**TRƯỜNG ĐẠI HỌC HỌC VĂN LANG**
**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO ĐỒ ÁN MÔN HỌC**

# LẬP TRÌNH PYTHON NÂNG CAO

**NGÀNH: CÔNG NGHỆ THÔNG TIN**

***Tên Đồ án:***

# GUI CALCULATOR USING TKINTER IN PYTHON

**SVTH: 2274802010739 _Nguyễn Ngọc Quý**
**GVHD: Huỳnh Thái Học**

**Tp. Hồ Chí Minh – năm 2024**

**LỜI CẢM ƠN**

Viết một báo cáo đồ án môn học là một trong những việc khó nhất mà chúng em phải hoàn thành trong quá trình học một môn học. Trong quá trình thực hiện đề tài chúng em đã gặp rất nhiều khó khăn và bỡ ngỡ. Nếu không có những sự giúp đỡ và lời động viên chân thành của nhiều người có lẽ chúng em khó có thể hoàn thành tốt tiểu luận này. Đầu tiên chúng em xin gửi lời biết ơn chân thành đến thầy NV A, người trực tiếp hướng dẫn chúng em hoàn thành tiểu luận này. Những ý kiến đóng góp của thầy là vô cùng hữu ích, nó giúp chúng em nhận ra các khuyết điểm của đồ án. Cảm ơn thầy và các bạn trường Đại học Văn Lang là những người đã cùng nhóm em sát cánh và trải nghiệm để hoàn thành đồ án môn học.
*Nhóm thực hiện báo cáo*

# MỤC LỤC

# CHƯƠNG 1. CƠ SỞ LÝ THUYẾT – BÀI TOÁN

## 1.1. CƠ SỞ LÝ THUYẾT

Máy tính bỏ túi là một công cụ tiện lợi, hỗ trợ người dùng thực hiện các phép toán cơ bản và nâng cao một cách nhanh chóng và chính xác. Sự phát triển của các ứng dụng máy tính đã giúp người dùng truy cập vào các công cụ này một cách dễ dàng thông qua máy tính hoặc thiết bị di động. Trong đồ án này, chúng ta sử dụng ngôn ngữ lập trình Python kết hợp với thư viện giao diện người dùng tkinter để xây dựng một ứng dụng máy tính bỏ túi

đơn giản, đáp ứng các yêu cầu cơ bản như tính toán các phép toán số học và các phép toán khoa học phổ biến.

**1. Kiến thức về lập trình giao diện người dùng (GUI)**

- Tkinter là một thư viện tiêu chuẩn trong Python, hỗ trợ xây dựng giao diện đồ họa đơn giản. Thư viện này cung cấp các widget như Button, Entry, Label, giúp tạo ra các thành phần giao diện như nút bấm, ô nhập liệu, nhãn.
- Trong ứng dụng này, tkinter giúp tạo giao diện máy tính bỏ túi thân thiện, dễ sử dụng.

**2. Các phép toán cơ bản**

Máy tính bỏ túi thường hỗ trợ các phép toán cơ bản bao gồm:

- Phép cộng (+), phép trừ (-), phép nhân (*), và phép chia (/).
- Các phép tính mở rộng như bình phương (x^2), căn bậc hai (√x), và phần trăm (%).
- Các phép toán này đều có thể thực hiện dễ dàng thông qua các phép tính số học cơ bản trong Python.

**3. Xử lý chuỗi và biểu thức số học**

- Để thực hiện tính toán, ứng dụng cần xử lý các biểu thức do người dùng nhập vào. Biểu thức toán học có thể chứa các ký tự đặc biệt và phép toán khác nhau. eval() là một hàm của Python giúp đánh giá biểu thức dạng chuỗi và trả về kết quả. Tuy nhiên, eval() cần được sử dụng cẩn thận để tránh các lỗi hoặc rủi ro bảo mật khi xử lý các chuỗi nhập từ người dùng.
- Các thao tác như xóa ký tự cuối, chuyển đổi dấu âm/dương được thực hiện thông qua việc xử lý chuỗi, hỗ trợ người dùng nhập và điều chỉnh dữ liệu nhanh chóng.

**4. Xử lý lỗi**

- Khi người dùng nhập vào biểu thức không hợp lệ, ứng dụng sẽ hiển thị thông báo lỗi để báo cho người dùng biết, nhằm tăng trải nghiệm sử dụng. Điều này có thể được thực hiện bằng cách sử dụng các khối try-except để bắt và xử lý các ngoại lệ.

