**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**CẤU TRÚC DỮ LIỆU & GIẢI THUẬT**

**Đề tài: Caculator**

**GVHD : Trần Công Tú**

**SVTH : LÊ MẠNH HÙNG**

**MSSV : 15110058**

**SVTH : NGUYỄN BÁ LÊ AN**

**MSSV : 151101**

**Tp. Hồ Chí Minh, tháng 09 năm 2018**

**MỤC LỤC**

[**Danh mục các hình** 4](#_Toc528655005)

[**Danh mục các bảng** 5](#_Toc528655006)

[**1.** **MÔ TẢ** 6](#_Toc528655007)

[**2.** **XÂY DỰNG THUẬT TOÁN** 7](#_Toc528655008)

[**3.** **THIẾT KẾ GIAO DIỆN** 9](#_Toc528655009)

[**4.** **CHỨC NĂNG CỦA CÁC HÀM** 10](#_Toc528655010)

[**5.** **ƯU ĐIỂM, NHƯỢC ĐIỂM CỦA ĐỀ TÀI** 11](#_Toc528655011)

[**6.** **HƯỚNG PHÁT TRIỂN** 12](#_Toc528655012)

[**7.** **CÀI ĐẶT VÀ KIỂM THỬ** 13](#_Toc528655013)

[**7.1.** **Cài đặt** 13](#_Toc528655014)

[**7.2.** **Kiểm thử** 13](#_Toc528655015)

[**8.** **KẾT LUẬN** 14](#_Toc528655016)

[**9.** **TÀI LIỆU THAM KHẢO** 15](#_Toc528655017)

**Danh mục các hình**

**Danh mục các bảng**

1. **MÔ TẢ**

Ứng dụng Caculator trong thực tế giúp mọi người tính toán các biểu thức đại số nhanh hơn, tiết kiệm thời gian trong việc tính toán.

Ứng dụng Caculator trong lập trình, các biểu thức đại số được sử dụng hằng ngày đều được biểu diễn dưới dạng trung tố (infix). Cách biểu diễn này rất dễ hiểu với con người vì hầu hết các toán tử (+, -, \*, /) đều là toán tử hai ngôi và chúng phân cách giữa hai toán hạng với nhau. Tuy nhiên đối với máy tính, để tính được giá trị của một biểu thức đại số theo dạng này không đơn giản như ta vẫn làm. Để khắc phục điều đó, máy tính cần chuyển cách biểu diễn các biểu thức đại số từ trung tố sang một dạng khác là tiền tố (prefix) hoặc hậu tố (postfix). Trong đồ án này nhóm em sẽ dùng Stack để chuyển biểu thức trung tố sang hậu tố thông qua ngôn ngữ minh họa C#.

1. **XÂY DỰNG THUẬT TOÁN**

Thuật toán để chuyển một biểu thức Infix sang dạng Postfix:

Đọc từng token trong biểu thức infix từ trái qua phải, với mỗi token ta thực hiện các bước sau:

* Nếu là toán hạng: cho ra output.
* Nếu là dấu mở ngoặc “(“: cho vào stack.
* Nếu là dấu đóng ngoặc “)”: lấy các toán tử trong stack ra và cho vào output cho đến khi gặp dấu mở ngoặc “(“. (Dấu mở ngoặc cũng phải được đưa ra khỏi stack).
* Nếu là toán tử:
  + *Chừng nào ở đỉnh stack là toán tử và toán tử đó có độ ưu tiên****lớn hơn hoặc bằng****toán tử hiện tại thì lấy toán tử đó ra khỏi stack và cho ra output.*
  + *Đưa toán tử hiện tại vào stack.*

Sau khi duyệt hết biểu thức infix, nếu trong stack còn phần tử thì lấy các token trong đó ra và cho lần lượt vào output.

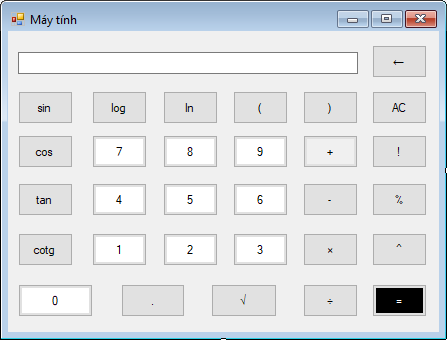
Hãy xem ví dụ sau để hiểu rõ hơn thuật toán này.

Chúng ta sẽ chuyển biểu thức A\*B+C\*((D-E)+F)/G từ dạng Infix sang dạng Postfix:

|  |  |  |
| --- | --- | --- |
| **Token** | **Stack** | **Output** |
| A | {Empty} | A |
| \* | \* | A |
| B | \* | A B |
| + | + | A B \* |
| C | + | A B \* C |
| \* | + \* | A B \* C |
| ( | + \* ( | A B \* C |
| ( | + \* ( ( | A B \* C |
| D | + \* ( ( | A B \* C D |
| – | + \* ( ( – | A B \* C D |
| E | + \* ( ( – | A B \* C D E |
| ) | + \* ( | A B \* C D E – |
| + | + \* ( + | A B \* C D E – |
| F | + \* ( + | A B \* C D E – F |
| ) | + \* | A B \* C D E – F + |
| / | + / | A B \* C D E – F + \* |
| G | + / | A B \* C D E – F + \* G |
|  | {Empty} | A B \* C D E – F + \* G / + |

1. **THIẾT KẾ GIAO DIỆN**

Giao diện gồm 1 form duy nhất:



1. **CHỨC NĂNG CỦA CÁC HÀM**
2. **ƯU ĐIỂM, NHƯỢC ĐIỂM CỦA ĐỀ TÀI**
3. **HƯỚNG PHÁT TRIỂN**
4. **CÀI ĐẶT VÀ KIỂM THỬ**
   1. **Cài đặt**

Ứng dụng Caculator được cài đặt trên Visual Studio 2017, chạy trên Window 10.

* 1. **Kiểm thử**

1. **KẾT LUẬN**
2. **TÀI LIỆU THAM KHẢO**