

**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP**

**KHOA ĐIỆN TỬ**

**Bộ môn: Công nghệ Thông tin**



# **BÀI TẬP KẾT THÚC MÔN HỌC**

**MÔN HỌC**

**LẬP TRÌNH PYTHON**

**Sinh viên : Nguyễn Như Khiêm**

**Lớp : K58KTP.K01**

**Giáo viên giảng dạy : TS. Nguyễn Văn Huy**

**Link GitHub**

**:**



**Thái Nguyên – 2025**

## **BÀI TẬP KẾT THÚC MÔN HỌC**

### **MÔN HỌC: LẬP TRÌNH PYTHON** **BỘ MÔN : CÔNG NGHỆ THÔNG TIN**

*Sinh viên:* Nguyễn Như Khiêm

*MSV :* K225480106030

*Lớp:* K58KTP.K01

*Ngành:* Kỹ thuật máy tính

*Giáo viên hướng dẫn:* TS.Nguyễn Văn Huy

*Ngày giao đề:* 20/05/2025

*Ngày hoàn thành :* 07/06/2025

*Tên đề tài :* Máy tính đơn giản (Simple Calculator GUI)

*Yêu cầu :*

- + Nhập số, kiểm tra lỗi (không phải số, chia 0).
- + Cập nhật kết quả ngay khi nhấn nút “Tính”.
- + Cho phép reset (xóa cả 2 ô nhập).
- + Bắt ngoại lệ với hộp thoại thông báo khi lỗi .

**GIÁO VIÊN HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

## NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày....tháng.....năm 2025

GIÁO VIÊN HƯỚNG DẪN

*(Ký ghi rõ họ tên)*

# MỤC LỤC

<b>LỜI CẢM ƠN .....</b>	<b>5</b>
<b>DANH MỤC HÌNH ẢNH .....</b>	<b>6</b>
<b>LỜI NÓI ĐẦU .....</b>	<b>7</b>
<b>CHƯƠNG 1. GIỚI THIỆU ĐẦU BÀI .....</b>	<b>8</b>
1.1. Mô tả đề bài .....	8
1.2. Tính năng .....	8
1.3. Thách thức .....	8
1.4. Kiến thức áp dụng: .....	8
<b>CHƯƠNG 2 CƠ SỞ LÝ THUYẾT .....</b>	<b>10</b>
2.1. Tkinter .....	10
2.2. Xử lý ngoại lệ .....	10
2.3. Lập trình hướng đối tượng .....	10
2.4. Bố cục Grid .....	11
<b>CHƯƠNG 3. THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH .....</b>	<b>12</b>
3.1. Sơ đồ khối hệ thống .....	12
3.2. Sơ đồ khối các thuật toán chính .....	13
3.3. Cấu trúc dữ liệu .....	18
3.4. Chương trình .....	19
<b>CHƯƠNG 4 THỰC NGHIỆM VÀ KẾT LUẬN .....</b>	<b>24</b>
4.1. Thực nghiệm .....	24
4.2. Kết luận .....	26
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>28</b>

# LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành đến thầy cô và các bạn đã hỗ trợ tôi trong quá trình thực hiện dự án này. Đặc biệt, tôi xin cảm ơn thầy cô hướng dẫn đã cung cấp đề bài, tài liệu tham khảo, và những góp ý quý báu để tôi hoàn thiện ứng dụng Máy Tính Đơn Giản. Sự hướng dẫn tận tình của thầy cô đã giúp tôi hiểu rõ hơn về lập trình giao diện người dùng, xử lý lỗi, và tổ chức mã nguồn theo hướng đối tượng.

Em cũng xin cảm ơn các bạn học đã cùng thảo luận, chia sẻ kinh nghiệm lập trình Python và sử dụng thư viện Tkinter, giúp tôi vượt qua những khó khăn khi xây dựng giao diện và xử lý các lỗi phát sinh. Ngoài ra, tôi biết ơn gia đình và bạn bè đã động viên, tạo điều kiện để tôi tập trung hoàn thành dự án này.

Dự án này là một bước tiến quan trọng trong hành trình học tập của tôi, không chỉ giúp tôi củng cố kiến thức lập trình mà còn rèn luyện kỹ năng tư duy logic và giải quyết vấn đề. Một lần nữa, tôi xin gửi lời tri ân sâu sắc đến tất cả những ai đã đồng hành và hỗ trợ tôi trong quá trình thực hiện dự án.

## DANH MỤC HÌNH ẢNH

Hình 1: Biểu đồ phân cấp chức năng .....	12
Hình 2 : Sơ đồ khối của thuật toán xử lý tính toán trung tâm .....	14
Hình 3 : Sơ đồ khối của thuật toán nhập số và kiểm tra lỗi .....	15
Hình 4 : Sơ đồ khối của thuật toán cập nhật kết quả khi nhấn “Tính” .....	16
Hình 5 : Sơ đồ khối của thuật toán Reset .....	17
Hình 6 : Sơ đồ khối của thuật toán bắt ngoại lệ và hiển thị hộp thoại .....	18
Hình 7 : Kết quả thực hiện phép cộng .....	24
Hình 8 : Kết quả thực hiện phép nhân .....	24
Hình 9 : Kết quả thực hiện phép cộng .....	25
Hình 10 : Kết quả thực hiện phép cộng .....	25
Hình 11 : Kết quả thực hiện phép cộng .....	26

# LỜI NÓI ĐẦU

Ứng dụng Máy Tính Đơn Giản là một dự án lập trình nhằm mục đích thực hành các kỹ năng phát triển giao diện người dùng đồ họa (GUI) bằng Python và thư viện Tkinter. Dự án này được thực hiện trong khuôn khổ bài tập học phần lập trình Python, với mục tiêu xây dựng một ứng dụng cho phép người dùng thực hiện các phép toán cơ bản (cộng, trừ, nhân, chia) trên hai số thực, đồng thời xử lý các lỗi nhập liệu và cung cấp giao diện thân thiện.

Mục đích của bài tập này là trình bày quá trình thiết kế, xây dựng, và kiểm thử ứng dụng, từ cơ sở lý thuyết đến thực nghiệm thực tế. Bài tập kết thúc môn học sẽ làm rõ các tính năng, thách thức, và kiến thức được áp dụng trong dự án, đồng thời cung cấp các sơ đồ khối, cấu trúc dữ liệu, và kết quả thực nghiệm để minh họa cách hoạt động của chương trình. Qua bài tập này, em không chỉ củng cố kiến thức về lập trình Python mà còn học được cách tổ chức mã nguồn, xử lý lỗi, và tối ưu hóa giao diện người dùng.

Bài tập kết thúc môn học được chia thành bốn chương chính: Giới thiệu đầu bài, Cơ sở lý thuyết, Thiết kế và xây dựng chương trình, Thực nghiệm và kết luận. Hy vọng bài tập này sẽ cung cấp cái nhìn toàn diện về quá trình phát triển ứng dụng và những bài học em đã rút ra.

# CHƯƠNG 1. GIỚI THIỆU ĐẦU BÀI

## 1.1. Mô tả đề bài

Viết chương trình máy tính GUI cho phép nhập hai số thực, chọn phép toán (+, -, ×, ÷), hiển thị kết quả, kiểm tra lỗi (nhập không phải số, chia cho 0), và có nút reset.