### 1.2. BÀI TOÁN

**Đặt vấn đề**

Người dùng thường xuyên cần thực hiện các phép tính cơ bản trong học tập và công việc. Tuy nhiên, không phải lúc nào họ cũng có một chiếc máy tính bỏ túi chuyên dụng. Do đó, nhu cầu sử dụng các ứng dụng máy tính trên máy tính cá nhân trở nên phổ biến. Mục tiêu của đồ án là xây dựng một ứng dụng máy tính bỏ túi sử dụng tkinter, cung cấp đầy đủ các chức năng cơ bản của một máy tính cầm tay thông thường, đồng thời có giao diện đơn giản, dễ sử dụng.

**Yêu cầu của bài toán**

Ứng dụng máy tính bỏ túi phải đáp ứng được các yêu cầu sau:

1. Thực hiện các phép toán số học cơ bản: Cộng, trừ, nhân, chia.
2. Thực hiện các phép toán khoa học cơ bản: Lấy nghịch đảo, bình phương, căn bậc hai, phần trăm.
3. Chức năng xóa và chỉnh sửa:
    - Cho phép người dùng xóa toàn bộ (nút "C") hoặc xóa ký tự cuối cùng (nút "DEL").
4. Xử lý lỗi: Thông báo khi người dùng nhập biểu thức không hợp lệ.
5. Thiết kế giao diện thân thiện, dễ sử dụng: Các nút bấm phải được bố trí hợp lý, dễ thao tác.

# CHƯƠNG 2. ÁP DỤNG BÀI TOÁN MINIMAX

## 2.1 Phát biểu bài toán

Ứng dụng máy tính bỏ túi thường được thiết kế để hỗ trợ người dùng thực hiện các phép tính cơ bản và một số phép tính khoa học. Yêu cầu đặt ra là xây dựng một ứng dụng máy tính có giao diện đơn giản, dễ sử dụng, và có thể thực hiện các phép tính số học phổ biến (cộng, trừ, nhân, chia), cùng với một số chức năng mở rộng như tính bình phương, căn bậc hai, tính phần trăm và đổi dấu âm/dương.

Ứng dụng này không phải là một trò chơi, nên thuật toán Minimax không được áp dụng trực tiếp vào tính toán. Tuy nhiên, để nâng cao khả năng mở rộng của chương trình trong tương lai, việc thiết kế phần mềm theo cách thức phân tách các mô-đun tính toán và giao diện là rất cần thiết, giúp dễ dàng tích hợp thêm các thuật toán tối ưu như Minimax nếu ứng dụng được mở rộng cho các trò chơi toán học hoặc tính toán chiến lược.

## 2.2 Hướng giải quyết bài toán

**Xây dựng giao diện người dùng (UI)** bằng thư viện tkinter của Python:

- Thiết kế các nút bấm tương ứng với các phép toán cơ bản, các chức năng đặc biệt ( C, phần trăm).
- Thiết lập hiển thị kết quả và bố trí các nút một cách hợp lý để người dùng dễ thao tác.

**Xử lý các phép tính cơ bản và nâng cao**:

- Sử dụng hàm eval() để đánh giá các biểu thức số học do người dùng nhập vào.
- Xử lý từng loại phép tính và các chức năng đặc biệt như đổi dấu, tính phần trăm, căn bậc hai.

**Xử lý lỗi và đầu vào**:

- Kiểm tra các lỗi trong quá trình nhập biểu thức hoặc tính toán, báo lỗi khi biểu thức không hợp lệ.

**Phân tách các mô-đun tính toán và giao diện**:

- Tách các chức năng tính toán và các sự kiện giao diện để có thể mở rộng hoặc thay đổi dễ dàng trong tương lai, chẳng hạn nếu muốn áp dụng thuật toán Minimax cho các ứng dụng toán học khác.

