số nguyên lớn (số nguyên tố lớn, giai thừa lớn).
- Hệ thống tài chính, khoa học máy tính, tính toán khoa học...

1.2.2 Tìm hiểu về giải thuật mật mã khóa công khai RSA

1.2.2.1 Giới thiệu về thuật toán mã hoá khoá công khai RSA

Thuật toán mã hoá RSA có cơ chế mã hóa khoá công khai, sử dụng cặp khóa: khoá công khai và khóa riêng tư để mã hóa và giải mã thông tin. Khóa công khai có thể được chia sẻ rộng rãi để mã hóa dữ liệu, trong khi khóa riêng tư được giữ bí mật và chỉ người sở hữu mới có để có thể giải mã.



Hình 1 - cơ chế hoạt động của RSA

Hệ mật sẽ bao gồm:

- Bản rõ (thông điệp): bản tin được sinh ra bởi bên gửi
- Bản mã: bản tin che giấu thông tin của bản rõ, được gửi tới bên nhận qua một kênh truyền không bí mật
- Khóa:
  - Khóa công khai: công bố cho tất cả mọi người biết
  - Khóa riêng: bên nhận giữ bí mật, không chia sẻ cho bất kỳ ai
- Mã hóa (encrypt)

- Giải mã (decrypt)

Cơ chế hoạt động:

- Người gửi gửi thông tin đã được mã hóa bằng khóa công khai của người nhận thông qua kênh truyền tin không bí mật
- Người nhận sẽ nhận được thông tin đó và giải mã bằng khóa riêng của mình.
- Hacker cũng sẽ biết khóa công khai của B tuy nhiên do không có khóa riêng nên Hacker không thể xem được thông tin gửi

1.2.2.2 Giải thuật sinh khoá

Giả sử mỗi bên liên lạc A và B cần trao đổi thông tin bí mật thông qua một kênh không an toàn. Với thuật toán RSA, mỗi bên liên lạc (A, B) cần tự tạo cho mình một cặp khóa công khai, bí mật theo các bước sau:

- Chọn 2 số nguyên tố lớn p và q và tính N = pq.

  Cần chọn p và q sao cho: M < 2i-1 < N < 2i với i là chiều dài bản rõ.
- Tính Φ(n) = (p - 1)(q - 1)
- Tìm số e sao cho:
  - e và Φ(n) là 2 số nguyên tố cùng nhau
  - 0 < e < Φ(n)
- Tìm số d sao cho: e.d ≡ 1 mod Φ(n) (d là nghịch đảo của e trong phép modulo Φ(n))
- Ta có:
  - Khóa công khai KU là cặp (e, N)
  - Khoá bí mật KR là cặp (d, N)

1.2.2.3 Mã hoá và giải mã

Mã hóa: là quá trình biến đổi thông tin ban đầu thành dạng mã hoặc mã hóa, mà chỉ những người có kiến thức, khóa hoặc phương pháp tương ứng mới có thể giải mã và đọc được nội dung ban đầu.

Giải mã: là quá trình chuyển đổi dữ liệu đã được mã hóa (hoặc mã hóa) thành dạng có thể đọc được, được hiểu bởi con người hoặc máy tính.

Có 2 phương pháp mã hoá và giải mã:

Bảng 1 - hai phương pháp mã hoá và giải mã RSA

	Mã hoá bảo mật	Mã hoá chứng thực
Mã hoá	$C = M^e \bmod n$	$C = M^d \bmod n$
Giải mã	$M = C^d \bmod n$	$M = C^e \bmod n$

