TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

**KHOA ĐIỆN TỬ**

**Bộ môn: Công nghệ Thông tin**.

**BÀI TẬP KẾT THÚC MÔN HỌC**

MÔN HỌC

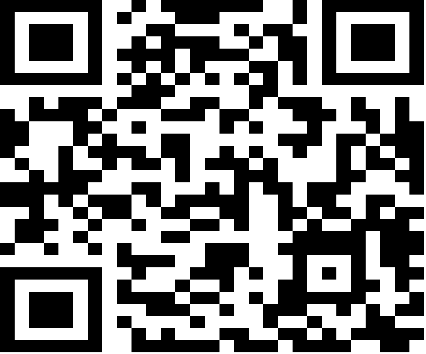
**LẬP TRÌNH PYTHON**

Sinh viên: .Lương Quang Hà . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Lớp: K58KTP.01 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Giáo viên GIẢNG DẠY:TS.Nguyễn Văn Huy . . . . . . . .. . . . . . . . . . .

Link GitHub: https://github.com/luongquangha04/BTL-PY-DE1



**Thái Nguyên – 2025**

|  |  |
| --- | --- |
| **TRƯỜNG ĐHKTCN** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| **KHOA ĐIỆN TỬ** | ***Độc lập - Tự do - Hạnh phúc*** |

**BÀI TẬP KẾT THÚC MÔN HỌC**

**MÔN HỌC: LẬP TRÌNH PYTHON**

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

*Sinh viên:* *Lương Quang Hà*

*Lớp*: K58KTP.01     *Ngành: Kỹ Thuật Máy Tính*

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

*Ngày giao đề* *Ngày hoàn thành*

*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 20....

## GIÁO VIÊN HƯỚNG DẪN

*(Ký ghi rõ họ tên)*

**MỤC LỤC**

[Lời mở đầu 4](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc6407)

[CHƯƠNG 1 GIỚI THIỆU ĐẦU BÀI 5](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc463)

[1.1 Mô tả bài toán 5](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc28427)

[1.1.1 Tính năng chương trình: 5](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc10831)

[*1.1.3* Ví dụ kiểm tra 5](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc556)

[1.2 Thách thức của bài toán 6](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc3705)

[1.3 Kiến thức vận dụng 6](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc20316)

[CHƯƠNG 2 CƠ SỞ LÝ THUYẾT 8](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc23394)

[2.1. Thư viện Python được sử dụng 8](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc30326)

[2.1.1. Thư viện tkinter 8](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc9165)

[2.1.2. Thư viện tkinter.messagebox 8](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc19695)

[2.2. Cấu trúc dữ liệu sử dụng 9](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc10764)

[2.2.1. Danh sách (List) 9](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc26603)

[2.2.2. Chuỗi (String) và biến StringVar 9](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc4241)

[2.3. Kỹ thuật lập trình 9](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc16302)

[2.3.1. Lập trình hướng đối tượng (OOP) 9](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc27053)

[2.3.2. Xử lý ngoại lệ 10](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc21662)

[2.3.3. Quản lý giao diện bằng hệ thống lưới (grid) 10](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc12579)

[2.3.4. Gắn kết sự kiện 10](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc5794)

[2.4. Xử lý và định dạng dữ liệu 10](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc458)

[2.5. Giao diện người dùng (GUI) 11](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc16985)

[CHƯƠNG 3 THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH 12](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc28694)

[3.1 Sơ đồ khối hệ thống 12](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc17874)

[3.1.1 Mô tả các module chính 12](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc24628)

[3.1.2 Biểu đồ phân cấp chức năng 13](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc21758)

[3.2 Sơ đồ khối các thuật toán chính 13](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc9924)

[3.2.1 Mô tả các khối thuật toán chính 13](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc27953)

[3.2.2 Quan hệ đầu vào - ra giữa các khối 14](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc16922)

[3.3 Cấu trúc dữ liệu 15](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc30849)

[3.3.1 Các bảng dữ liệu 15](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc27864)

[3.3.2 Các cấu trúc dữ liệu khác 15](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc25693)

[3.4 Mô tả các hàm trong chương trình 15](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc28322)

