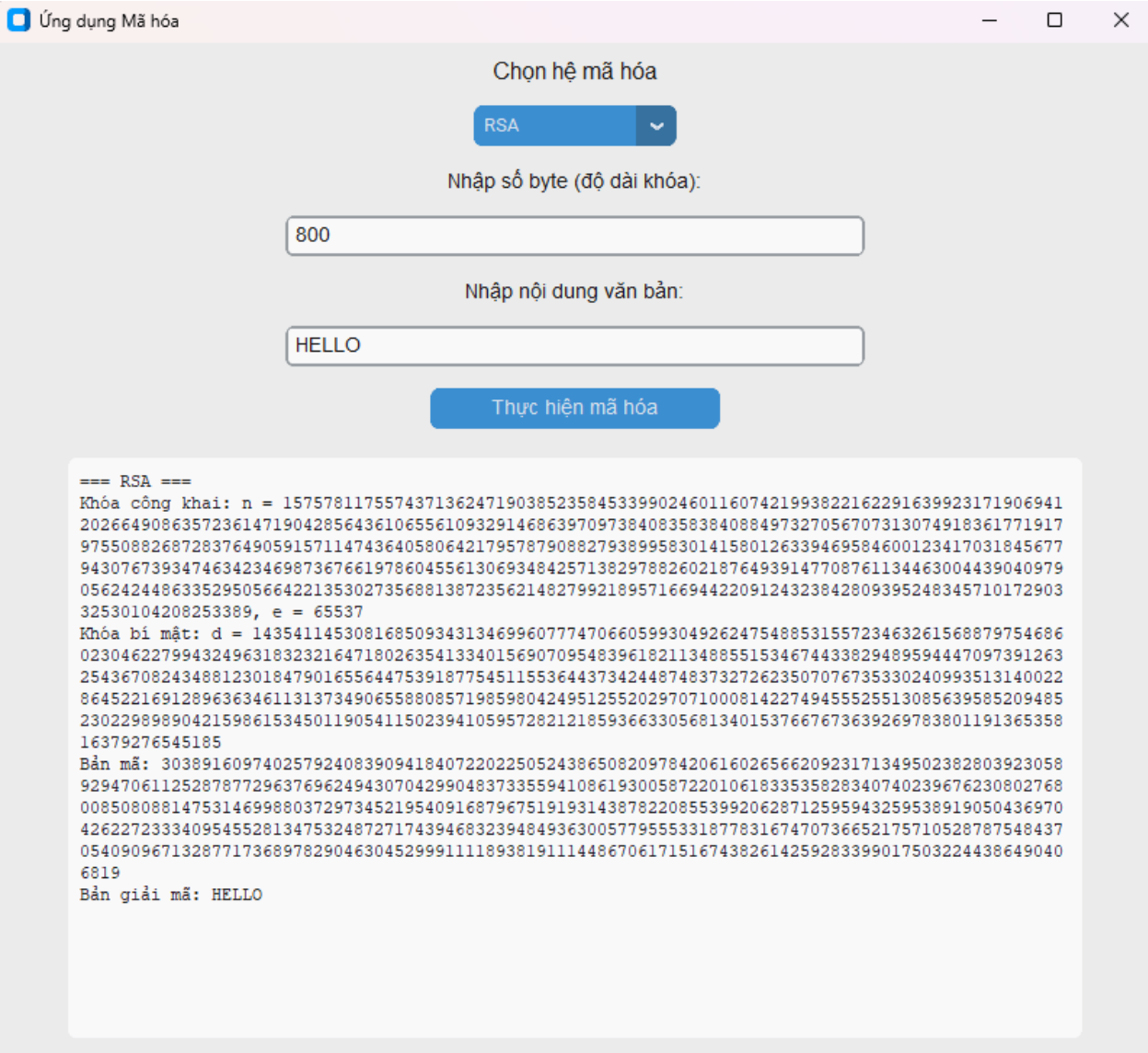
Ứng dụng mã hóa của bạn cung cấp một nền tảng đa dạng để thực hiện và so sánh các hệ mã hóa hiện đại như **RSA**, **ElGamal**, và **Elliptic Curve Cryptography (ECC)**. Với giao diện đồ họa thân thiện, người dùng có thể dễ dàng nhập thông tin cần mã hóa, chọn hệ mã phù hợp, và nhận kết quả trực tiếp. Đây là mô tả chi tiết về ứng dụng:



---

1. Mục đích

Ứng dụng được thiết kế để:

- **Mã hóa và giải mã dữ liệu an toàn**.
- **Hỗ trợ nhiều thuật toán mã hóa** hiện đại nhằm tăng tính bảo mật.
- **Trực quan hóa** quy trình mã hóa, giải mã và quản lý khóa.