## 2.3 Mã nguồn

Mã nguồn sẽ được chia thành các phần cụ thể như sau:

### 2.3.1 Tạo giao diện người dùng (UI)

Thiết lập giao diện với tkinter, tạo các nút bấm cho từng phép toán và chức năng cụ thể.

```python
# Khởi tạo cửa sổ chính
root = tk.Tk()
root.title("Calculator")
root.geometry("405x550")
root.config(bg="white")
root.resizable(0, 0)

# Tạo biến lưu trữ văn bản
entry_var = tk.StringVar()

# Thiết kế hộp nhập và hiển thị kết quả
entry = tk.Entry(root, textvariable=entry_var, font=("Arial", 33), justify='right', bd=10, insertwidth=2, bg="#333333", fg="white")
entry.grid(row=0, column=0, columnspan=4, padx=10, pady=20, sticky="nsew")
```

### 2.3.2. Xử lý các phép toán và chức năng

Hàm button_click xử lý tất cả các phép toán, bao gồm các phép tính số học cơ bản và các chức năng đặc biệt.

```
# Hàm xử lý sự kiện khi nhấn nút
def click(button_text):
    current_expression = entry.get()
    if button_text == "C":
        entry.delete(0, tk.END)
    elif button_text == "=":
        try:
            result = str(eval(current_expression))
            entry.delete(0, tk.END)
            entry.insert(tk.END, result)
        except Exception:
            messagebox.showerror("Error", "Biểu thức không hợp lệ")
    elif button_text == "√":
        try:
            result = str(math.sqrt(float(current_expression)))
            entry.delete(0, tk.END)
            entry.insert(tk.END, result)
        except ValueError:
            messagebox.showerror("Error", "Số không hợp lệ cho căn bậc hai")
    elif button_text == "%":
        try:
            result = str(float(current_expression) / 100)
            entry.delete(0, tk.END)
            entry.insert(tk.END, result)
        except ValueError:
            messagebox.showerror("Error", "Biểu thức không hợp lệ cho %")
    elif button_text == "DEL":
        entry.delete(len(current_expression) - 1, tk.END)
    elif button_text == "x^2":
        try:
            result = str(float(current_expression) ** 2)
            entry.delete(0, tk.END)
            entry.insert(tk.END, result)
        except ValueError:
            messagebox.showerror("Error", "Biểu thức không hợp lệ cho mũ 2")
    else:
        entry.insert(tk.END, button_text)
```

### 2.3.3 Bố trí các nút bấm và thiết lập sự kiện

Tạo bố cục nút bấm và gán các sự kiện cho từng nút, phân loại màu sắc theo chức năng của từng loại nút.

```python
# Tạo các nút và thêm vào lưới
row_val = 1
col_val = 0

for button_text in buttons:
    if button_text:
        # Bỏ qua các nút trống
        color = "white"  # Màu trắng mặc định cho các nút
        if button_text == "=":
            color = "green"  # Màu xanh lá cho nút "="
        elif button_text in {'+', '-', '*', '/', 'DEL', '√', '%', '(', ')', 'x^2'}:
            color = "gray70"  # Màu xám cho các nút chức năng
        elif button_text == "C":
            color = "red"  # Màu đỏ cho nút "C"

        button = tk.Button(root, text=button_text, width=5, height=2, font=("Arial", 14),
                           bg=color, fg="black", relief="raised", command=lambda text=button_text: click(text))
        button.grid(row=row_val, column=col_val, padx=5, pady=5,sticky="nsew")

    col_val += 1
    if col_val > 3:
        col_val = 0
        row_val += 1

# Cho phép các nút co giãn theo chiều dọc và chiều ngang
for i in range(4):
    root.columnconfigure(i, weight=1)
for i in range(7):
    root.rowconfigure(i, weight=1)

# Chạy ứng dụng
root.mainloop()
```

…..

### *Nhận xét –ưu điểm & nhược điểm

**Ưu điểm**:

- Giao diện thân thiện, dễ sử dụng: Giao diện của ứng dụng được thiết kế đơn giản với bố cục các nút bấm hợp lý, giúp người dùng dễ dàng thao tác ngay cả khi sử dụng lần đầu.
- Thực hiện các phép toán cơ bản nhanh chóng và chính xác: Ứng dụng đáp ứng tốt các phép tính số học phổ biến như cộng, trừ, nhân, chia, cũng như các chức năng tính toán mở rộng (bình phương, căn bậc hai, phần trăm).
- Dễ dàng mở rộng và bảo trì: Việc phân tách rõ ràng giữa các mô-đun tính toán và giao diện giúp dễ dàng thay đổi hoặc bổ sung chức năng mới mà không ảnh hưởng đến cấu trúc tổng thể của ứng dụng.
- Khả năng xử lý ngoại lệ: Chương trình có cơ chế xử lý lỗi khi người dùng nhập biểu thức không hợp lệ, giúp tránh tình trạng treo hoặc báo lỗi không mong muốn.