[3.4.1 Hàm \_\_init\_\_(self, root) 15](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc29187)

[3.4.2 Hàm calculate(self) 16](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc25409)

[3.4.3 Hàm reset(self) 16](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc14229)

[CHƯƠNG 4 KIỂM THỬ CHƯƠNG TRÌNH 17](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc13281)

[4.1. Chương trình đã làm được 17](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc16085)

[4.2. Kết luận 20](file:///C:\Users\Administrator\Documents\Zalo%20Received%20Files\Python_Ha.docx#_Toc23498)

# Lời mở đầu

Trong thời đại công nghệ thông tin phát triển mạnh mẽ, các ứng dụng giao diện đồ họa người dùng (GUI) đóng vai trò quan trọng trong việc cung cấp các công cụ trực quan, dễ sử dụng cho người dùng cuối. Chương trình ”Máy Tính Đơn Giản” (Simple Calculator GUI) được phát triển nhằm đáp ứng yêu cầu của một bài toán lập trình cơ bản: xây dựng một ứng dụng GUI cho phép người dùng thực hiện các phép toán số học cơ bản (+, -, ×, ÷) trên hai số thực, với khả năng xử lý lỗi và quản lý trạng thái giao diện.

# CHƯƠNG 1 GIỚI THIỆU ĐẦU BÀI

Chương này giới thiệu tổng quan về bài toán ”Máy Tính Đơn Giản” (Simple Calcu- lator GUI) dựa trên đề bài được giao và mã nguồn trong tài liệu Python\_Ha.py. Nội dung bao gồm mô tả tính năng của chương trình, các thách thức gặp phải, và các kiến thức được vận dụng để giải quyết bài toán.

## Mô tả bài toán

Chương trình ”Máy Tính Đơn Giản” là một ứng dụng giao diện người dùng đồ họa (GUI) được phát triển bằng Python, sử dụng thư viện tkinter. Mục tiêu của chương trình là cung cấp một công cụ cho phép người dùng thực hiện các phép toán cơ bản (+, -, ×, ÷) trên hai số thực được nhập từ giao diện.

### Tính năng chương trình:

Lựa chọn phép toán (+, -, ×, ÷) thông qua nhóm nút radio (Radiobutton).

## Xử lý và hiển thị kết quả:

* + - * + Nhấn nút ”Tính” để thực hiện phép toán và hiển thị kết quả trên nhãn (Label) với định dạng số thực có hai chữ số thập phân.
        + Nhấn nút ”Reset” để xóa nội dung hai ô nhập liệu và đặt lại phép toán mặc định là cộng (+).

## Xử lý lỗi:

* + - * + Kiểm tra và hiển thị thông báo lỗi thông qua hộp thoại (messagebox) trong các trường hợp:

Nhập liệu không phải số hợp lệ (gây lỗi ValueError).

Thực hiện phép chia cho số 0 (gây lỗi ZeroDivisionError).

### Đầu vào và đầu ra

## Đầu vào:

* + - * + Hai số thực được nhập qua hai ô Entry.
        + Phép toán được chọn từ nhóm Radiobutton (+, -, ×, ÷).

## Đầu ra:

* + - * + Kết quả phép toán hiển thị trên nhãn (Label) với định dạng Kết quả:

<số thực> (ví dụ: Kết quả: 7.00).

* + - * + Thông báo lỗi qua hộp thoại nếu có lỗi xảy ra.

### Ví dụ kiểm tra

* + - * **Trường hợp hợp lệ**: Nhập 3.5 và 2, chọn phép toán ×, kết quả hiển thị:

Kết quả: 7.00.

## Trường hợp lỗi:

* + - * + Nhập ”a” và 1, chương trình hiển thị hộp thoại: Vui lòng nhập số hợp lệ..
        + Nhập 5 và 0, chọn phép toán ÷, chương trình hiển thị hộp thoại: Không thể chia cho 0.

## Thách thức của bài toán

1. **Xử lý lỗi đầu vào**:
   * Đảm bảo chương trình bắt được lỗi khi người dùng nhập dữ liệu không

phải số (ValueError) hoặc thực hiện phép chia cho 0 (ZeroDivisionError).