CHƯƠNG 2. NỘI DUNG THỰC HÀNH

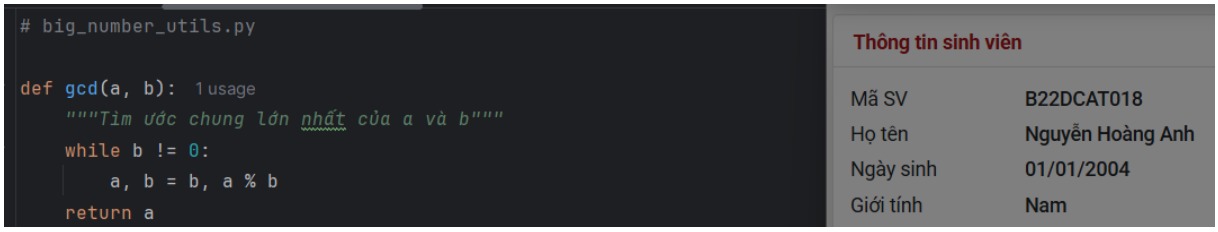
2.1 Chuẩn bị môi trường

Môi trường lập trình Python

2.2 Các bước thực hiện

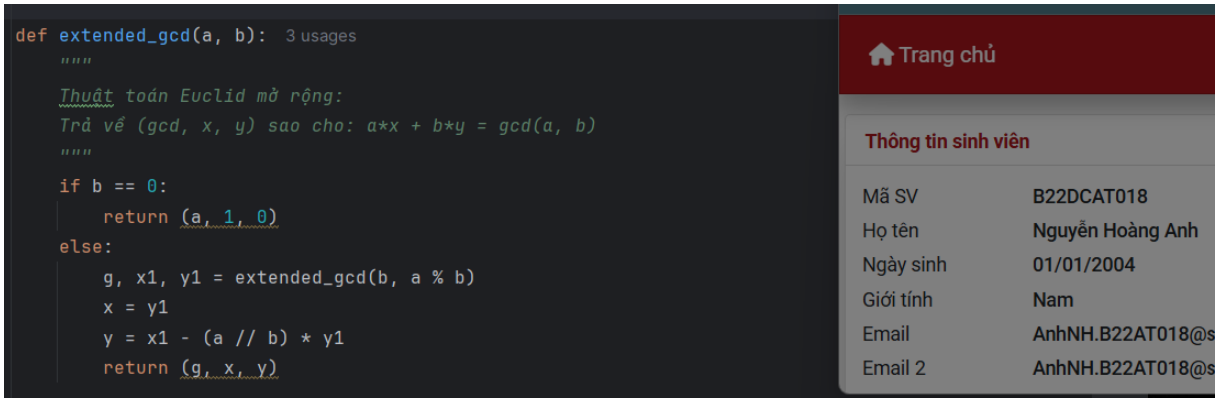
2.2.1 Lập trình thư viện số lớn

Hàm gcd: tính ước chung lớn nhất của 2 số a, b bằng thuật toán Euclid



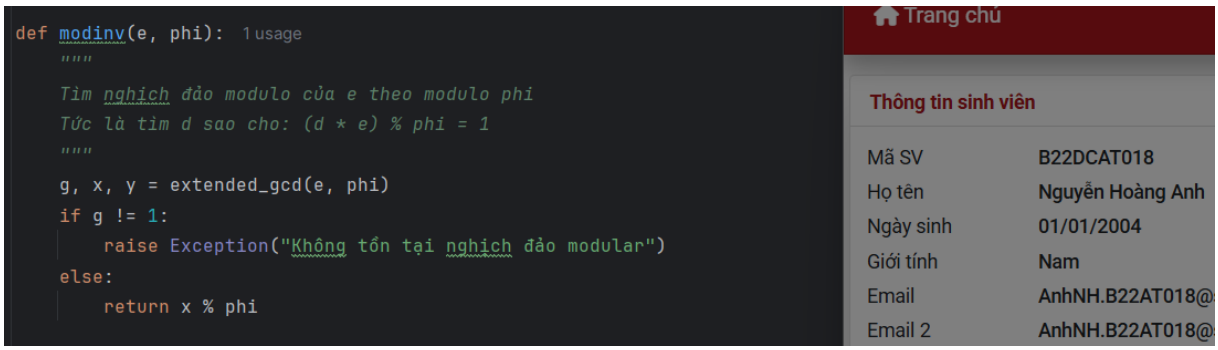
Hình 2 - tính gcd

Hàm extended_gcd (thuật toán Euclide mở rộng): Tính ước chung lớn nhất của a và b và đồng thời tìm các hệ số x và y sao cho: a.x + b.y = gcd(a, b)



Hình 3 - thuật toán euclid mở rộng

Hàm modinv: Tìm nghịch đảo modulo của số e modulo phi (Nghĩa là tìm d sao cho: (d.e) % phi = 1)