- Tính linh hoạt: Ứng dụng có khả năng tích hợp thêm các phép toán hoặc thuật toán phức tạp như Minimax nếu cần thiết trong tương lai.

**Nhược điểm**:

- Giới hạn trong các phép toán nâng cao: Hiện tại, ứng dụng chỉ hỗ trợ các phép toán cơ bản. Để hỗ trợ các phép toán phức tạp hơn (như hàm lượng giác, logarit), cần mở rộng mã nguồn và tích hợp thêm các thư viện toán học.
- Rủi ro bảo mật khi sử dụng eval(): Hàm eval() có thể tiềm ẩn rủi ro bảo mật nếu không kiểm soát đầu vào chặt chẽ, đặc biệt khi người dùng nhập các ký tự hoặc biểu thức không hợp lệ.
- Tính năng chưa phong phú: So với các ứng dụng máy tính bỏ túi chuyên nghiệp, ứng dụng này còn thiếu các tính năng nâng cao như bộ nhớ lưu tạm, lịch sử phép tính, hoặc khả năng nhập các biểu thức toán học phức tạp.
- Không hỗ trợ thao tác phím tắt: Người dùng phải thao tác trực tiếp trên giao diện mà chưa có các phím tắt hoặc tổ hợp phím hỗ trợ, làm giảm tính tiện dụng khi thao tác nhanh.
- Không có kiểm tra và hạn chế đầu vào: Người dùng có thể nhập các ký tự không mong muốn, có thể dẫn đến lỗi chương trình nếu không được xử lý đầy đủ.

# CHƯƠNG 3. KẾT LUẬN ĐỀ TÀI

## 3.Giới thiệu chung về đề tài

### 3.1. Mô tả đề tài:

Đề tài GUI Calculator là một ứng dụng máy tính đơn giản được xây dựng bằng ngôn ngữ lập trình Python và thư viện Tkinter để tạo giao diện người dùng đồ họa (GUI). Mục đích của đề tài là giúp sinh viên phát triển kỹ năng lập trình cơ bản, đặc biệt là lập trình giao diện người dùng, đồng thời học cách xử lý các phép toán cơ bản và nâng cao trong toán học.

### 3.2. Lí do chọn đề tài:

+Tính gần gũi: Máy tính bỏ túi là 1 công cụ hữu ích đóng vai trò quan trọng trong đời sống của con người, từ học sinh đến các tính chất công việc quan trọng. Việc phát triển ứng dụng máy tính giúp sinh viên tìm hiểu sâu hơn về các phép tính và thuật toán vận hành trong thực tế.

+Tính học hỏi: Việc tạo mội giao diện GUI giúp sinh viên có thể nắm các kĩ năng quan trọng trong việc thiết kế xây dựng giao diện và người dùng, đồng thời giúp sinh viên hiểu được tầm quan trọng của Tkinter.

+Đề tài vừa sức cho sinh viên để hoàn thành trong 1 khoảng thời gian ngắn.

+Khả năng cá nhân hóa và sáng tạo.

### 3.3. Phạm vi và đối tượng ứng dụng:

- Phạm vi: Các tính năng cơ bản của GUI Calculator như phép tính cộng trừ nhân chia và nâng cao như lũy thừa, căn bậc 2, phép nghịch đảo,...

-Đối tượng ứng dụng: Sinh viên khoa Công Nghệ Thông Tin.

# 4.Thiết kế hệ thống

### 4.1 Kiến trúc tổng quan:

- Giao diện người dùng UI: UI là phần giao diện trực quan mà người dùng sẽ tương tác với. Đối với một máy tính đơn giản, UI bao gồm:

+Ô nhập: nơi người dùng có thể xem và nhập các phép tính.

+Các nút: Bao gồm các nút số từ 0 đến 9, các nút phép toán (+, -, *, /), nút dấu chấm thập phân (.), và các nút đặc biệt như = để tính toán và C để xóa.