* + Hiển thị thông báo lỗi rõ ràng và thân thiện với người dùng thông qua hộp thoại.

## Thiết kế giao diện người dùng:

* + Sắp xếp các thành phần giao diện (ô nhập liệu, nút radio, nút bấm, nhãn) sao cho trực quan và dễ sử dụng.
  + Sử dụng hệ thống lưới (Grid) để bố trí các thành phần một cách gọn gàng và đồng nhất.

## Quản lý trạng thái:

* + Duy trì trạng thái của phép toán được chọn thông qua biến StringVar.
  + Cập nhật kết quả ngay khi nhấn nút ”Tính” và đặt lại trạng thái ban đầu khi nhấn nút ”Reset”.

## Tính chính xác của phép toán:

* + Đảm bảo các phép toán (+, -, ×, ÷) được thực hiện đúng và kết quả được định dạng với hai chữ số thập phân.

## Kiến thức vận dụng

Để giải quyết bài toán, các kiến thức và kỹ thuật sau đã được vận dụng:

## Thư viện tkinter:

* + Sử dụng các thành phần giao diện như Label, Entry, Button, Radiobutton

để xây dựng GUI.

* + Sử dụng messagebox để hiển thị thông báo lỗi.
  + Sử dụng hệ thống lưới (Grid) để bố trí giao diện.

## Lập trình hướng đối tượng (OOP):

* + Xây dựng lớp SimpleCalculator để đóng gói giao diện và logic xử lý.
  + Sử dụng các phương thức ( init , calculate, reset) để quản lý các chức năng của chương trình.

## Xử lý ngoại lệ:

* + Sử dụng khối try-except để bắt và xử lý lỗi ValueError (dữ liệu nhập không hợp lệ) và ZeroDivisionError (chia cho 0).

## Quản lý trạng thái giao diện:

* + Sử dụng biến StringVar để theo dõi phép toán được chọn.
  + Cập nhật nội dung nhãn và ô nhập liệu thông qua các phương thức như config, delete, và set.

## Kiểm tra và định dạng dữ liệu:

* + Chuyển đổi dữ liệu nhập từ chuỗi sang số thực (float) để thực hiện phép toán.
  + Định dạng kết quả với hai chữ số thập phân bằng :.2f

# CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

Chương này trình bày các khái niệm và kỹ thuật chuyên môn được sử dụng trong chương trình “Máy Tính Đơn Giản” (Simple Calculator GUI) dựa trên mã nguồn trong tài liệu Python\_Ha.py. Nội dung bao gồm các thư viện, cấu trúc dữ liệu và kỹ thuật lập trình được áp dụng để xây dựng chương trình.

## 2.1. Thư viện Python được sử dụng

### 2.1.1. Thư viện tkinter

**Mô tả:**  
tkinter là thư viện chuẩn của Python dùng để xây dựng giao diện đồ họa người dùng (GUI). Thư viện này cung cấp các thành phần như nhãn (Label), ô nhập liệu (Entry), nút bấm (Button), nút chọn (Radiobutton), cùng với công cụ quản lý sự kiện và hiển thị thông báo.

**Ứng dụng trong chương trình:**

Tạo cửa sổ chính bằng tk.Tk().

Sử dụng các thành phần GUI:

Label: Hiển thị văn bản tĩnh như "Số thứ nhất", "Kết quả".

Entry: Nhận dữ liệu từ người dùng (hai số thực).

Button: Tạo nút “Tính” và “Reset” để tính toán hoặc xóa dữ liệu.

Radiobutton: Cho phép chọn phép toán (+, –, ×, ÷).

Sử dụng messagebox để hiển thị thông báo lỗi (ví dụ: nhập sai hoặc chia cho 0).

Bố trí giao diện bằng hệ thống lưới (grid) giúp giao diện gọn gàng, dễ quan sát.

### 2.1.2. Thư viện tkinter.messagebox

**Mô tả:**  
Là module con của tkinter, dùng để hiển thị các hộp thoại thông báo (message box), như cảnh báo hoặc lỗi.

**Ứng dụng trong chương trình:**

Hiển thị thông báo lỗi khi người dùng nhập dữ liệu không hợp lệ (ValueError) hoặc thực hiện phép chia cho 0 (ZeroDivisionError).