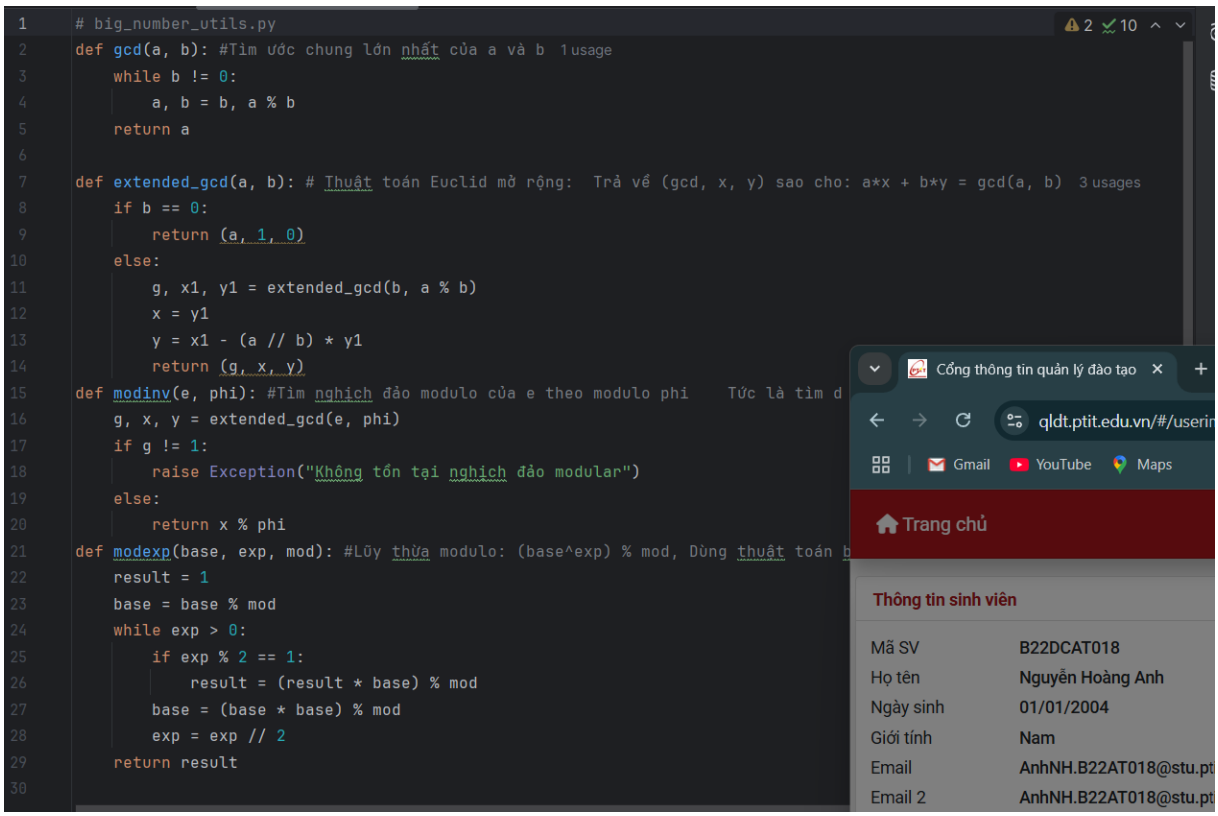


Hình 4 - tính nghịch đảo modulo

Hàm modexp: Tính lũy thừa của base với số mũ exp theo modulo mod, tức là tính: (base^exp) % mod