---

2. Chức năng chính

- **Hỗ trợ các thuật toán mã hóa**:
    1. **RSA**:
        - Dựa trên số nguyên tố lớn và độ khó của bài toán phân tích thừa số.
    2. **ElGamal**:
        - Áp dụng trên số học modulo với cơ sở lý thuyết lũy thừa rời rạc.
    3. **ECC (Elliptic Curve Cryptography)**:
        - Sử dụng các điểm trên đường cong elliptic để mã hóa dữ liệu.
- **Giao diện đồ họa**:
    - Người dùng có thể nhập tin nhắn cần mã hóa, chọn độ dài khóa và thuật toán.
    - Hiển thị chi tiết khóa, kết quả mã hóa và giải mã.

---

3. Thành phần chính

a. Giao diện người dùng (File demo.py)

- **Giao diện bằng CustomTkinter**:
    - **Menu chọn thuật toán mã hóa**: RSA, ElGamal, ECC.
    - **Ô nhập tin nhắn**: Văn bản cần mã hóa.
    - **Ô nhập độ dài khóa**: (ví dụ: 1024 bit cho RSA).
    - **Kết quả đầu ra**:
        - Hiển thị khóa công khai, khóa bí mật, bản mã, và bản giải mã.
- **Nút thực thi mã hóa**:
    - Kích hoạt thuật toán mã hóa dựa trên tùy chọn của người dùng.

b. Hệ mã RSA (File He_mat_RSA.py)

- **Tính năng**:
    - **Sinh khóa**: Khóa công khai (n,e) và khóa bí mật (n,d).
    - **Mã hóa**:
        - Văn bản được chuyển đổi sang số và mã hóa bằng khóa công khai.
    - **Giải mã**:
        - Tin nhắn mã hóa được giải mã bằng khóa bí mật để khôi phục lại văn bản gốc.

c. Hệ mã ElGamal (File He_mat_ElGamal.py)

- **Tính năng**:

    - **Sinh khóa**: Khóa công khai gồm p,g,y và khóa bí mật x.
    - **Mã hóa**:
        - Sử dụng lũy thừa rời rạc với khóa công khai để tạo c1 và c2.
    - **Giải mã**:
        - Sử dụng khóa bí mật để tính toán lại thông điệp gốc.

d. Hệ mã ECC (File He_mat_Elliptic.py)

- **Tính năng**:
    - **Tính toán trên đường cong elliptic**:
        - Dựa trên phép nhân điểm và phép cộng điểm.
    - **Mã hóa**:
        - Sử dụng điểm cơ sở P, khóa bí mật sss để tạo kết quả mã hóa.

---

4. Quy trình hoạt động

Bước 1: Chọn hệ mã

Người dùng chọn một trong ba hệ mã được hỗ trợ:

- **RSA**: Dựa trên số nguyên tố.
- **ElGamal**: Áp dụng lý thuyết modulo.
- **ECC**: Sử dụng toán học đường cong elliptic.

Bước 2: Nhập thông tin

- **Tin nhắn cần mã hóa**: Văn bản văn bản ASCII.
- **Độ dài khóa**: Số bit khóa (ví dụ: 1024 cho RSA, 256 cho ECC).

Bước 3: Mã hóa

- Thuật toán sinh khóa công khai và bí mật.
- Tin nhắn được mã hóa bằng khóa công khai.
- Hiển thị bản mã và thông tin chi tiết.

Bước 4: Giải mã

- Dùng khóa bí mật để giải mã và khôi phục lại văn bản gốc.

---

5. Ví dụ minh họa

Mã hóa tin nhắn "HELLO" với RSA:

1. **Thông tin nhập**:
    - Tin nhắn: HELLO.
    - Độ dài khóa: 1024 bit.
2. **Kết quả**:
    - **Khóa công khai**: n=134123...341,e=65537
    - **Khóa bí mật**: d=8923...921.
    - **Bản mã**: 12982349234.
    - **Bản giải mã**: HELLO.

---

6. Ứng dụng thực tiễn

- **Gửi tin nhắn bảo mật**: Mã hóa tin nhắn quan trọng.
- **Thử nghiệm thuật toán mã hóa**: So sánh hiệu quả các hệ mã.
- **Học tập và nghiên cứu**: Ứng dụng trong việc giảng dạy mã hóa.

---

7. Điểm nổi bật

- **Tính linh hoạt**: Hỗ trợ nhiều thuật toán mã hóa.
- **Bảo mật cao**: Dựa trên các thuật toán mã hóa hiện đại.
- **Giao diện thân thiện**: Phù hợp cho cả người học và người nghiên cứu.