## 2.2. Cấu trúc dữ liệu sử dụng

### 2.2.1. Danh sách (List)

**Mô tả:**  
Danh sách là cấu trúc dữ liệu động trong Python, cho phép lưu trữ nhiều phần tử với kiểu dữ liệu khác nhau.

**Ứng dụng trong chương trình:**

Danh sách operations = [("+", "Cộng"), ("-", "Trừ"), ("\*", "Nhân"), ("/", "Chia")] được sử dụng để định nghĩa các phép toán và nhãn tương ứng.

Duyệt danh sách để tạo các Radiobutton tương ứng với từng phép toán.

### 2.2.2. Chuỗi (String) và biến StringVar

**Mô tả:**  
Chuỗi là kiểu dữ liệu cơ bản, còn StringVar là lớp của tkinter dùng để quản lý chuỗi có thể thay đổi trong giao diện.

**Ứng dụng trong chương trình:**

Biến self.operation (kiểu StringVar) lưu phép toán được chọn.

Giá trị này thay đổi khi người dùng chọn một phép toán và được sử dụng trong hàm calculate.

## 2.3. Kỹ thuật lập trình

### 2.3.1. Lập trình hướng đối tượng (OOP)

**Mô tả:**  
Chương trình sử dụng OOP bằng cách định nghĩa lớp SimpleCalculator, bao gồm thuộc tính và phương thức để tổ chức logic và giao diện.

**Ứng dụng trong chương trình:**

Lớp SimpleCalculator quản lý toàn bộ ứng dụng

Các phương thức như \_\_init\_\_(), calculate(), và reset() thực hiện các chức năng tương ứng.

### 2.3.2. Xử lý ngoại lệ

**Mô tả:**  
Kỹ thuật try-except giúp quản lý lỗi, tránh chương trình bị dừng đột ngột.

**Ứng dụng trong chương trình:**

Trong calculate(), xử lý lỗi nhập không hợp lệ (ValueError) và chia cho 0 (ZeroDivisionError).

Khi xảy ra lỗi, hiển thị thông báo bằng messagebox.showerror().

### 2.3.3. Quản lý giao diện bằng hệ thống lưới (grid)

**Mô tả:**  
Hệ thống grid trong tkinter giúp sắp xếp các thành phần theo hàng và cột.

**Ứng dụng trong chương trình:**

Các thành phần như Label, Entry, Radiobutton, Button được bố trí hợp lý bằng các tham số như row, column, padx, pady, sticky.

### 2.3.4. Gắn kết sự kiện

**Mô tả:**  
Các sự kiện như nhấn nút được liên kết với các hàm xử lý qua thuộc tính command.

**Ứng dụng trong chương trình:**

Nút “Tính” gọi hàm calculate().

Nút “Reset” gọi hàm reset() để làm sạch giao diện.

## 2.4. Xử lý và định dạng dữ liệu

**Mô tả:**  
Chuyển đổi dữ liệu nhập từ chuỗi sang số thực và hiển thị kết quả có định dạng.

**Ứng dụng trong chương trình:**

Dữ liệu từ Entry được ép kiểu bằng float

Kết quả được định dạng với hai chữ số thập phân bằng cú pháp "{result:.2f}".

## 2.5. Giao diện người dùng (GUI)

**Mô tả:**  
Giao diện được thiết kế bằng tkinter, thân thiện và trực quan cho người dùng.

**Ứng dụng trong chương trình:**

Entry: Nhập hai số thực

Radiobutton: Chọn phép toán

Button: Gồm nút “Tính” và “Reset”

Label: Hiển thị kết quả

messagebox: Hiển thị thông báo lỗi

# CHƯƠNG 3 THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

Chương này trình bày chi tiết quá trình thiết kế và xây dựng chương trình “Máy Tính Đơn Giản” dựa trên mã nguồn trong tập tin Python\_Ha.py. Nội dung bao gồm: sơ đồ khối hệ thống, thuật toán chính, cấu trúc dữ liệu và mô tả các hàm.