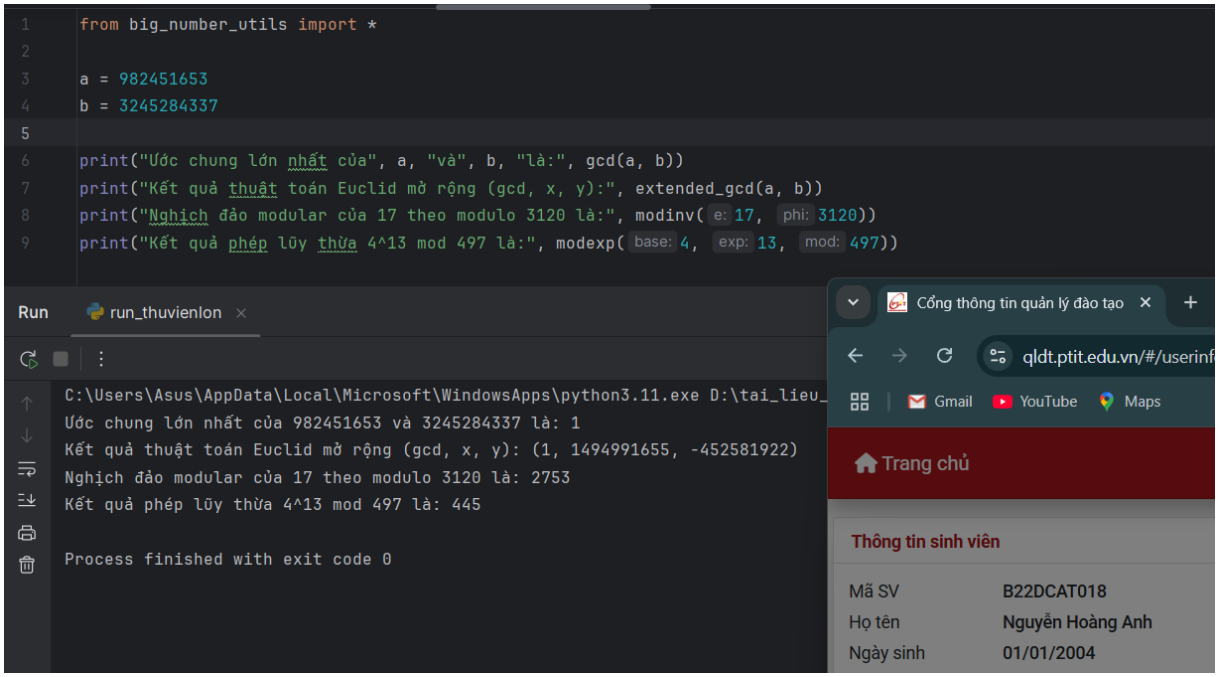
Hình 5 - tính luỹ thừa modulo

Từ các hàm trên ta có thư viện số lớn



Hình 6 - tạo thư viện để sử dụng các hàm tính toán

Thử nghiệm chứng minh thư viện hoạt động tốt

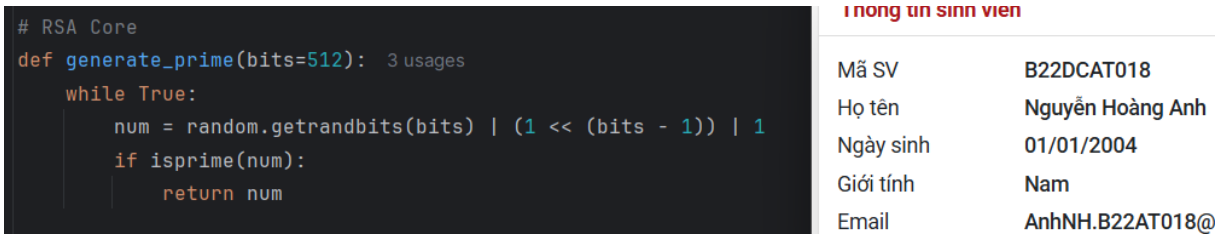


Hình 7 - ví dụ thử nghiệm các hàm tính toán số lớn

2.2.2 Lập trình mã hoá và giải mã

Hàm generate_prime: tạo số nguyên tố lớn

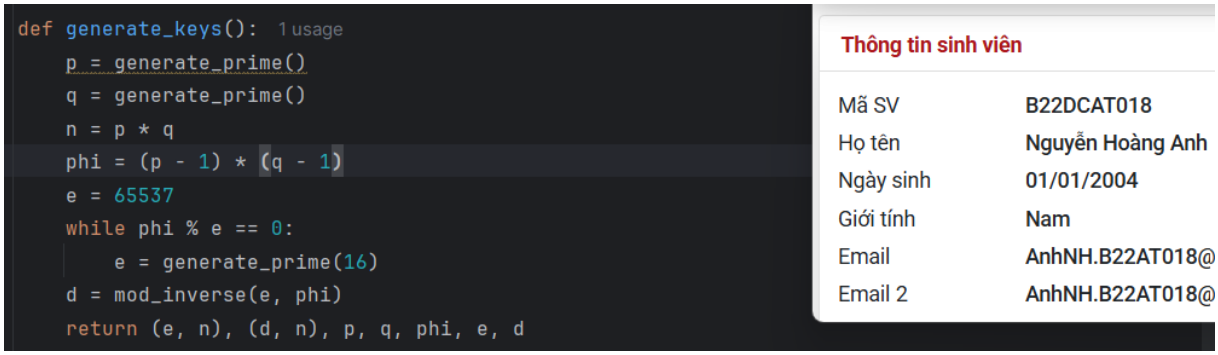
- Sinh một số ngẫu nhiên có độ dài 512 bits
- Đảm bảo số này là số lẻ và có bit cao nhất = 1 để đạt đúng độ dài.
- Lặp lại đến khi tìm được số nguyên tố.
