2274802010504 : Trần Ngọc Long

Lí thuyết : Python

# 1. Nhập thư viện cần thiết

```
import tkinter as tk from tkinter import ttk
from tkinter import messagebox
```

**tkinter:** Thư viện chính để xây dựng giao diện người dùng (GUI) trong Python.

**ttk:** Thư viện con của tkinter cung cấp các widget (thành phần giao diện) đẹp hơn và hiện đại hơn.

**messagebox:** Thư viện cung cấp các hộp thoại thông báo cho người dùng

# 2. Khởi tạo cửa sổ chính

```
win = tk.Tk() win.title("Giao
diện tính toán")
 win.geometry("230x70")
win.resizable(False,
False)
```

**tk.Tk():** Tạo một đối tượng cửa sổ chính. **title():**

Thiết lập tiêu đề cho cửa sổ. **geometry():** Đặt kích

thước cửa sổ là 230x70 pixel.

**resizable():** Đặt giá trị False cho cả chiều rộng và chiều cao để người dùng không thể thay đổi kích thước cửa sổ.

## 3. Tạo nhãn và ô nhập cho các giá trị a và b

```
# Các nhãn & ô nhập #
Ô nhập số A
ttk.Label(win, text="Số a: ").grid(column=0, row=0,
sticky='w') so_a = tk.StringVar() so_a_entered =
ttk.Entry(win, width=12, textvariable=so_a)
so_a_entered.grid(column=1, row=0)

# Ô nhập số B ttk.Label(win, text="Số b: ").grid(column=0,
row=1, sticky='w') so_b = tk.StringVar()
so_b_entered = ttk.Entry(win, width=12,
textvariable=so_b) so_b_entered.grid(column=1, row=1)
```

**ttk.Label:** Tạo nhãn với văn bản "Số a". **grid():** Đặt nhãn vào vị trí cột 0, hàng 0 trong lưới, với sticky='w' để căn trái. **tk.StringVar():** Tạo biến để lưu giá trị nhập vào từ ô nhập. **ttk.Entry:** Tạo ô nhập với chiều rộng là 12 ký tự. **textvariable=so_a:** Liên kết ô nhập với biến so_a.

## 4. Tạo nhãn hiển thị kết quả

```
# Hiển thị kết quả ket_qua_label = ttk.Label(win, text="Kết
quả: ")
ket_qua_label.grid(column=0, row=2, sticky='w', columnspan=2)
```

Tạo nhãn để hiển thị kết quả và đặt nó ở hàng 2, chiếm 2 cột (columnspan=2).

## 5. Hàm kiểm tra xem giá trị có phải là số hay không

```
# Hàm kiểm tra kí tự input có phải là số
không def is_number(gia_tri): try:
float(gia_tri)
return True
except
ValueError:
return False
```

Hàm này nhận vào một chuỗi (gia_tri) và cố gắng chuyển đổi nó thành số thực (float).

Nếu thành công, trả về True; nếu không, trả về False. Điều này giúp đảm bảo người dùng chỉ nhập giá trị số.

## 6. Các hàm phép tính và button

```
# Các hàm phép tính
# Phép cộng def
cong():
if (is_number(so_a.get()) and is_number(so_b.get())):
ket_qua_label.configure(text="Kết quả: " + str((float(so_a.get())
float(so_b.get()))))) else:
                                                                +
messagebox.showerror("Lỗi nhập liệu", "Vui lòng chỉ nhập giá trị số")
 button_cong = ttk.Button(win, text="+", width=3,
command=cong) button_cong.grid(column=2, row=0)

# Phép trừ def tru(): if (is_number(so_a.get()) and
is_number(so_b.get())):
ket_qua_label.configure(text="Kết quả: " + str((float(so_a.get())


                                                                -
float(so_b.get()))))) else:
messagebox.showerror("Lỗi nhập liệu", "Vui lòng chỉ nhập giá trị số")
 button_tru = ttk.Button(win, text="-", width=3,
command=tru) button_tru.grid(column=3, row=0)

# Phép nhân def nhan(): if (is_number(so_a.get())
and is_number(so_b.get())):
ket_qua_label.configure(text="Kết quả: " + str((float(so_a.get())



                                                                *
```

```
float(so_b.get())))) else:
messagebox.showerror("Lỗi nhập liệu", "Vui lòng chỉ nhập giá trị số")
 button_nhan = ttk.Button(win, text="*", width=3,
command=nhan) button_nhan.grid(column=2, row=1)

# Phép chia def chia(): if (is_number(so_a.get())
and is_number(so_b.get())):
ket_qua_label.configure(text="Kết quả: " + str((float(so_a.get())
float(so_b.get())))) else:
messagebox.showerror("Lỗi nhập liệu", "Vui lòng chỉ nhập giá trị số")
 button_chia = ttk.Button(win, text="/", width=3,
command=chia) button_chia.grid(column=3, row=1)
```

Mỗi phép toán đều có các nút bấm cho phép cộng, trừ, nhân, chia, và liên kết chúng với các hàm tương ứng đã định nghĩa và một hàm riêng, kiểm tra giá trị đầu vào và hiển thị kết quả.