## 3.1 Sơ đồ khối hệ thống

### 3.1.1 Mô tả các module chính

Chương trình được thiết kế theo mô hình hướng đối tượng, tổ chức thành lớp SimpleCalculator với 3 module chính:

**Module giao diện người dùng (GUI)**:

**Chức năng**: Giao tiếp với người dùng, cho phép nhập số, chọn phép toán và hiển thị kết quả.

**Thành phần**:

tk.Tk: Cửa sổ chính chứa toàn bộ giao diện.

tk.Label: Hiển thị văn bản và kết quả.

tk.Entry: Nhận đầu vào là hai số thực.

tk.Radiobutton: Cho phép chọn phép toán: +, -, ×, ÷.

tk.Button: Gồm nút "Tính" và "Reset".

messagebox: Hiển thị thông báo lỗi khi cần.

**Module xử lý phép toán**:

**Chức năng**: Thực hiện các phép toán cơ bản và xử lý lỗi.

**Thành phần**:

calculate(): Thực hiện phép toán dựa trên dữ liệu đầu vào và cập nhật kết quả.

Xử lý lỗi: ValueError (nhập sai định dạng) và ZeroDivisionError (chia cho 0).

**Module quản lý trạng thái**:

**Chức năng**: Đặt lại trạng thái ban đầu của giao diện.

**Thành phần**:

reset(): Xóa dữ liệu nhập và cập nhật giao diện về mặc định.

### 3.1.2 Biểu đồ phân cấp chức năng

**Chức năng chính**: Máy tính đơn giản

**Nhập liệu**:

Nhập hai số thực.

Chọn phép toán bằng nút radio.

**Thực hiện phép toán**:

Chuyển đổi dữ liệu sang kiểu số thực.

Thực hiện phép toán tương ứng.

Hiển thị kết quả trên nhãn.

**Xử lý lỗi**:

Cảnh báo lỗi nhập sai hoặc chia cho 0 bằng hộp thoại.

**Đặt lại trạng thái**:

Xóa nội dung các ô nhập.

Đặt lại phép toán mặc định là cộng.

Cập nhật kết quả về trạng thái ban đầu.

## 3.2 Sơ đồ khối các thuật toán chính

### 3.2.1 Mô tả các khối thuật toán chính

**Thuật toán tính toán (**calculate**)**:

**Đầu vào**: Hai chuỗi từ self.entry1 và self.entry2, phép toán từ self.operation.

**Đầu ra**: Kết quả hiển thị trên self.result\_label.

**Luồng xử lý**:

Đọc giá trị từ ô nhập và chuyển thành float.

Lấy phép toán từ self.operation.

Thực hiện phép toán tương ứng.

Kiểm tra chia cho 0 nếu là phép chia.

Hiển thị kết quả với định dạng hai chữ số thập phân.

Nếu lỗi xảy ra, hiển thị thông báo lỗi bằng messagebox.

**Thuật toán đặt lại trạng thái (**reset**)**:

**Đầu vào**: Không có.

**Đầu ra**: Các ô nhập trống, phép toán là cộng (+), nhãn kết quả trở về "Kết quả:".

**Luồng xử lý**:

Xóa nội dung hai ô nhập.

Đặt lại phép toán mặc định.

Cập nhật nhãn kết quả.

### 3.2.2 Quan hệ đầu vào - ra giữa các khối

**Nhập liệu → Tính toán**: Dữ liệu từ các ô nhập và phép toán truyền vào hàm calculate.

**Tính toán → Hiển thị kết quả**: Kết quả hiển thị trên result\_label hoặc thông báo lỗi từ messagebox.

**Đặt lại → Giao diện**: Hàm reset cập nhật lại các thành phần giao diện.

## 3.3 Cấu trúc dữ liệu

### 3.3.1 Các bảng dữ liệu

Chương trình không sử dụng cơ sở dữ liệu hay bảng dữ liệu phức tạp mà lưu trữ thông tin thông qua thành phần giao diện:

self.result\_label **(tk.Label)**:

**Thông tin hiển thị**: Chuỗi kết quả như “Kết quả: 7.00” hoặc trạng thái mặc định.

**Cập nhật**: Qua hàm calculate() hoặc reset().

### 3.3.2 Các cấu trúc dữ liệu khác

self.operation **(tk.StringVar)**: Biến lưu phép toán được chọn (+, -, \*, /).

operations **(list)**: Danh sách các phép toán:

python

Sao chépChỉnh sửa

operations = [("+", "Cộng"), ("-", "Trừ"), ("\*", "Nhân"), ("/", "Chia")]

self.entry1**,** self.entry2 **(tk.Entry)**: Nhận chuỗi đầu vào, được chuyển thành số thực trong calculate.