- Dùng để tạo p và q trong thuật toán RSA.



Hình 8 - tạo số nguyên lớn

Hàm generate_keys: sinh khoá RSA

- Tạo 2 số nguyên tố p, q
- Tính n = p.q và φ(n) = (p-1).(q-1)
- Chọn số mũ công khai e = 65537
- Tính khoá bí mật d = e^{-1} mod φ(n)
- Trả về các giá trị cần thiết: cặp khoá (e,n), (d,n), p, q, phi, e, d.



Hình 9 - sinh khoá rsa

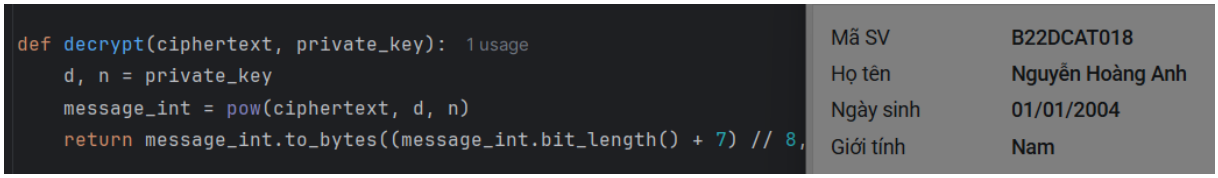
Hàm encrypt: mã hoá văn bản

- Chuyển chuỗi ký tự thành số nguyên (int.from_bytes)
- Mã hoá bằng công thức cipher = message^e mod n
- Trả về bản mã hoá dạng số nguyên.