## 1.2. Tính năng

- + Nhập liệu số thực: Người dùng nhập hai số thực thông qua hai ô nhập liệu.
- + Lựa chọn phép toán: Hỗ trợ bốn phép toán (+, -, ×, ÷) thông qua các nút radio được sắp xếp theo bố cục lưới 2x2.
- + Hiển thị kết quả: Kết quả phép tính được hiển thị trên nhãn với định dạng số thực làm tròn.
- + Xử lý lỗi: Phát hiện và thông báo lỗi khi người dùng nhập dữ liệu không hợp lệ hoặc thực hiện phép chia cho 0.
- + Chức năng reset: Xóa toàn bộ dữ liệu nhập, đặt lại phép toán mặc định, và xóa kết quả hiển thị.
- + Giao diện thân thiện: Giao diện được thiết kế với màu sắc bắt mắt, font chữ rõ ràng, và hỗ trợ hiển thị tốt trên màn hình độ phân giải cao.

## 1.3. Thách thức

- + **Xây dựng giao diện trực quan:** Đảm bảo các thành phần giao diện (ô nhập liệu, nút radio, nút chức năng) được sắp xếp hợp lý, dễ sử dụng.
- + **Xử lý lỗi nhập liệu:** Cần phát hiện và xử lý các trường hợp như nhập chữ thay vì số hoặc chia cho 0, đồng thời hiển thị thông báo lỗi thân thiện.
- + **Tối ưu hóa giao diện:** Đảm bảo giao diện hiển thị tốt trên các màn hình có độ phân giải cao bằng cách hỗ trợ DPI cao, tránh hiện tượng mờ hoặc sai tỷ lệ.
- + **Tổ chức mã nguồn:** Sử dụng lập trình hướng đối tượng để mã nguồn dễ bảo trì, mở rộng và tái sử dụng.

## 1.4. Kiến thức áp dụng:

- **Thư viện Tkinter:** Sử dụng các widget như Entry, Label, Button, Radiobutton, Frame để tạo giao diện đồ họa.



- **Lập trình hướng đối tượng (OOP):** Tổ chức mã nguồn thành lớp (MayTinhDonGian) để quản lý các thành phần giao diện và chức năng.
- **Xử lý ngoại lệ:** Sử dụng try-except để xử lý lỗi như ValueError (nhập sai định dạng) và ZeroDivisionError (chia cho 0).
- **Tùy chỉnh giao diện:** Áp dụng style (ttk) để tùy chỉnh font, màu sắc, và bố cục, tạo trải nghiệm người dùng tốt hơn.
- **Hỗ trợ DPI cao:** Sử dụng ctypes để đảm bảo giao diện hiển thị sắc nét trên màn hình có độ phân giải cao.

Ứng dụng này là một bài tập thực hành lý tưởng để làm quen với lập trình GUI, xử lý lỗi, và tổ chức mã nguồn theo hướng đối tượng, đồng thời cung cấp nền tảng để mở rộng thêm các tính năng nâng cao trong tương lai.

*Tóm tắt Chương 1 – Giới thiệu đầu bài: Chương trình được xây dựng là một máy tính đơn giản với giao diện GUI, cho phép người dùng nhập hai số thực, chọn phép toán (+, −, ×, ÷), tính toán và hiển thị kết quả. Ứng dụng hỗ trợ xử lý lỗi khi nhập sai dữ liệu hoặc chia cho 0, đồng thời cung cấp nút Reset để xóa dữ liệu và khôi phục trạng thái ban đầu. Giao diện được thiết kế thân thiện, hỗ trợ hiển thị tốt trên màn hình độ phân giải cao. Trong quá trình phát triển, người thực hiện đối mặt với các thách thức như sắp xếp giao diện trực quan, xử lý ngoại lệ hiệu quả, và tổ chức mã nguồn theo hướng đối tượng. Chương trình sử dụng các kiến thức về Tkinter, lập trình hướng đối tượng, xử lý ngoại lệ, và tùy chỉnh giao diện. Đây là một bài tập thực hành thiết thực giúp làm quen với lập trình GUI Python và tạo nền tảng cho các ứng dụng nâng cao sau này.*

## CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

### 2.1. Tkinter

Tkinter là thư viện chuẩn của Python để xây dựng giao diện người dùng đồ họa, cung cấp các widget cơ bản như Label (nhãn), Entry (ô nhập liệu), Button (nút), Radiobutton (nút radio), và Frame (khung chứa). Tkinter hỗ trợ ba phương pháp quản lý bố cục: pack, grid, và place. Trong chương trình này, phương pháp grid được sử dụng để sắp xếp các widget theo hàng và cột, đảm bảo bố cục rõ ràng và dễ điều chỉnh.

- **Widget:** Mỗi widget có các thuộc tính như font, màu sắc, kích thước, và hành vi được cấu hình thông qua tham số hoặc phương thức. Ví dụ, ô nhập liệu (Entry) cho phép người dùng nhập dữ liệu, trong khi nhãn (Label) hiển thị văn bản tĩnh hoặc động.
- **ttk:** Là phiên bản cải tiến của Tkinter, cung cấp các widget có giao diện hiện đại hơn và hỗ trợ tùy chỉnh style (ví dụ: font, màu nền, padding). Trong chương trình, ttk.Style được sử dụng để định dạng các widget như TLabel, TButton, và TRadiobutton.
- **Quản lý sự kiện:** Các widget như Button và Radiobutton được liên kết với các hàm xử lý (command) để thực hiện hành động khi người dùng tương tác.

### 2.2. Xử lý ngoại lệ

Sử dụng try-except để bắt lỗi ValueError và ZeroDivisionError.

Chương trình sử dụng cơ chế xử lý ngoại lệ để đảm bảo tính mạnh mẽ:

- **ValueError:** Xảy ra khi người dùng nhập dữ liệu không phải số (ví dụ: chữ hoặc ký tự đặc biệt). Chương trình hiển thị thông báo lỗi qua messagebox.showerror và reset giao diện.
- **ZeroDivisionError:** Xảy ra khi người dùng cố chia một số cho 0. Chương trình kiểm tra và thông báo lỗi cụ thể.
- **Messagebox:** Hộp thoại của Tkinter được sử dụng để hiển thị thông báo lỗi một cách trực quan, giúp người dùng hiểu rõ vấn đề.

### 2.3. Lập trình hướng đối tượng

Cách tổ chức mã trong một lớp (SimpleCalculator).

Lập trình hướng đối tượng được áp dụng để tổ chức mã nguồn một cách, dễ bảo trì và mở rộng. Lớp `MayTinhDonGian` đóng vai trò là trung tâm quản lý toàn bộ ứng dụng, chứa các thuộc tính (như widget) và phương thức (như xử lý phép tính, reset). Các khái niệm OOP được sử dụng bao gồm:

- **Encapsulation:** Các thuộc tính như `o_nhap_so_1`, `o_nhap_so_2`, và `phép_toan_duoc_chon` được đóng gói trong lớp để quản lý trạng thái giao diện.
- **Modularity:** Mỗi chức năng (tạo ô nhập liệu, tạo nút radio, xử lý phép tính) được tách thành các phương thức riêng, giúp mã nguồn dễ đọc và bảo trì.
- **Object instantiation:** Một đối tượng của lớp `MayTinhDonGian` được tạo trong hàm main để khởi chạy ứng dụng.

## 2.4. Bố cục Grid

Phương pháp bố cục grid của Tkinter được sử dụng để sắp xếp widget theo lưới hàng-cột. Mỗi widget được đặt vào một ô cụ thể (row, column) với các tham số như `padx`, `pady` để tạo khoảng cách. Trong chương trình, ô nhập liệu, nhãn, và nút radio được sắp xếp trong khung chính (`ttk.Frame`) theo bố cục lưới, đảm bảo giao diện gọn gàng và dễ mở rộng. Ví dụ, các nút radio phép toán được sắp xếp trong lưới 2x2, giúp người dùng dễ dàng lựa chọn.