## 3.4 Mô tả các hàm trong chương trình

### 3.4.1 Hàm \_\_init\_\_(self, root)

**Chức năng**: Khởi tạo giao diện và các thành phần của lớp SimpleCalculator.

**Tham số**: root là cửa sổ chính (tk.Tk).

**Hoạt động**:

Đặt tiêu đề cửa sổ.

Tạo nhãn, ô nhập, nút radio, nút bấm và nhãn kết quả.

Dùng grid() để bố trí thành phần.

Khởi tạo biến self.operation với giá trị mặc định “+”.

### 3.4.2 Hàm calculate(self)

**Chức năng**: Thực hiện phép toán và hiển thị kết quả.

**Tham số**: Không có.

**Hoạt động**:

Lấy giá trị nhập, chuyển thành float.

Thực hiện phép toán theo self.operation.

Cập nhật kết quả dạng Kết quả: {result:.2f}.

Xử lý lỗi bằng messagebox.

### 3.4.3 Hàm reset(self)

**Chức năng**: Đặt lại giao diện về trạng thái ban đầu

**Tham số**: Không có.

**Hoạt động**:

Xóa nội dung các ô nhập.

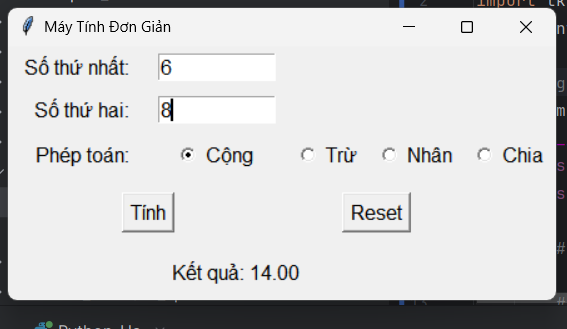
Đặt lại phép toán về “+”.

Cập nhật nhãn kết quả về “Kết quả:”.

# **CHƯƠNG 4 KIỂM THỬ CHƯƠNG TRÌNH**

# 4.1. Chương trình đã làm được

Cộng



# Trừ

# 

# Nhân

# 

# Chia

# 

# Reset

# 

### 4.2. Kết luận

**Sản phẩm đã làm được những gì:**

Giao diện đồ họa thân thiện với người dùng, sử dụng thư viện Tkinter.

Cho phép người dùng nhập hai số và chọn một trong bốn phép toán: Cộng, Trừ, Nhân, Chia.

Xử lý và hiển thị kết quả chính xác cho từng phép toán.

Có nút "Tính" để thực hiện phép toán và nút "Reset" để xóa dữ liệu đã nhập.

Có kiểm tra lỗi chia cho 0 và báo lỗi phù hợp cho người dùng.

**Học được gì:**

Hiểu và sử dụng cơ bản thư viện Tkinter để tạo GUI trong Python.

Nắm vững cách sử dụng các widget như Label, Entry, Button, Radiobutton.

Biết cách xử lý sự kiện người dùng và cập nhật giao diện động.

Rèn luyện tư duy lập trình hướng đối tượng trong Python.

**Sẽ cải tiến gì:**

Bổ sung kiểm tra đầu vào để đảm bảo chỉ nhận giá trị số.

Thêm chức năng lịch sử tính toán để lưu và hiển thị các phép toán đã thực hiện.

Thiết kế lại giao diện đẹp mắt hơn với màu sắc và bố cục rõ ràng.

Có thể mở rộng thêm các phép toán như lũy thừa, căn bậc hai, phần trăm,...

Chuyển chương trình thành file .exe để có thể chạy độc lập trên Windows mà không cần Python.