Hình 10 - mã hoá văn bản

Hàm decrypt: giải mã

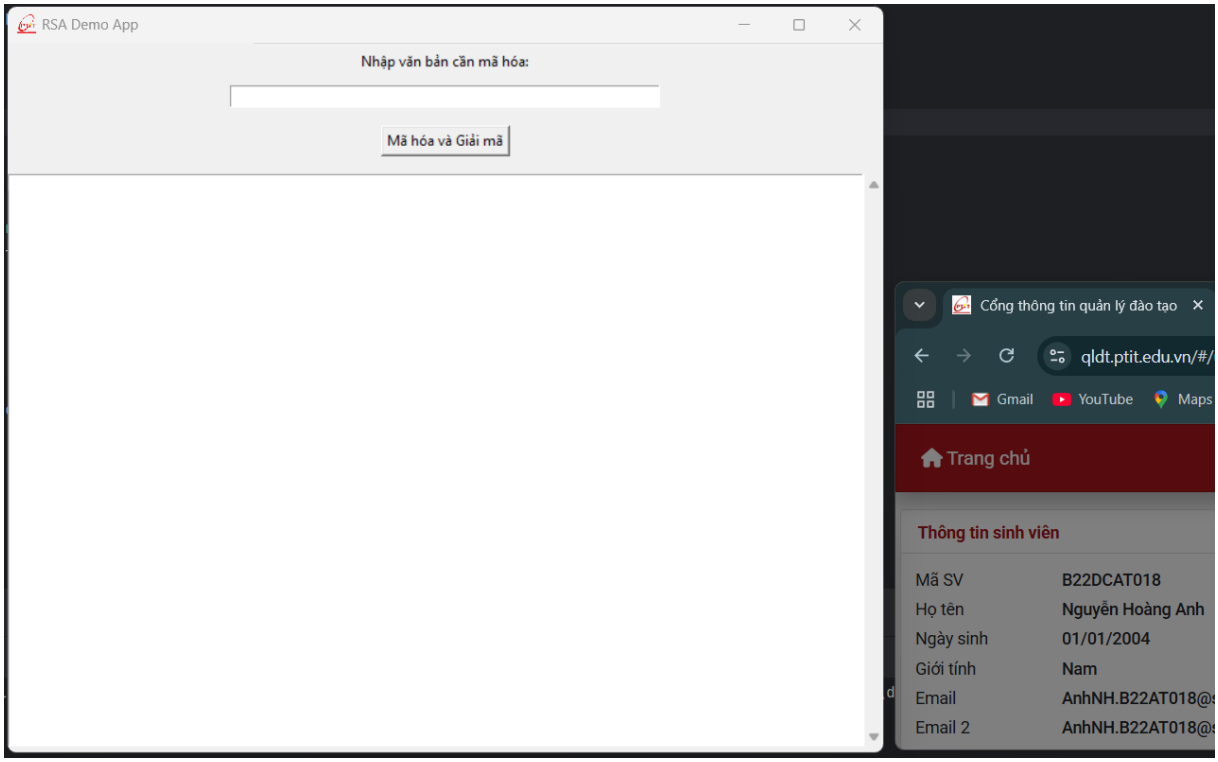
- Giải mã bằng công thức message = cipher^d mod n
- Chuyển số nguyên ngược lại thành chuỗi (.to_bytes().decode()).



Hình 11 - giải mã

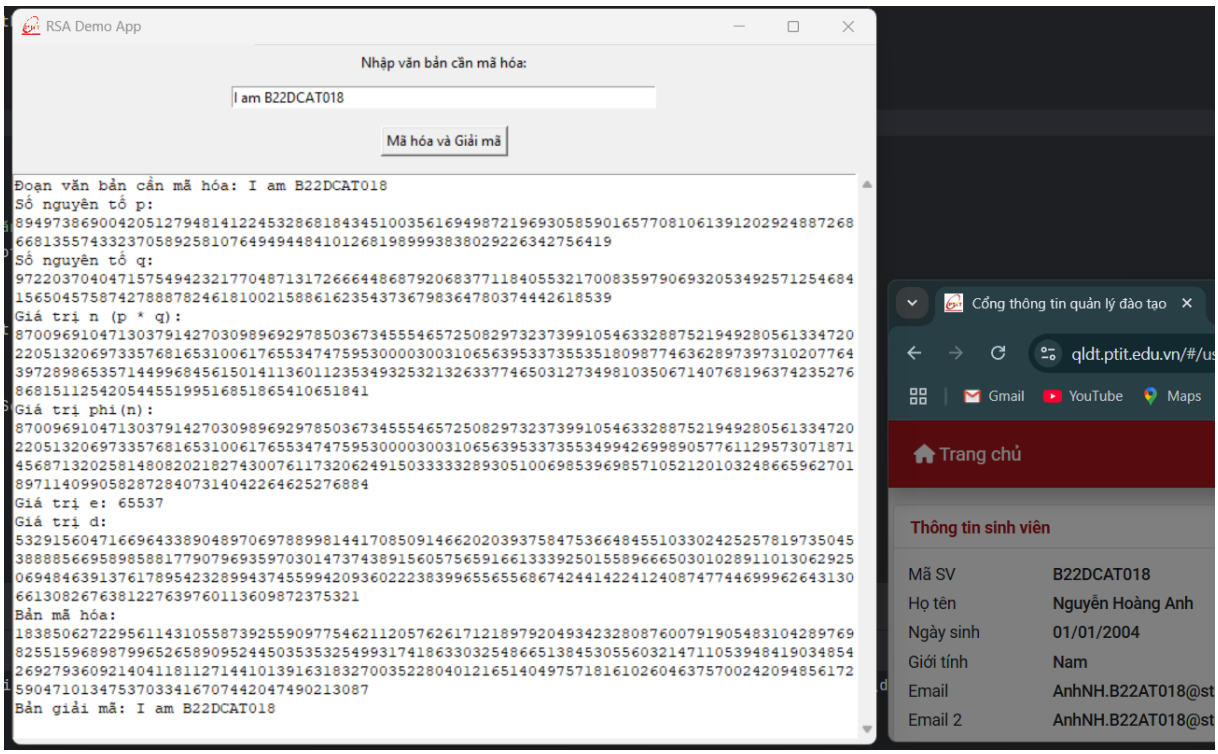
Thử nghiệm mã hoá và giải mã chuỗi kí tự: “I am B22DCAT018”