*Tóm tắt Chương 2 – Cơ sở lý thuyết: Chương trình sử dụng thư viện Tkinter để xây dựng giao diện người dùng, trong đó các widget như Entry, Label, Button và Radiobutton được bố trí bằng phương pháp grid giúp sắp xếp rõ ràng theo hàng – cột. Phiên bản ttk được sử dụng để cải tiến giao diện với style đẹp và hiện đại. Việc xử lý lỗi được thực hiện thông qua try-except, bao gồm ValueError khi nhập sai định dạng và ZeroDivisionError khi chia cho 0, đi kèm hộp thoại messagebox để thông báo cho người dùng. Chương trình được tổ chức theo hướng đối tượng với lớp MayTinhDonGian, nơi quản lý toàn bộ giao diện và chức năng, áp dụng các nguyên lý như đóng gói, mô-đun hóa và khởi tạo đối tượng. Bố cục lưới (grid) không chỉ giúp dễ quản lý mà còn đảm bảo tính trực quan và mở rộng linh hoạt cho giao diện.*

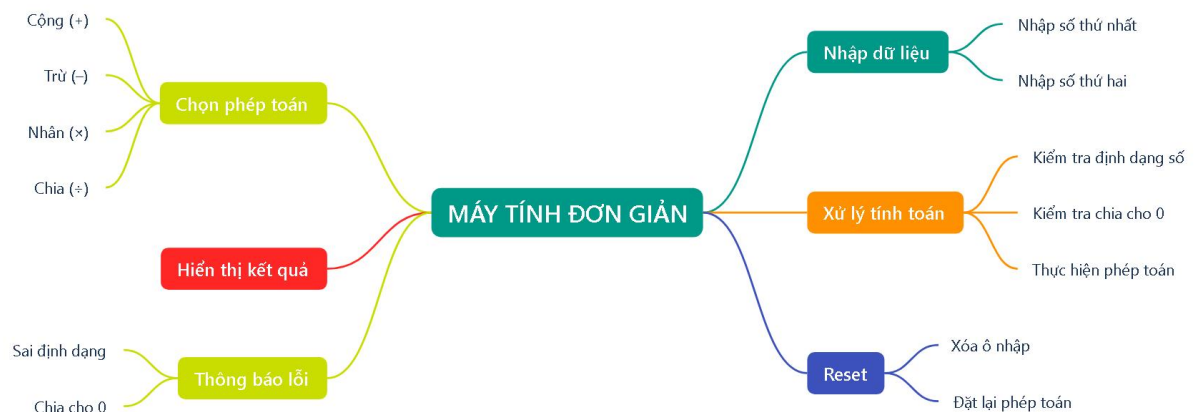
## CHƯƠNG 3. THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

### 3.1. Sơ đồ khối hệ thống

❖ Chương trình được tổ chức thành một lớp `MayTinhDonGian` với các module chính:

1. **Khởi tạo giao diện:** Thiết lập cửa sổ chính (450x500, không thay đổi kích thước), màu nền `#e0f7fa`, và khung chính (`ttk.Frame`).
2. **Tạo widget:** Tạo ô nhập liệu, nút radio, nút chức năng (Tính, Reset), và nhãn kết quả.
3. **Xử lý phép toán:** Lấy dữ liệu, thực hiện phép tính, và hiển thị kết quả.
4. **Xử lý lỗi:** Phát hiện lỗi nhập liệu và chia cho 0, hiển thị thông báo.
5. **Reset:** Xóa dữ liệu và đặt lại trạng thái ban đầu.

❖ Biểu đồ phân cấp chức năng



Hình 1: Biểu đồ phân cấp chức năng

**Mô tả biểu đồ:** Biểu đồ là một cây phân cấp với nút gốc "Máy Tính Đơn Giản". Mỗi nhánh con là một chức năng, chia nhỏ thành các tác vụ cụ thể.

**Các chức năng chính được liệt kê là:**

1. **Nhập dữ liệu:** Bao gồm việc nhập số thứ nhất và nhập số thứ hai. Đáng chú ý là nguồn cũng liệt kê các phép toán Cộng, Trừ, Nhân, Chia dưới nhánh "Nhập dữ liệu" này.

2. **Xử lý tính toán:** Nhánh này chi tiết hóa các bước xử lý, bao gồm kiểm tra định dạng số, kiểm tra chia cho 0, và thực hiện phép toán.
3. **Hiển thị kết quả**
4. **Reset:** Chức năng này bao gồm các hoạt động như xóa ô nhập1 và đặt lại phép toán
5. **Thông báo lỗi:** Nhánh này xử lý các trường hợp lỗi, cụ thể là sai định dạng hoặc chia cho 0.

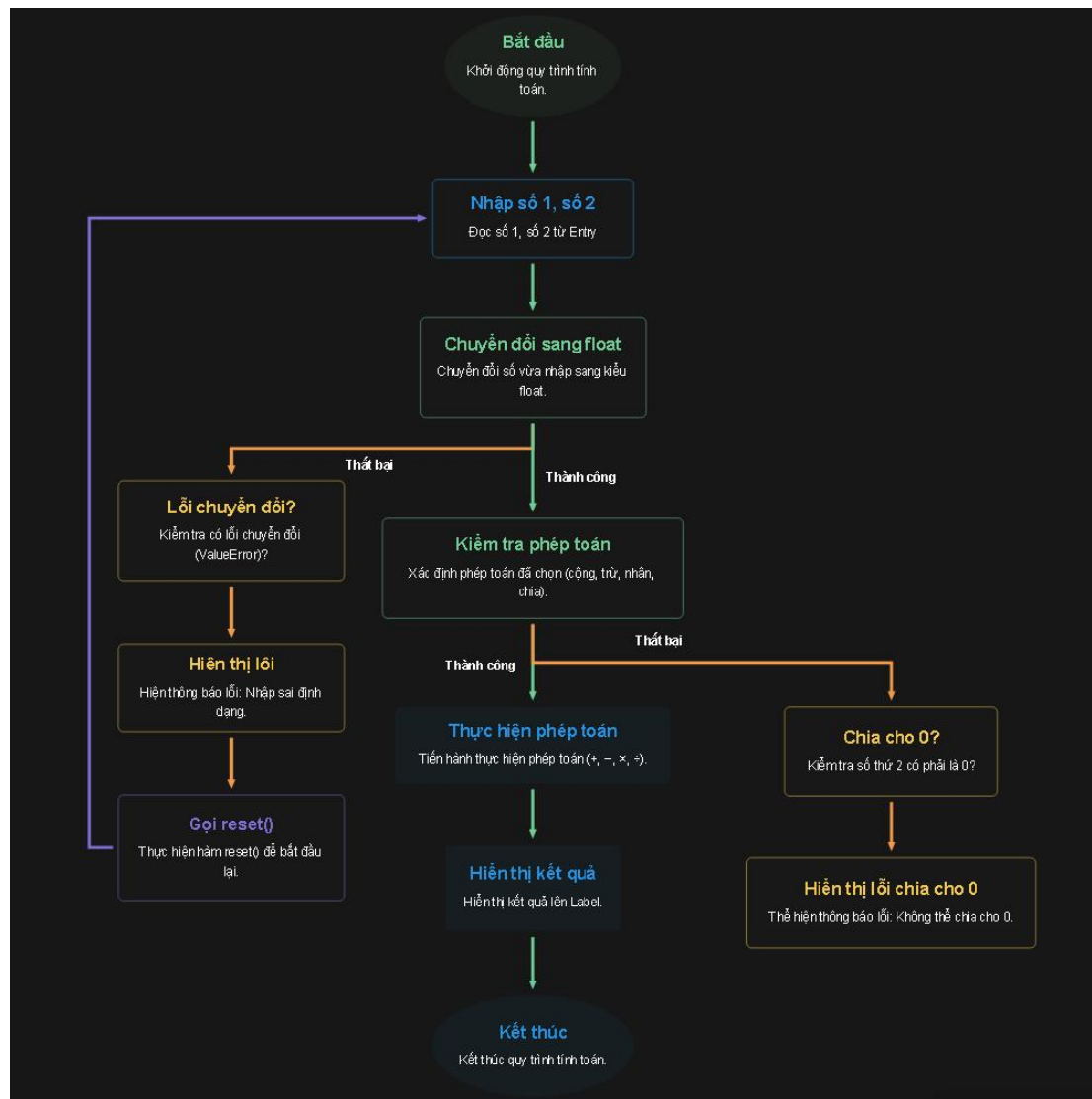
Biểu đồ phân cấp chức năng này giúp hình dung rõ ràng cách các chức năng lớn của máy tính đơn giản được chia nhỏ thành các chức năng con cụ thể hơn