- Chức năng tính toán:

+Xử lý biểu thức từ ô nhập

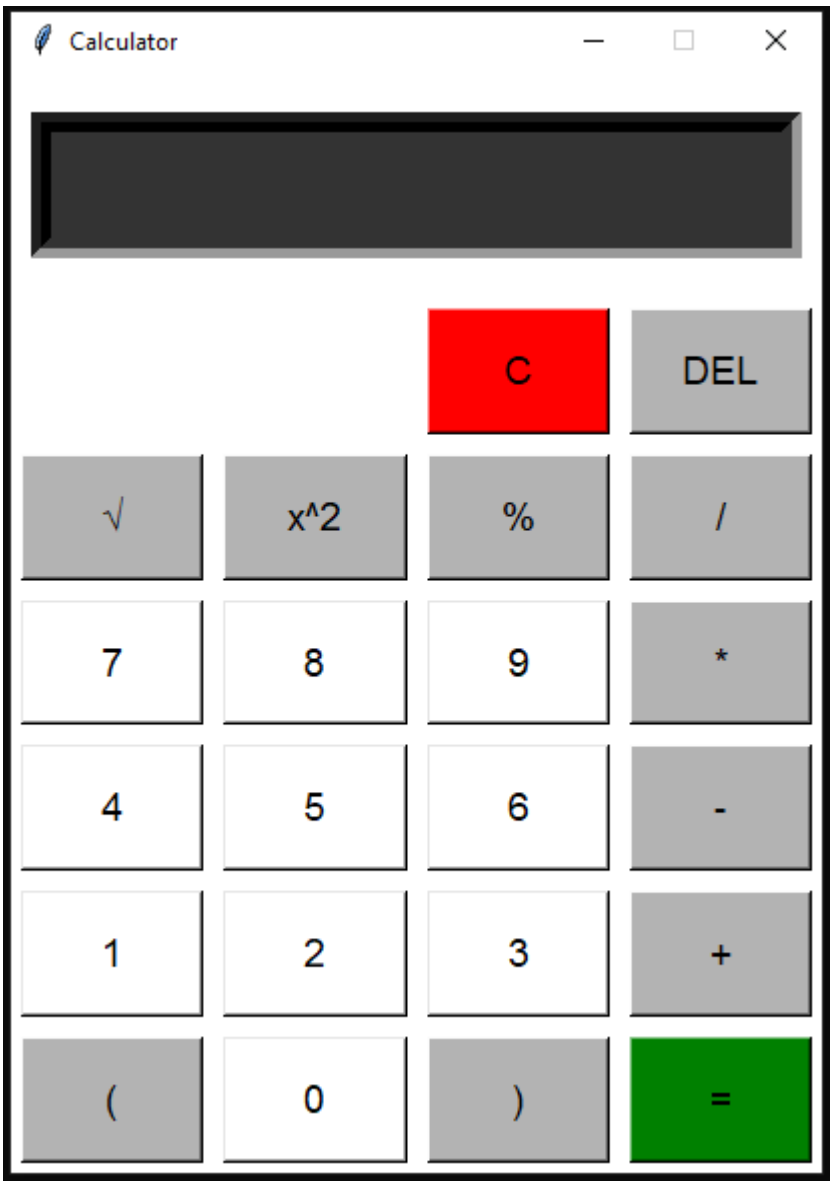
+Kết quả: Trả về và hiển thị trên UI

- Xử lý sự kiện:

+Nhận sự kiện khi người dùng nhấn nút

+Kết nối với các hàm click button(), calculate(), và clear()

### 4.2 Sơ đồ giao diện người dùng (GUI):


Calculator
C
DEL
√
x^2
%
/
7
8
9
*
4
5
6
-
1
2
3
+
(
0
)
=

### 4.3 Mô hình dữ liệu

Mô hình dữ liệu của ứng dụng máy tính bỏ túi này khá đơn giản do không có cơ sở dữ liệu phức tạp. Dưới đây là cấu trúc dữ liệu và các biến quan trọng được sử dụng trong chương trình để quản lý và hiển thị các phép tính.

Các thành phần trong mô hình dữ liệu

4.3.1. Biến lưu trữ phép tính đầu vào (entry_var):

- Kiểu dữ liệu: StringVar
- Chức năng: Đây là biến chính để lưu trữ và hiển thị biểu thức mà người dùng nhập vào. Mỗi khi người dùng nhấn vào một nút, biến này sẽ được cập nhật để hiển thị nội dung mới trên ô nhập.

4.3.2. Các nút trên giao diện:

- Danh sách nút: Bao gồm các nút từ 0 đến 9, các phép toán cơ bản +, -, *, /, và các nút đặc biệt C (xóa), DEL (xóa ký tự cuối), = (tính toán), . (dấu thập phân), và các chức năng mở rộng như √x, x^2,%.
- Kiểu dữ liệu: Các nút được khai báo là các đối tượng Button của thư viện tkinter.
- Chức năng: Mỗi nút sẽ liên kết với một hàm xử lý sự kiện khi nhấn để thêm giá trị hoặc thực hiện các phép tính theo yêu cầu của người dùng.