Giao diện khi chạy code (sử dụng thư viện tkinter)



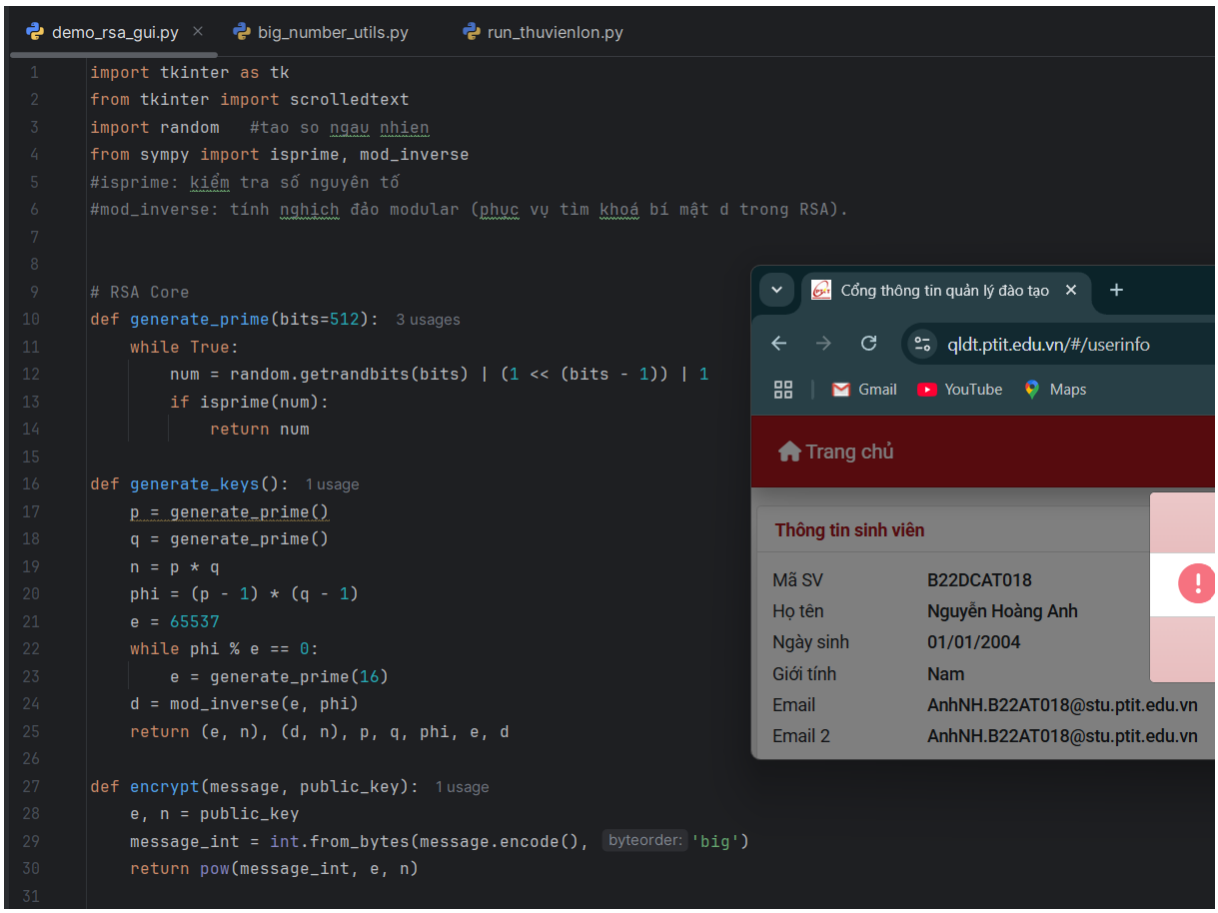
Hình 12 - giao diện demo

Nhập input: “I am B22DCAT018” và ấn nút “mã hoá và giải mãi”