### 3.2. Sơ đồ khối các thuật toán chính

#### ❖ Thuật toán xử lý tính toán (trung tâm) :

- **Chức năng:** Điều phối toàn bộ quá trình tính toán và xử lý lỗi.
- **Đầu vào:** Hai chuỗi số từ Entry, phép toán từ Radio.
- **Xử lý:**
  - + Chuyển chuỗi  $\rightarrow$  float
  - + Nếu lỗi  $\rightarrow$  gọi thuật toán xử lý lỗi
  - + Nếu phép toán là chia và số 2 = 0  $\rightarrow$  lỗi
  - + Ngược lại thực hiện phép toán đã chọn
- **Đầu ra:** Kết quả số thực hoặc lỗi được xử lý

➤ Sơ đồ khối :



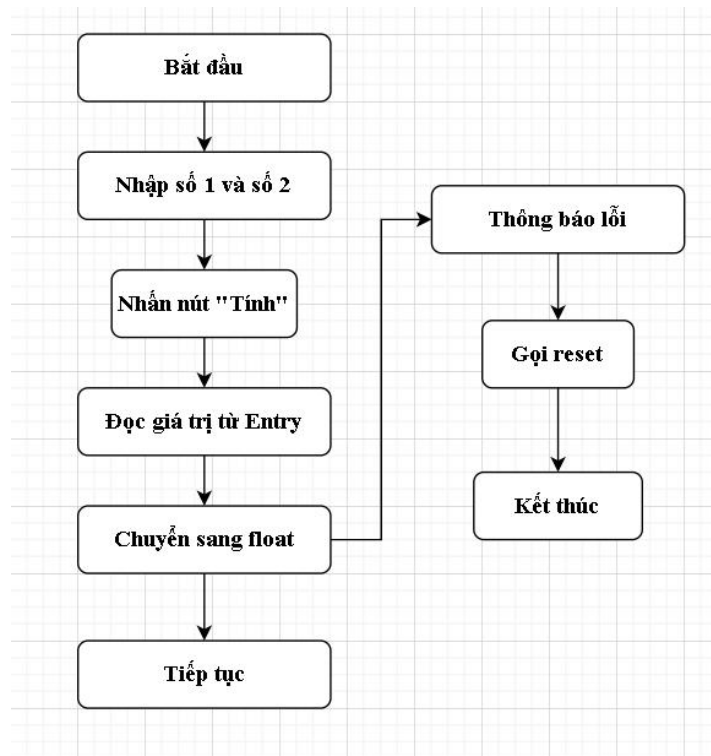
Hình 2 : Sơ đồ khối của thuật toán xử lý tính toán trung tâm

❖ Thuật toán nhập số và kiểm tra lỗi

- **Chức năng:** Cho phép người dùng nhập hai số vào hai ô Entry, đồng thời kiểm tra xem dữ liệu nhập có phải là số hợp lệ không.
- **Đầu vào:** Chuỗi ký tự từ hai ô Entry (số thứ nhất và số thứ hai).
- **Xử lý:**
  - + Đọc chuỗi từ Entry.
  - + Dùng float() để chuyển đổi chuỗi.
  - + Nếu chuyển đổi thất bại → phát sinh ValueError.

- **Đầu ra:**
  - + Nếu hợp lệ: tiếp tục xử lý.
  - + Nếu lỗi: hiện hộp thoại thông báo và gọi reset().

➤ **Sơ đồ khối**



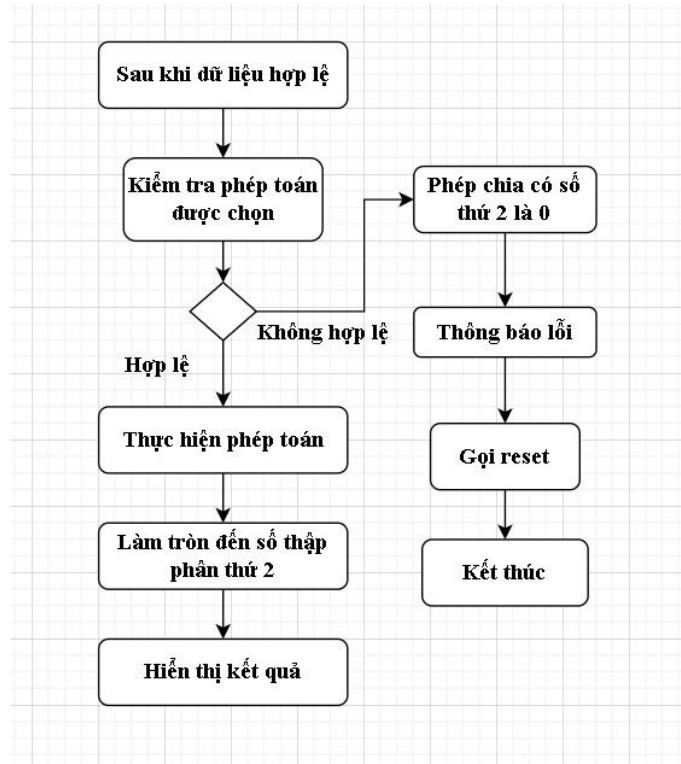
Hình 3 : Sơ đồ khối của thuật toán nhập số và kiểm tra lỗi

❖ **Thuật toán cập nhật kết quả khi nhấn “Tính”**

- **Chức năng:** Khi người dùng nhấn nút “Tính”, hệ thống thực hiện phép toán đã chọn và hiển thị kết quả.
- **Đầu vào:**
  - + Hai số thực (sau khi chuyển đổi).
  - + Phép toán từ Radiobutton (+, -, ×, ÷).
- **Xử lý:**
  - + Kiểm tra nếu phép toán là chia và số thứ hai bằng 0 → phát sinh ZeroDivisionError.
  - + Thực hiện phép toán theo đúng ký hiệu đã chọn.
  - + Làm tròn kết quả tới 2 chữ số thập phân.

- **Đầu ra:**
  - + Nếu thành công: kết quả hiển thị trong Label.
  - + Nếu lỗi: gọi xử lý lỗi → hiện hộp thoại và reset.