4.3.3. Các hàm xử lý:

- button_click(value):
  - Chức năng: Nhận giá trị khi người dùng nhấn nút và cập nhật chuỗi biểu thức trong entry_var.
  - Kiểu dữ liệu đầu vào: value là string, biểu thị giá trị của nút được nhấn.
  - Xử lý:
    - Nếu value là một toán tử hoặc số, giá trị đó sẽ được thêm vào chuỗi biểu thức.

        - Nếu value là = thì hàm sẽ tính toán kết quả của biểu thức.
        - Nếu value là C hoặc DEL, nó sẽ xóa biểu thức hoặc ký tự cuối cùng trong biểu thức.
    - calculate():
        - Chức năng: Tính toán và trả về kết quả biểu thức hiện tại trong entry_var.
        - Xử lý: Sử dụng hàm eval() để tính toán biểu thức, sau đó cập nhật kết quả lên entry_var. Nếu có lỗi, hàm sẽ đặt giá trị của entry_var thành "Error".
    - clear():
        - Chức năng: Xóa nội dung của biểu thức trong entry_var.
        - Xử lý: Đặt entry_var về chuỗi rỗng để xóa toàn bộ biểu thức.

4.3.4. Cấu trúc bố trí giao diện:

- Grid layout: Các nút và ô nhập được đặt trong layout dạng lưới (grid), với từng hàng và cột tương ứng với vị trí của các nút. Mỗi nút được đặt vào một ô với các tọa độ xác định trước, giúp giao diện cân đối và rõ ràng.

# 5. Cài đặt và phát triển

## 5.1 Các bước triển khai

Phân tích yêu cầu: Xác định các chức năng cơ bản của máy tính, bao gồm các phép tính số học, phép tính căn bậc hai, lũy thừa, và phần trăm. Ngoài ra, giao diện cần đơn giản, dễ sử dụng, với bố cục rõ ràng.

Thiết kế giao diện: Tạo giao diện người dùng (UI) bằng thư viện tkinter của Python, với một ô nhập để hiển thị phép tính và các nút chức năng (các số, phép toán, và các nút đặc biệt).

Lập trình các chức năng tính toán: Cài đặt các hàm xử lý biểu thức và thực hiện phép tính. Sử dụng hàm eval() để tính toán biểu thức, xử lý các trường hợp lỗi, và đảm bảo tính chính xác của kết quả.

Tích hợp và kiểm thử: Tích hợp các thành phần vào một ứng dụng hoàn chỉnh. Thực hiện các bài kiểm thử với nhiều phép tính khác nhau để đảm bảo ứng dụng hoạt động chính xác và ổn định.

Chỉnh sửa và tối ưu hóa: Dựa trên các bài kiểm thử, tiến hành chỉnh sửa và tối ưu mã nguồn để đảm bảo hiệu suất và tránh các lỗi khi người dùng thao tác.

### 5.2 Các vấn đề gặp phải và giải pháp

Xử lý biểu thức không hợp lệ: Sử dụng hàm eval() có thể gây lỗi khi người dùng nhập biểu thức không hợp lệ. Để xử lý, chương trình có thêm khối try-except để kiểm tra và báo lỗi.

Sử dụng % để tính phần trăm: Cần thay thế % bằng /100 trước khi tính toán để đảm bảo tính chính xác.

Kiểm tra ngoại lệ khi chia cho 0: Để tránh lỗi chia cho 0, chương trình sẽ hiển thị "Error" khi gặp lỗi này.

# 6. Kết quả và đánh giá

### 6.1 Chạy thử và kiểm tra ứng dụng

Ứng dụng máy tính bỏ túi đã được chạy thử và kiểm tra các phép toán cơ bản cũng như các phép toán đặc biệt. Ứng dụng hoạt động ổn định và thực hiện đúng các phép tính đầu vào. Các trường hợp đặc biệt, như phép chia cho 0 hoặc biểu thức không hợp lệ, cũng được xử lý tốt, hiển thị thông báo lỗi khi cần thiết.

### 6.2 Đánh giá về hiệu quả và tính ổn định

Hiệu quả: Ứng dụng thực hiện nhanh chóng các phép tính và hiển thị kết quả tức thì. Giao diện đơn giản và các nút dễ nhận biết giúp người dùng thao tác thuận tiện.