Hình 13 - thử nghiệm với input theo đề bài

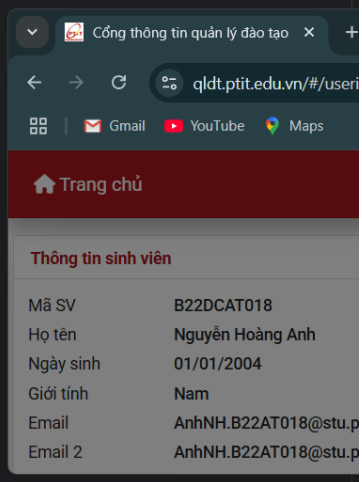
Phần code thử nghiệm RSA:



Hình 14 - code demo RSA (1)

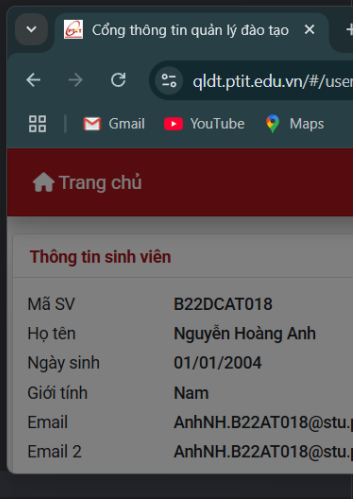
demo_rsa_gui.py | big_number_utils.py | run_thuvienlon.py

```python
def decrypt(ciphertext, private_key):  1 usage
    d, n = private_key
    message_int = pow(ciphertext, d, n)
    return message_int.to_bytes((message_int.bit_length() + 7) // 8, byteorder: 'big').decode()


# Giao diện
def process_rsa():  1 usage
    message = entry_message.get()
    if not message:
        return

    public_key, private_key, p, q, phi, e, d = generate_keys()

    try:
        ciphertext = encrypt(message, public_key)
        decrypted = decrypt(ciphertext, private_key)

        text_result.delete( index1: 1.0, tk.END)
        text_result.insert(tk.END, chars: f"Đoạn văn bản cần mã hóa: {message}\n")
        text_result.insert(tk.END, chars: f"Số nguyên tố p: {p}\n")
        text_result.insert(tk.END, chars: f"Số nguyên tố q: {q}\n")
        text_result.insert(tk.END, chars: f"Giá trị n (p * q): {p * q}\n")
        text_result.insert(tk.END, chars: f"Giá trị phi(n): {phi}\n")
        text_result.insert(tk.END, chars: f"Giá trị e: {e}\n")
        text_result.insert(tk.END, chars: f"Giá trị d: {d}\n")
        text_result.insert(tk.END, chars: f"Bản mã hóa: {ciphertext}\n")
        text_result.insert(tk.END, chars: f"Bản giải mã: {decrypted}\n")
```



Hình 15 - code demo RSA (2)

```python
    except Exception as e:
        text_result.delete( index1: 1.0, tk.END)
        text_result.insert(tk.END, chars: f"Lỗi: {e}")


# UI
root = tk.Tk()
root.title("RSA Demo App")
root.iconbitmap('ptit.ico')

tk.Label(root, text="Nhập văn bản cần mã hóa:").pack(pady=5)
entry_message = tk.Entry(root, width=60)
entry_message.pack(pady=5)

btn_process = tk.Button(root, text="Mã hóa và Giải mã", command=process_rsa)
btn_process.pack(pady=10)

text_result = scrolledtext.ScrolledText(root, width=90, height=30, wrap=tk.WORD)
text_result.pack(pady=5)

root.mainloop()
```



Hình 16 - code demo RSA (3)

TÀI LIỆU THAM KHẢO

1. Đinh Trường Duy, Phạm Hoàng Duy, Bài giảng Hệ điều hành Windows và Linux/Unix, Học viện Công Nghệ Bưu Chính Viễn Thông, 2022.
2. Đỗ Xuân Chợ, Bài giảng Mật mã học cơ sở, Học viện Công Nghệ Bưu Chính Viễn Thông, 2021