**Tổng Hợp Chi Tiết Chương Trình Minh Họa Diffie-Hellman**

Chương trình của bạn có thể được chia thành 4 phần chính:

1. **Thiết lập ban đầu:** Khai báo thư viện và chuẩn bị các tài nguyên.
2. **Lớp DiffieHellmanApp:** "Bộ não" chính, chứa toàn bộ logic và giao diện.
3. **Các hàm xử lý:** Các "công thức" thực hiện các nhiệm vụ cụ thể khi người dùng tương tác.
4. **Điểm khởi chạy:** Nơi chương trình bắt đầu chạy.

**Phần 1: Thiết lập ban đầu (Dòng 1-13)**

Python

# -\*- coding-utf-8 -\*-

# CHƯƠG TRÌNH MINH HỌA THUẬT TOÁN TRAO ĐỔI KHÓA DIFFIE-HELLMAN

# --- 1. KHAI BÁO CÁC THƯ VIỆN ---

import tkinter as tk

from tkinter import ttk, messagebox

import ttkbootstrap as ttk

from ttkbootstrap.constants import \*

import math

import secrets

* **Mục đích:** Khai báo các thư viện cần thiết.
* **Giải thích:**
  + tkinter, ttk, ttkbootstrap: Cung cấp các công cụ để xây dựng giao diện người dùng (cửa sổ, nút bấm, ô nhập liệu).
  + messagebox: Dùng để hiển thị các hộp thoại thông báo, cảnh báo.
  + math: Cung cấp các hàm toán học.
  + secrets: Dùng để tạo ra các số ngẫu nhiên an toàn cho việc tạo khóa bí mật.

**Phần 2: Lớp DiffieHellmanApp và Hàm khởi tạo \_\_init\_\_ (Dòng 16-43)**

Python

class DiffieHellmanApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Minh Họa Thuật Toán Trao Đổi Khóa Diffie-Hellman")

try:

self.root.iconbitmap('logo.ico')

except tk.TclError:

print("Lưu ý: Không tìm thấy file 'logo.ico'.")

self.preset\_groups = {

"p = 23, g = 5": (23, 5),

# ... các cặp (p, g) khác

}

self.create\_widgets()

self.center\_window()

* **Mục đích:** Đây là phần trung tâm của chương trình. Hàm \_\_init\_\_ sẽ chạy ngay khi chương trình khởi động.
* **Giải thích:**
  + class DiffieHellmanApp:: Định nghĩa một "bản thiết kế" cho toàn bộ ứng dụng.
  + self.root = root: Lưu lại cửa sổ chính để có thể thao tác (đặt tiêu đề, kích thước...).
  + try...except: Cố gắng đặt logo cho cửa sổ từ file logo.ico. Nếu không tìm thấy file, nó sẽ báo lỗi trong console nhưng chương trình vẫn chạy bình thường.
  + self.preset\_groups: Tạo một từ điển (dictionary) chứa các cặp (p, g) có sẵn để người dùng lựa chọn.
  + self.create\_widgets(): Gọi hàm để bắt đầu "vẽ" các thành phần giao diện.
  + self.center\_window(): Gọi hàm để căn chỉnh cửa sổ ra giữa màn hình.

**Phần 3: Các Hàm Xử Lý**

Đây là nơi chứa toàn bộ "trí thông minh" của chương trình, thực hiện các công việc cụ thể.

**3.1. Các hàm dựng giao diện create\_widgets (Dòng 46-136)**

Python

def create\_widgets(self):

# ... tạo các Frame, Label, Entry, Button, Combobox ...

self.group\_combo.bind("<<ComboboxSelected>>", self.on\_group\_select)

# ...

* **Mục đích:** Xây dựng toàn bộ giao diện người dùng một cách có tổ chức.
* **Giải thích:**
  + Hàm này tạo ra các Frame (khung) để chia giao diện thành các khu vực logic (chọn tham số, khu vực bên A, bên B, kết quả).
  + Trong mỗi khung, nó tạo các Label (nhãn), Entry (ô nhập), Button (nút bấm), Combobox (hộp chọn).
  + Nó sử dụng .pack() và .grid() để sắp xếp vị trí của các thành phần này.
  + self.group\_combo.bind(...): Dòng này rất quan trọng, nó "lắng nghe" sự kiện khi người dùng chọn một mục trong Combobox và sẽ tự động gọi hàm on\_group\_select để xử lý.