Tính ổn định: Ứng dụng hoạt động ổn định, có thể xử lý ngoại lệ khi người dùng nhập sai biểu thức. Chương trình không gặp sự cố hay treo máy khi thao tác với các phép tính.

### 6.3 Cải tiến và hướng phát triển trong tương lai

Bổ sung các phép toán phức tạp: Trong tương lai, ứng dụng có thể bổ sung các phép tính khoa học như lượng giác, logarit, và các hằng số như π, e để trở thành một máy tính khoa học.

Lịch sử phép tính: Thêm tính năng lưu trữ và xem lại lịch sử các phép tính đã thực hiện để người dùng tiện theo dõi.

Tối ưu giao diện và hỗ trợ phím tắt: Tối ưu giao diện với thiết kế hiện đại hơn và bổ sung phím tắt giúp người dùng thao tác nhanh hơn.

### 6.4 Kết luận và bài học kinh nghiệm

**Kết luận**: Ứng dụng máy tính bỏ túi được phát triển đáp ứng đầy đủ các yêu cầu cơ bản về tính toán và hoạt động ổn định. Giao diện trực quan và các chức năng chính của ứng dụng giúp người dùng dễ dàng thao tác. Bằng cách xử lý ngoại lệ, ứng dụng đảm bảo tính an toàn khi người dùng nhập các biểu thức không hợp lệ, hạn chế lỗi và ngăn chặn sự cố ứng dụng.

**Bài học kinh nghiệm**:

1. **Tầm quan trọng của xử lý ngoại lệ**: Trong quá trình phát triển, việc xử lý các trường hợp ngoại lệ, như phép chia cho 0 hay nhập biểu thức không hợp lệ, đã giúp nâng cao tính ổn định của ứng dụng. Đây là bài học quan trọng khi làm việc với dữ liệu đầu vào từ người dùng.
2. **Thiết kế giao diện trực quan, dễ dùng**: Qua quá trình phát triển, nhóm nhận thấy việc thiết kế giao diện đơn giản nhưng dễ thao tác là yếu tố cần thiết để tạo ra một trải nghiệm người dùng tốt.
3. **Quản lý và kiểm thử mã nguồn**: Việc thường xuyên kiểm thử trong quá trình cài đặt giúp phát hiện sớm các lỗi phát sinh và sửa chữa kịp thời. Bài học ở đây là nên chia nhỏ từng chức năng để thử nghiệm độc lập, giúp tối ưu hóa quy trình phát triển và đảm bảo chất lượng.
4. **Khả năng mở rộng trong thiết kế**: Trong quá trình phát triển, chúng tôi đã suy nghĩ về khả năng mở rộng cho ứng dụng. Điều này không chỉ giúp ứng dụng dễ nâng cấp về sau mà còn giúp nhóm thực hiện dễ dàng hơn khi phát sinh yêu cầu mới.
5. **Quản lý dự án và làm việc nhóm**: Việc chia nhỏ các nhiệm vụ, làm rõ vai trò của từng thành viên và tuân thủ theo tiến độ đã giúp đảm bảo hiệu quả

làm việc của cả nhóm. Nhóm cũng rút ra bài học về sự quan trọng của việc giao tiếp và làm việc phối hợp.

Nhìn chung, dự án này mang lại những kinh nghiệm quý giá trong việc xây dựng một ứng dụng thực tiễn và phát triển kỹ năng lập trình, thiết kế, cũng như quản lý dự án.

# TÀI LIỆU THAM KHẢO


**Tài liệu Python và thư viện Tkinter:**

- https://docs.python.org/3/library/tkinter.html

**Hướng dẫn sử dụng hàm eval() trong Python**:

- https://quantrimang.com/hoc/ham-eval-trong-python-160687

**Hướng dẫn xử lý ngoại lệ trong Python**:

- Tài liệu hướng dẫn về cách xử lý ngoại lệ trong Python với cấu trúc try-except nhằm ngăn chặn các lỗi không mong muốn.
- Link: Python Exception Handling

**Thiết kế giao diện người dùng (UI) với Tkinter**:

- Tài liệu hướng dẫn chi tiết về cách thiết kế giao diện người dùng với các widget trong Tkinter, phù hợp cho việc xây dựng các nút và ô nhập.
- Link: Tkinter Widgets