➤ **Sơ đồ khối**



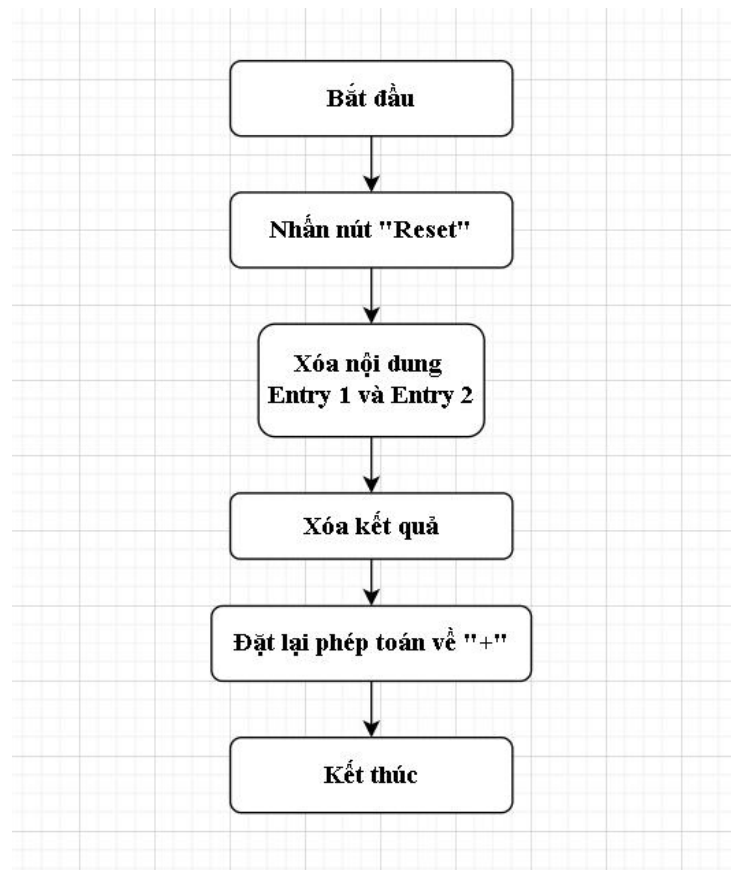
Hình 4 : Sơ đồ khối của thuật toán cập nhật kết quả khi nhấn “Tính”

❖ **Thuật toán Reset**

- **Chức năng:** Đặt lại toàn bộ giao diện về trạng thái ban đầu.
- **Đầu vào:** Sự kiện nhấn nút “Reset” hoặc sau khi xảy ra lỗi.
- **Xử lý:**
  - + Gọi `Entry.delete(0, END)` để xóa dữ liệu ở hai ô nhập.
  - + Đặt lại biến phép toán (`StringVar`) về dấu “+”.
  - + Cập nhật Label kết quả về giá trị mặc định.
- **Đầu ra:** Giao diện trống, sẵn sàng để nhập mới.



### ➤ Sơ đồ khối

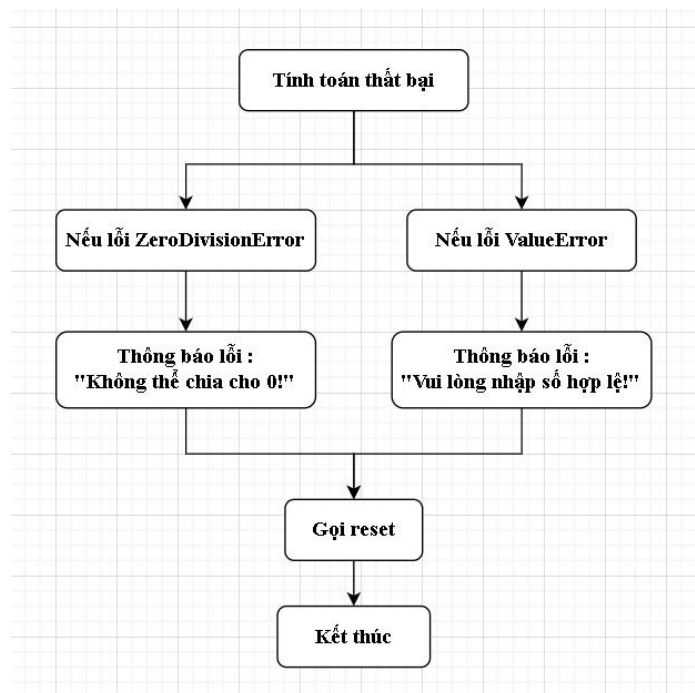


Hình 5 : Sơ đồ khối của thuật toán Reset

### ❖ Thuật toán bắt ngoại lệ và hiển thị hộp thoại

- **Chức năng:** Thông báo cho người dùng khi xảy ra lỗi nhập dữ liệu hoặc chia cho 0.
- **Đầu vào:** Các ngoại lệ được phát hiện trong khối try-except (ValueError, ZeroDivisionError).
- **Xử lý:**
  - + Nếu ValueError: dùng messagebox.showerror() để hiển thị thông báo “Vui lòng nhập số hợp lệ”.
  - + Nếu ZeroDivisionError: hiển thị “Không thể chia cho 0!”.
  - + Sau đó gọi reset() để quay về trạng thái ban đầu.
- **Đầu ra:** Hộp thoại lỗi và giao diện được làm mới.

### ➤ Sơ đồ khối



Hình 6 : Sơ đồ khối của thuật toán bắt ngoại lệ và hiển thị hộp thoại

### 3.3. Cấu trúc dữ liệu

Chương trình lưu trữ dữ liệu đơn giản, chủ yếu là trạng thái giao diện:

- **Thuộc tính lớp:**
  - `cua_so_chinh`: Cửa sổ chính (tk.Tk).
  - `khung_chinh`: Khung chính (ttk.Frame, màu #e0f7fa).
  - `o_nhap_so_1`, `o_nhap_so_2`: Ô nhập liệu (ttk.Entry).
  - `phép_toán_được_chọn`: Biến tk.StringVar lưu phép toán (+, -, ×, ÷).
  - `nhân_kết_quả`: Nhãn hiển thị kết quả (ttk.Label, màu #ffe0b2).

Trường	Kiểu dữ liệu	Mô tả
Số thứ nhất	Chuỗi → float	Nhập từ <code>o_nhap_so_1</code>
Số thứ hai	Chuỗi → float	Nhập từ <code>o_nhap_so_2</code>
Phép toán	StringVar	"+", "-", "×", "÷"
Kết quả	Chuỗi (Label)	Hiển thị trên <code>nhân_kết_quả</code>

### 3.4. Chương trình

#### ❖ Hàm trong lớp MayTinhDonGian:

1. `__init__`: Khởi tạo cửa sổ, khung chính, và gọi các phương thức tạo widget.
2. `thiet_lap_style`: Cấu hình style cho Label, Button, Radiobutton.
3. `tao_o_nhap_so_thu_nhat`, `tao_o_nhap_so_thu_hai`: Tạo ô nhập liệu và nhấn.
4. `tao_vung_chon_phep_toan`: Tạo nút radio phép toán (2x2).
5. `tao_cac_nut_chuc_nang`: Tạo nút Tính và Reset.
6. `tao_vung_hien_thi_ket_qua`: Tạo nhãn kết quả.
7. `thuc_hien_phep_tinh`: Thực hiện phép toán và xử lý lỗi.
8. `reset`: Xóa dữ liệu và đặt lại trạng thái.

#### ❖ Hàm ngoài lớp:

- + `main`: Tạo cửa sổ chính, khởi tạo ứng dụng, và chạy vòng lặp mainloop.