Lấy giá trị từ so_a và so_b, kiểm tra xem chúng có phải là số không. Nếu đúng, tính toán cộng trừ nhân chai và cập nhật nhãn kết quả; nếu không, hiển thị thông báo lỗi.

## 7. Cấu hình cột để tránh ô nhập bị đẩy

```
# Ghi chú để tránh ô nhập bị đẩy win.grid_columnconfigure(0,
weight=1) win.grid_columnconfigure(1, weight=1)
win.grid_columnconfigure(2, weight=0)
win.grid_columnconfigure(3, weight=0)
```

Cấu hình cột trong lưới:

- Cột 0 và 1 (có chứa nhãn và ô nhập) được thiết lập với weight=1, cho phép chúng tự động chiếm không gian có sẵn.
- Cột 2 và 3 (chứa nút bấm) được thiết lập với weight=0, giúp giữ cho chúng ở kích thước cố định mà không ảnh hưởng đến các cột khác.

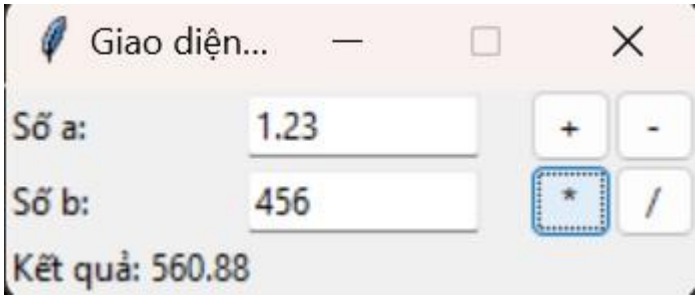
## 8. Chạy ứng dụng

```
win.mainloop()
```

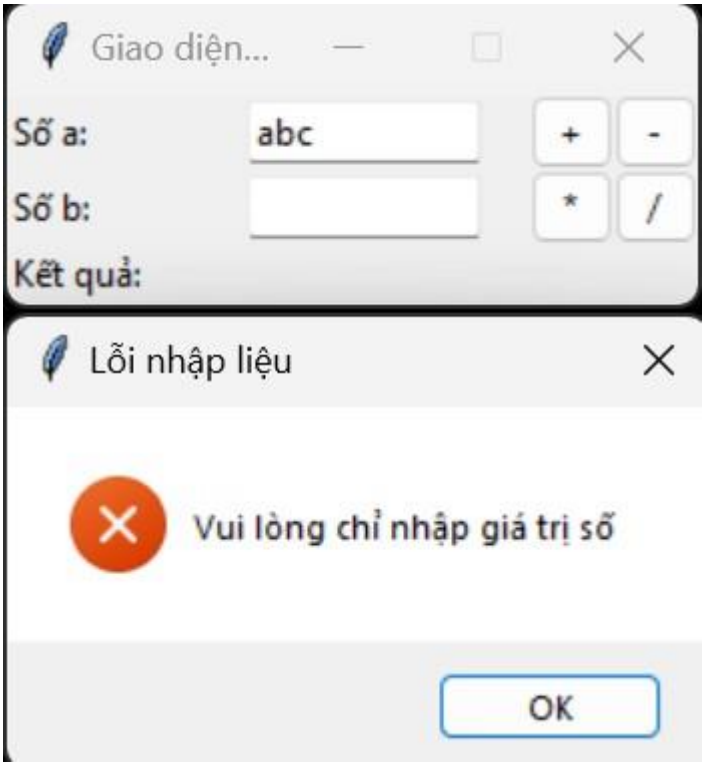
**mainloop():** Bắt đầu vòng lặp chính của ứng dụng, cho phép người dùng tương tác với giao diện cho đến khi đóng cửa sổ.

# Giao diện

**Tính toán bình thường:**




**Nhập sai định dạng:**



# Chức năng

Cộng trừ nhân chia như một chiếc máy tính bình thường, có thể tính toán các giá trị thập phân và sẽ báo lỗi nếu nhập input sai định dạng