**3.2. Các hàm tương tác và tính toán**

* **on\_group\_select(self, event) (Dòng 141-146):**
  + **Mục đích:** Tự động điền giá trị p và g vào các ô nhập liệu khi người dùng chọn một cặp có sẵn từ danh sách.
  + **Hoạt động:** Lấy tên lựa chọn từ Combobox, tìm cặp (p, g) tương ứng trong từ điển preset\_groups và cập nhật vào các ô Entry.
* **generate\_and\_set\_random\_key(self, party) (Dòng 149-163):**
  + **Mục đích:** Tạo một khóa bí mật ngẫu nhiên an toàn khi người dùng nhấn nút "Ngẫu nhiên".
  + **Hoạt động:** Lấy giá trị p hiện tại, sau đó dùng secrets.randbelow() để tạo một số ngẫu nhiên trong khoảng an toàn () rồi điền vào ô nhập a hoặc b.
* **validate\_inputs(self, p, g, a, b) (Dòng 173-199):**
  + **Mục đích:** Kiểm tra tất cả các giá trị đầu vào trước khi tính toán để đảm bảo chúng hợp lệ.
  + **Hoạt động:** Lần lượt kiểm tra p có phải số nguyên tố, g có nằm trong khoảng hợp lệ, a và b có nằm trong khoảng an toàn không. Nếu có lỗi, nó sẽ hiển thị một hộp thoại cảnh báo (messagebox.askyesno) và hỏi người dùng có muốn tiếp tục hay không.
* **calculate\_shared\_key(self) (Dòng 201-224):**
  + **Mục đích:** Đây là hàm xử lý chính, thực hiện toàn bộ thuật toán Diffie-Hellman.
  + **Hoạt động:**
    1. Lấy tất cả giá trị p, g, a, b từ các ô nhập liệu.
    2. Gọi hàm validate\_inputs để kiểm tra. Nếu không hợp lệ, hàm sẽ dừng lại.
    3. Sử dụng hàm pow(base, exp, mod) để tính toán các khóa công khai A, B và khóa chung K.
    4. Gọi hàm \_update\_readonly\_entry để hiển thị các kết quả A, B, K lên giao diện.
    5. Hiển thị hộp thoại thông báo thành công.

**3.3. Các hàm trợ giúp**

* **is\_prime(self, n) (Dòng 165-171):** Một hàm toán học đơn giản để kiểm tra tính nguyên tố của một số.
* **\_show\_validation\_warning(self, title, message) (Dòng 173-176):** Gom code hiển thị hộp thoại cảnh báo vào một chỗ để tránh lặp lại code.
* **\_update\_readonly\_entry(self, entry\_widget, text) (Dòng 226-231):** Một hàm tiện ích để cập nhật văn bản cho các ô Entry ở trạng thái chỉ đọc.
* **clear\_fields(self) (Dòng 233-241):** Xóa tất cả các ô nhập liệu và kết quả để người dùng có thể bắt đầu một phiên làm việc mới.
* **center\_window(self) (Dòng 243-251):** Tự động tính toán kích thước cửa sổ cần thiết và đặt nó ra giữa màn hình.

**Phần 4: Điểm khởi chạy (Dòng 253-258)**

Python

# --- 3. ĐIỂM KHỞI CHẠY CỦA CHƯƠG TRÌNH ---

if \_\_name\_\_ == "\_\_main\_\_":

root = ttk.Window(themename="litera")

app = DiffieHellmanApp(root)

root.mainloop()

* **Mục đích:** Khởi động chương trình.
* **Giải thích:**
  + if \_\_name\_\_ == "\_\_main\_\_":: Đảm bảo đoạn code này chỉ chạy khi file được thực thi trực tiếp.
  + root = ttk.Window(...): Tạo cửa sổ chính với theme "litera".
  + app = DiffieHellmanApp(root): Tạo đối tượng ứng dụng từ lớp DiffieHellmanApp, kích hoạt hàm \_\_init\_\_ để bắt đầu xây dựng mọi thứ.
  + root.mainloop(): Bắt đầu vòng lặp vô tận của cửa sổ, chờ đợi và xử lý các hành động của người dùng (nhấp chuột, gõ phím...).