#### ❖ Code :

```
import tkinter as tk
from tkinter import messagebox, ttk
import ctypes
# Thiết lập hỗ trợ DPI cao cho màn hình độ phân giải cao
ctypes.windll.shcore.SetProcessDpiAwareness(1)

# Tạo lớp MayTinhDonGian để quản lý giao diện
class MayTinhDonGian:
    def __init__(self, cua_so_chinh):
        self.cua_so_chinh = cua_so_chinh
        self.cua_so_chinh.title("Máy Tính Đơn Giản")
        self.cua_so_chinh.geometry("450x500") # Kích thước cửa sổ
        self.cua_so_chinh.resizable(False, False) # Không cho phép thay đổi kích thước
        self.cua_so_chinh.configure(bg="#e0f7fa") # Màu nền xanh nhạt

        # Thiết lập style cho các thành phần giao diện
        self.thiet_lap_style()

        # Tạo khung chính chứa tất cả các thành phần
        self.khung_chinh = ttk.Frame(cua_so_chinh, padding=20, style="Main.TFrame")
        self.khung_chinh.grid(row=0, column=0, sticky="nsew")

        # Cấu hình grid để khung chính mở rộng theo cửa sổ
        self.cua_so_chinh.grid_rowconfigure(0, weight=1)
        self.cua_so_chinh.grid_columnconfigure(0, weight=1)

        # Tạo các thành phần giao diện
        self.tao_o_nhap_so_thu_nhat()
```

```

self.tao_o_nhập_so_thứ_hai()
self.tao_vùng_chọn_phep_toan()
self.tao_cac_nút_chức_năng()
self.tao_vùng_hien_thi_ket_qua()

# Thiết lập style cho các thành phần giao diện
def thiết_lap_style(self):
    style = ttk.Style()
    # Cấu hình style cho khung chính
    style.configure("TLabel", font=("Times New Roman", 14, "bold"), background="#e0f7fa") # Style
cho nhãn (Label)
    style.configure("TButton", font=("Times New Roman", 13, "bold"), padding=10) # Style cho nút
(Button)
    style.configure("TRadiobutton", font=("Times New Roman", 16, "bold"), background="#fdfefe") #
Style cho nút radio

def tao_o_nhập_so_thứ_nhất(self):
    # Nhãn "Số thứ nhất"
    self.nhan_so_1 = ttk.Label(self.khung_chinh, text="Số thứ nhất:", foreground="#e91e63")
    self.nhan_so_1.grid(row=0, column=0, padx=10, pady=10, sticky="w")
    # Ô nhập số thứ nhất
    self.o_nhập_so_1 = ttk.Entry(self.khung_chinh, font=("Times New Roman", 14), width=20)
    self.o_nhập_so_1.grid(row=0, column=1, padx=10, pady=10)

# Tạo ô nhập số thứ hai và nhãn tương ứng
def tao_o_nhập_so_thứ_hai(self):
    # Nhãn "Số thứ hai"
    self.nhan_so_2 = ttk.Label(self.khung_chinh, text="Số thứ hai:", foreground="#e91e63")
    self.nhan_so_2.grid(row=1, column=0, padx=10, pady=10, sticky="w")
    # Ô nhập số thứ hai
    self.o_nhập_so_2 = ttk.Entry(self.khung_chinh, font=("Times New Roman", 14), width=20)
    self.o_nhập_so_2.grid(row=1, column=1, padx=10, pady=10)

# Tạo vùng chọn phép toán với các nút radio
def tao_vùng_chọn_phep_toan(self):
    # Nhãn "Phép toán"
    self.nhan_phep_toan = ttk.Label(self.khung_chinh, text="Phép toán:", foreground="#000")
    self.nhan_phep_toan.grid(row=2, column=0, padx=10, pady=10, sticky="nw")

    # Khung chứa các nút radio phép toán
    self.khung_phep_toan = tk.Frame(self.khung_chinh, bg="#fdfefe")
    self.khung_phep_toan.grid(row=2, column=1, padx=10, pady=10)

    # Biến lưu phép toán được chọn (mặc định là cộng)
    self.phep_toan_duoc_chon = tk.StringVar(value="+")

    # Danh sách các phép toán và màu sắc tương ứng
    cac_phep_toan = [
        ("+", "#10e96a"),

```

```

("-", "#e32611"),
("×", "#ef1bba"),
("÷", "#cea80e")
]

```

# Tạo nút radio cho từng phép toán

for chi so, (ky hieu phep toan, mau sac) in enumerate(cac phep toan):

```

    nut_radio = tk.Radiobutton(
        self.khung_phep_toan,
        text=ky hieu phep toan,
        variable=self.phep toan duoc chon,
        value=ky_hieu_phep_toan,
        font=("Times New Roman", 25, "bold"),
        bg="#fdfefe",
        fg=mau sac,
        selectcolor="#fdfefe", # Màu nền khi chọn
        indicatoron=True
    )

```

# Sắp xếp nút radio theo dạng lưới 2x2

```

    nut_radio.grid(row=chi so // 2, column=chi so % 2, padx=15, pady=5)

```

# Tạo các nút chức năng: Tính và Reset

def tao cac nut chuc nang(self):

# Khung chứa các nút chức năng

```

self.khung cac nut = tk.Frame(self.khung chinh, bg="#e0f7fa")

```

```

self.khung cac nut.grid(row=3, column=0, columnspan=2, pady=20)

```

# Nút Tính - thực hiện phép tính

```

self.nut tinh = ttk.Button(self.khung cac nut, text="Tính", command=self.thuc hien phep tinh)

```

```

self.nut tinh.grid(row=0, column=0, padx=20)

```

# Nút Reset - xóa dữ liệu và reset về trạng thái ban đầu

```

self.nut_reset = ttk.Button(self.khung cac nut, text="Reset", command=self.reset)

```

```

self.nut_reset.grid(row=0, column=1, padx=20)

```

# Tạo vùng hiển thị kết quả phép tính

def tao\_vung\_hien\_thi\_ket\_qua(self):

```

    self.nhan_ket_qua = ttk.Label(
        self.khung_chinh,
        text="Kết quả: ",
        foreground="#000000",
        font=("Times New Roman", 14, "bold"),
        background="#ffe0b2"
    )

```

```

    self.nhan_ket_qua.grid(row=4, column=0, columnspan=2, pady=15, sticky="ew")

```

#Thực hiện phép tính dựa trên dữ liệu nhập vào và phép toán được chọn  
 Xử lý các lỗi có thể xảy ra (nhập sai định dạng, chia cho 0)

def thuc\_hien\_phep\_tinh(self):

```

try:
    # Lấy giá trị từ ô nhập và chuyển đổi sang số thực
    so_thu_nhat = float(self.o_nhap_so_1.get())
    so_thu_hai = float(self.o_nhap_so_2.get())
    phep_toan = self.phep_toan_duoc_chon.get()

    # Thực hiện phép tính dựa trên phép toán được chọn
    if phep_toan == "+":
        ket_qua = so_thu_nhat + so_thu_hai
    elif phep_toan == "-":
        ket_qua = so_thu_nhat - so_thu_hai
    elif phep_toan == "×":
        ket_qua = so_thu_nhat * so_thu_hai
    elif phep_toan == "÷":
        if so_thu_hai == 0: # Kiểm tra chia cho 0
            raise ZeroDivisionError("Không thể chia cho 0!")
        ket_qua = so_thu_nhat / so_thu_hai

    self.nhan_ket_qua.config(text=f"Kết quả: {ket_qua:.2f}") # Hiển thị kết quả (làm tròn 2 chữ số thập phân)

except ValueError:
    messagebox.showerror("Lỗi", "Vui lòng nhập số hợp lệ!", parent=self.cua_so_chinh)
    self.reset()
except ZeroDivisionError as loi_chia_0:
    messagebox.showerror("Lỗi", str(loi_chia_0), parent=self.cua_so_chinh)
    self.reset()

# Phương thức reset - xóa dữ liệu và đặt lại trạng thái ban đầu
def reset(self):
    self.o_nhap_so_1.delete(0, tk.END)
    self.o_nhap_so_2.delete(0, tk.END)
    self.phep_toan_duoc_chon.set("+")
    self.nhan_ket_qua.config(text="Kết quả: ")

# Chương trình chính - tạo cửa sổ và khởi chạy ứng dụng máy tính
def main():
    cua_so_chinh = tk.Tk() # Tạo cửa sổ chính
    style = ttk.Style() # Cấu hình style cho khung chính
    style.configure("Main.TFrame", background="#e0f7fa")
    ung_dung_may_tinh = MayTinhDonGian(cua_so_chinh) # Khởi tạo ứng dụng máy tính
    cua_so_chinh.mainloop() # Chạy vòng lặp chính của giao diện

# Điểm khởi đầu của chương trình
if __name__ == "__main__":
    main()

