**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: XÂY DỰNG ỨNG DỤNG CALCULATOR**

**SVTH : LÊ MẠNH HÙNG**

**MSSV : 15110058**

**SVTH : NGUYỄN BÁ LÊ AN**

**MSSV : 15110001**

**Tp. Hồ Chí Minh, tháng 11 năm 2018**

**MỤC LỤC**

[**MỤC LỤC** 2](#_Toc531069876)

[**Danh mục các hình** 3](#_Toc531069877)

[**Danh mục các bảng** 3](#_Toc531069878)

[**1.** **Mô tả** 3](#_Toc531069879)

[**2.** **Mục tiêu** 4](#_Toc531069880)

[**3.** **Thuật toán** 4](#_Toc531069881)

[**4.** **Thiết kế giao diện** 6](#_Toc531069882)

[**5.** **Các hàm chính** 6](#_Toc531069883)

[**6.** **Cài đặt và kiểm thử** 11](#_Toc531069884)

[**6.1.** **Cài đặt** 11](#_Toc531069885)

[**6.2.** **Kiểm thử** 11](#_Toc531069886)

[**7.** **Kết luận** 12](#_Toc531069887)

[**7.1.** **Ưu điểm và nhược điểm** 12](#_Toc531069888)

[**7.2.** **Hướng phát triển** 12](#_Toc531069889)

[**8.** **Tài liệu tham khảo** 12](#_Toc531069890)

**Danh mục các hình**

Hình 1: Form giao diện Calculator.

**Danh mục các bảng**

Bảng 1: Mức độ ưu tiên toán tử.

Bảng 2: Chạy tay thuật toán

**1. Mô tả**

Ứng dụng Calculator 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 Calculator 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. Trong đề tài này nhóm em sẽ ứng dụng Stack để để tính giá trị biểu thức trung tố thông qua ngôn ngữ minh họa C#. Chương trình này thực hiện được các phép tính số nguyên và các phép tính số thực, bao gồm các phép toán sau: phép (+), (-), (\*), (/), (sin), (cos), (tan), (cotg), căn bậc 2, phép chia lấy dư (%).

**2. Mục tiêu**

Mục tiêu cần đạt được trong đề tài xây dựng ứng dụng Calculator là:

* Về giao diện:
* Sử dụng các Toolbox của C# Winform: button, textbox để tạo giao diện cho ứng dụng.
* Nhập được dữ liệu từ bán phím.
* Về thuật toán:
* Ứng dụng được Stack để tính giá trị biểu thức trung tố.
* Thực hiện được biểu thức có chứa các toán tử như sau: +, -, \* , / , sin, cos, tag, cotg, sqrt, %, (, )

**3. Thuật toán**

***3.1. Trình bày thuật toán***

Trước khi đi vào thuật toán, ta phải hiểu thế nào là toán tử, toán hạng và biểu thức trung tố là gì:

* Toán tử là các ký hiệu như: +, -, \*, / , %, (, ),…
* Toán hạng là các phần tử như: a,b,1,2,…
* Biểu thức trung tố (Infix): được biểu diễn bằng cách đặt toán tử giữa hai toán hạng. Ví dụ: 1+3.

Thuật toán tính giá trị biểu thức trung tố (Infix):

* Giả sử S là một biểu thức được cho ở dạng trung tố. Khởi tạo 2 Stack: Sh, St.
* Stack Sh dùng để lưu trữ toán hạng, Stack St dùng để lưu trữ toán tử.
* Duyệt S từ trái sang phải:
* Nếu S[i] là toán hạng, Push vào Sh.
* Nếu S[i] = ’(’, Push vào St.
* Nếu S[i] là toán tử có độ ưu tiên cao hơn toán tử hiện có trên đỉnh St thì Push vào St
* Nếu S[i] là toán tử có độ ưu tiên thấp hơn hoặc bằng toán tử hiện có trên đỉnh St thì Pop toán tử có trên đỉnh St và Pop 2 toán hạng trên đỉnh Sh. Sau đó, thực hiện phép tính, kết quả Push vào Sh.Tiếp theo, Push S[i] vào St.
* Nếu S[i] = ’)’ thì Pop toán tử có trên đỉnh St và Pop 2 toán hạng trên đỉnh Sh. Sau đó, thực hiện phép tính, kết quả Push vào Sh. Tiếp theo, bỏ dấu ‘(‘ gặp phải đầu tiên ra khỏi St.
* Thực hiện đến khi nào St rỗng và Sh còn 1 toán hạng duy nhất thì đó là kết quả.

Mức độ ưu tiên toán tử:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Toán tử | + | - | \* | / | % | sin | cos | tan | cotg | sqrt |
| Mức ưu tiên | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 |

Bảng 1: Mức độ ưu tiên toán tử.