```

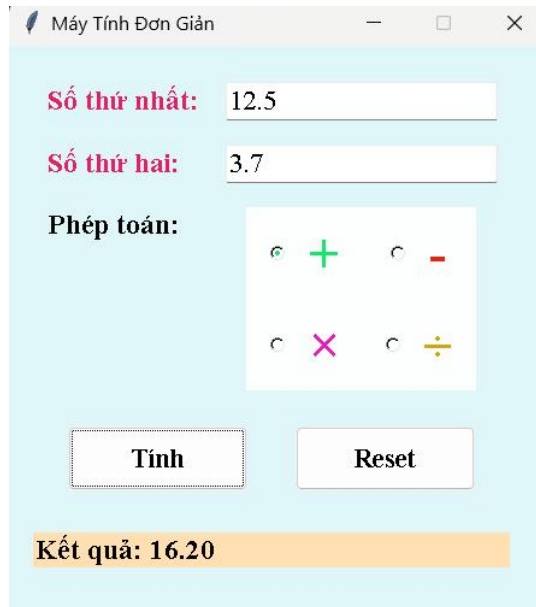
*Tóm tắt Chương 3 – Thiết kế và xây dựng chương trình: Chương trình được thiết kế theo hướng đối tượng với lớp `MayTinhDonGian` là trung tâm điều khiển toàn bộ giao diện và chức năng. Giao diện gồm các thành phần chính: ô nhập liệu, nhóm nút chọn phép toán, nút “Tính” và “Reset”, cùng nhãn hiển thị kết quả. Mỗi chức năng được chia thành các thuật toán riêng như: nhập số và kiểm tra lỗi, xử lý phép toán, hiển thị kết quả, bắt ngoại lệ và reset. Các thuật toán này được mô tả bằng sơ đồ khối chi tiết, thể hiện rõ đầu vào, xử lý và đầu ra. Giao diện sử dụng bố cục grid giúp sắp xếp các widget khoa học và dễ mở rộng. Dữ liệu được lưu dưới dạng thuộc tính trong lớp như `Entry`, `Label`, và `StringVar`. Toàn bộ chương trình được chia thành các hàm riêng biệt để đảm bảo tính mô-đun, dễ đọc và dễ bảo trì. Code được tổ chức rõ ràng, có xử lý DPI cao, và chạy mượt thông qua vòng lặp chính `mainloop()` trong hàm `main()`.*

## CHƯƠNG 4 THỰC NGHIỆM VÀ KẾT LUẬN

### 4.1. Thực nghiệm

#### ❖ Kiểm tra :

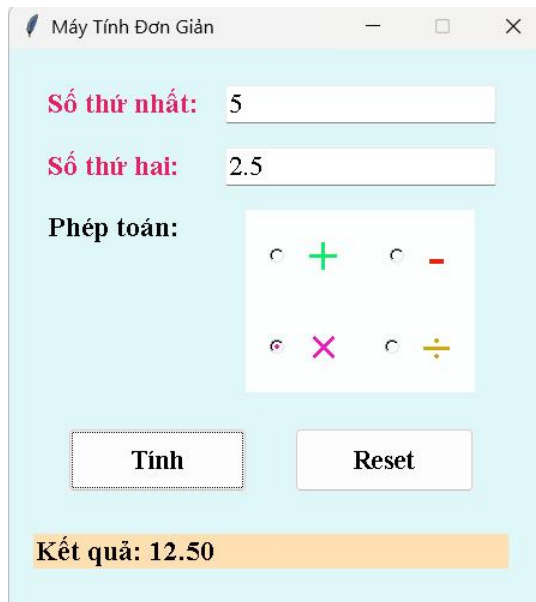
1. **Phép cộng:** Nhập 12.5 và 3.7, chọn "+", kết quả: "Kết quả: 16.20".



The screenshot shows a web-based calculator interface titled "Máy Tính Đơn Giản". It features two input fields: "Số thứ nhất:" with the value "12.5" and "Số thứ hai:" with the value "3.7". Below these is a "Phép toán:" section with four radio buttons for "+", "-", "×", and "÷". The "+" button is selected. At the bottom, there are "Tính" and "Reset" buttons. A yellow box at the very bottom displays the result: "Kết quả: 16.20".

Hình 7 : Kết quả thực hiện phép cộng

2. **Phép nhân:** Nhập 5 và 2.5, chọn "×", kết quả: "Kết quả: 12.50".

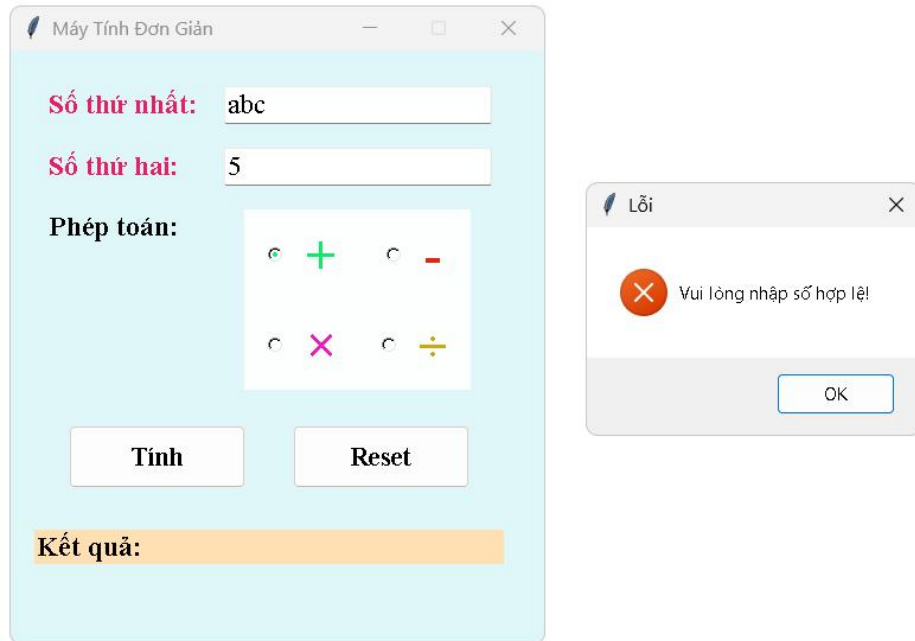


The screenshot shows the same calculator interface as Figure 7. The "Số thứ nhất:" field contains "5" and the "Số thứ hai:" field contains "2.5". In the "Phép toán:" section, the "×" button is selected. The "Tính" and "Reset" buttons are visible. A yellow box at the bottom displays the result: "Kết quả: 12.50".

Hình 8 : Kết quả thực hiện phép nhân

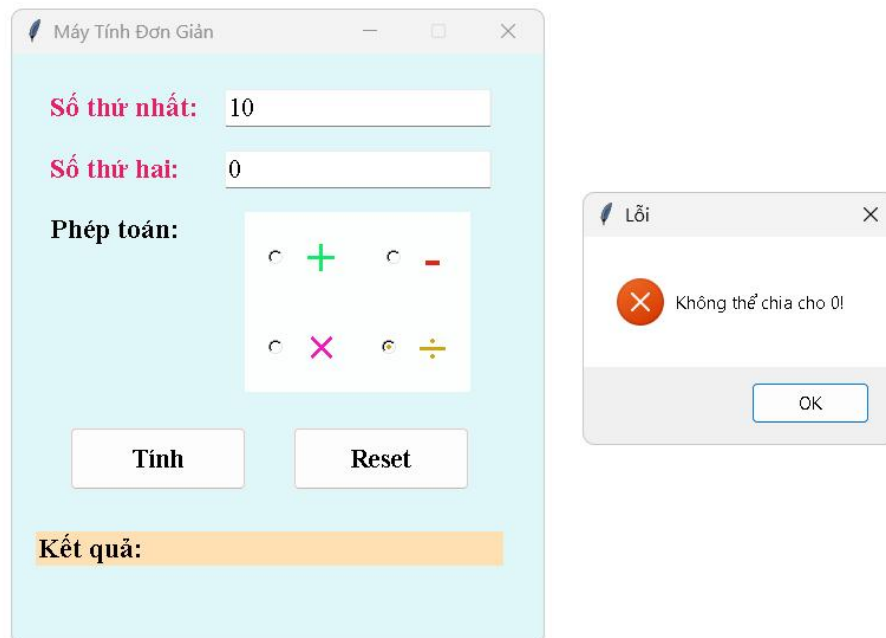


3. **Lỗi nhập liệu:** Nhập "abc" và 5, chọn "-", hiển thị thông báo: "Vui lòng nhập số hợp lệ!".



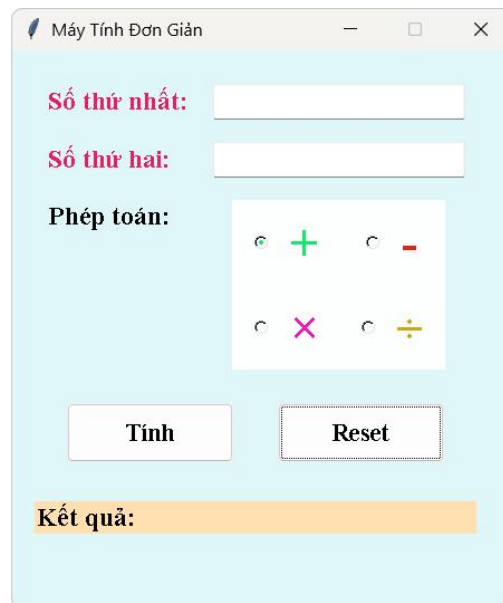
Hình 9 : Kết quả thực hiện phép cộng

4. **Chia cho 0:** Nhập 10 và 0, chọn "÷", hiển thị thông báo: "Không thể chia cho 0!".



Hình 10 : Kết quả thực hiện phép cộng

5. **Reset:** Nhấn "Reset" sau khi nhập dữ liệu, giao diện trở về trạng thái ban đầu.



Hình 11 : Kết quả thực hiện phép cộng

❖ **Đánh giá:**

- + Ưu điểm: Giao diện trực quan, xử lý lỗi tốt, hỗ trợ DPI cao.
- + Nhược điểm: Thiếu phép toán nâng cao, không thay đổi kích thước được.

## 4.2. Kết luận

❖ **Thành tựu đạt được:**

Thông qua quá trình thực hiện bài tập lớn theo yêu cầu giảng viên, em đã xây dựng thành công một ứng dụng máy tính đơn giản với giao diện đồ họa sử dụng thư viện Tkinter. Ứng dụng hỗ trợ bốn phép toán cơ bản gồm cộng, trừ, nhân, chia, đồng thời xử lý tốt các tình huống lỗi như nhập sai dữ liệu hoặc chia cho 0. Giao diện được tổ chức hợp lý bằng bố cục grid, sử dụng các widget hiện đại từ thư viện ttk, kết hợp cùng tính năng hỗ trợ DPI cao giúp giao diện hiển thị sắc nét, thân thiện với người dùng.

❖ **Bài học:**

Quá trình triển khai giúp em hiểu sâu hơn về cách thiết kế giao diện người dùng với Tkinter, cách tổ chức chương trình theo hướng đối tượng để đảm bảo tính rõ ràng, dễ bảo trì. Đồng thời, em cũng nắm được cách sử dụng cấu trúc try-except để xử lý

ngoại lệ trong các tình huống thực tế như nhập sai định dạng hoặc chia cho 0. Việc tùy chỉnh style và tối ưu hiển thị cho màn hình có độ phân giải cao cũng là một kỹ năng thực tiễn quan trọng mà em tích lũy được qua bài tập này.

#### ❖ **Cải tiến:**

Mặc dù chương trình đã đáp ứng tốt các yêu cầu cơ bản, nhưng vẫn còn nhiều hướng để phát triển thêm. Trong các phiên bản tiếp theo, em mong muốn sẽ:

- Bổ sung các phép toán nâng cao như lũy thừa, căn bậc hai, phần trăm...
- Hỗ trợ nhập liệu bằng bàn phím để tăng trải nghiệm người dùng.
- Xây dựng chức năng lưu lại lịch sử phép tính và xem lại kết quả trước đó.
- Thêm khả năng tùy chỉnh giao diện như thay đổi chủ đề (theme), màu sắc, hoặc font chữ.

Những cải tiến này sẽ góp phần nâng cao tính ứng dụng và độ hoàn thiện của phần mềm, đồng thời giúp em rèn luyện thêm kỹ năng lập trình nâng cao và phát triển phần mềm theo hướng chuyên nghiệp hơn.

*Tóm tắt Chương 4 – Thực nghiệm và kết luận: Chương trình được kiểm thử với nhiều trường hợp khác nhau như cộng, nhân, nhập sai dữ liệu, chia cho 0 và chức năng reset, cho kết quả đúng và phản hồi lỗi rõ ràng. Giao diện hoạt động ổn định, dễ sử dụng và hỗ trợ DPI cao. Qua thực nghiệm, chương trình thể hiện rõ ưu điểm về tính trực quan và khả năng xử lý ngoại lệ, nhưng vẫn còn hạn chế như chưa hỗ trợ phép toán nâng cao hay thay đổi kích thước cửa sổ. Việc thực hiện bài tập giúp người học nắm vững cách sử dụng Tkinter, lập trình hướng đối tượng, xử lý lỗi và tùy chỉnh giao diện. Trong tương lai, chương trình có thể được mở rộng thêm các tính năng như phép toán nâng cao, nhập bằng bàn phím, hiển thị lịch sử phép tính và thay đổi giao diện theo chủ đề.*

# TÀI LIỆU THAM KHẢO

1. Giáo trình môn Lập trình Python : Python Programming for the Absolute Beginner- 3rd Edition
2. W3Schools. Python Try Except.  
[https://www.w3schools.com/python/python\\_try\\_except.asp](https://www.w3schools.com/python/python_try_except.asp)
3. GeeksforGeeks. Python GUI – tkinter :  
<https://www.geeksforgeeks.org/python-gui-tkinter/>
4. TkDocs. Tkinter Tutorial - Modern GUI Development.  
<https://tkdocs.com/tutorial/index.html>