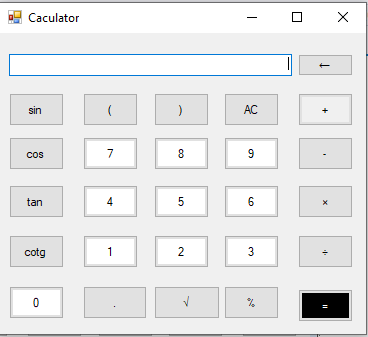
***3.2. Chạy tay thuật toán***

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | Cho M = 5 \* 3 – (3 + 5) | | |
| 2 | Khởi tạo Sh = {} và St = {}. | | |
| 3 | M[0]= "5" -> Đưa vào Sh | Sh={5} | St={} |
| 4 | M[1]= "\*" -> Đưa vào St | Sh={5} | St={\*} |
| 5 | M[2]= "3" -> Đưa vào Sh | Sh={5,3} | St={\*} |
| 6 | M[3]= "-" -> Tính 5\*3=15 rồi lưu vào Sh | Sh={15} | St={-} |
| 7 | M[4]= "(" -> Đưa vào St | Sh={15} | St={-,(} |
| 8 | M[5]= "3" -> Đưa vào Sh | Sh={15,3} | St={-,(} |
| 9 | M[6]= "+" -> Đưa vào St | Sh={15,3} | St={-,(,+} |
| 10 | M[7]= "5" -> Đưa vào Sh | Sh={15,3,5} | St={-,(,+} |
| 11 | M[8]= ")" -> Tính 3+5=8 rồi lưu vào Sh | Sh={15,8} | St={-} |
| 12 | Top(St)="-" -> Tính 15-8=7 rồi lưu vào Sh | Sh={7} | St={} |
| 13 | Dừng. |  |  |

Bảng 2: Chạy tay thuật toán [1]

**4. Thiết kế giao diện**

Giao diện gồm 1 form duy nhất:



Hình 2: Form giao diện Caculator.

**5. Các hàm chính**

Trong đề tài sẽ có những hàm chính sau đây:

|  |
| --- |
| public static int GetPriority(string op)  {  if (op == "sin" || op == "cos" || op == "tan" || op == "cotg" ||op == "sqrt")  return 3;  if (op == "\*" || op == "/" || op == "%")  return 2;  if (op == "+" || op == "-")  return 1;  return 0;  } |

* Nạp sin, cos, tan, cotag vào st

|  |
| --- |
| if (manhinh.Text[i] >= 'a' && manhinh.Text[i] <= 'z')  {  j = i;  while (manhinh.Text[i] >= 'a' && manhinh.Text[i] <= 'z')  i++;  string str = manhinh.Text.Substring(j, i - j);  st.Push((str.ToString()));  } |

* Nạp toán hạng vào sh

|  |
| --- |
| if (manhinh.Text[i] >= '0' && manhinh.Text[i] <= '9')  {  j = i;  while (manhinh.Text[i+1] >= '0' && manhinh.Text[i+1] <= '9' ||manhinh.Text[i+1] =='.' && manhinh.Text[i + 1] != ' ' && i+1 < manhinh.Text.Length)  i++;  string str = manhinh.Text.Substring(j, i + 1-j);  sh.Push((str.ToString()));  } |

* Nạp dấu ( vào st

|  |
| --- |
| if (manhinh.Text[i] == '(')  {  st.Push(manhinh.Text[i]);  if (manhinh.Text[i + 1] == '-' || manhinh.Text[i + 1] == '+')  sh.Push(0);  } |

* Nạp toán tử vào st

|  |
| --- |
| if (manhinh.Text[i] == '+' || manhinh.Text[i] == '-' || manhinh.Text[i] == '\*' || manhinh.Text[i] == '/' || manhinh.Text[i] == '%' || manhinh.Text[i] == '^')  {  while (st.Count > 0 && GetPriority(manhinh.Text[i].ToString()) <= GetPriority(st.Peek().ToString()))  {  str1 = st.Pop().ToString();  //nếu là các phép tính  if (str1 == "+" || str1 == "-" || str1 == "\*" || str1 == "/" || str1 == "%" || str1 == "^")  {  a = double.Parse((sh.Pop().ToString()));  b = double.Parse((sh.Pop().ToString()));  switch (str1)  {  case "+":  ketQua = (a + b);  break;  case "-":  ketQua = (b - a);  break;  case "\*":  ketQua = (a \* b);  break;  case "/":  if (a == 0)  {  MessageBox.Show("Không thể chia cho 0", "Thông báo");  }  ketQua = (b / a);  break;  case "%":  if (a == 0)  {  MessageBox.Show("Không thể chia cho 0", "Thông báo");  }  ketQua = (b % a);  break;  case "^":  ketQua = Math.Pow(b, a);  break;  }  sh.Push(ketQua);  } |

* xử lí nhiều dấu + hoặc - hoặc + - hoặc - + liên tiếp(số âm)

|  |
| --- |
| if (manhinh.Text[i] == '-' && manhinh.Text[i + 1] == '-' || manhinh.Text[i] == '-' && manhinh.Text[i + 1] == '+' || manhinh.Text[i] == '+' && manhinh.Text[i + 1] == '+'|| manhinh.Text[i] == '+' && manhinh.Text[i + 1] == '-')  {  if (manhinh.Text[i] == '-')  j = 1;  while (manhinh.Text[i + 1] == '-' || manhinh.Text[i + 1] == '+')  {  if (manhinh.Text[i + 1] == '-')  j++;  i++;  }  if (( j) % 2 == 0)  str1 = "+";  else  str1 = "-";  st.Push((str1.ToString()));  } |

* Xử lí các phép toán trong dấu ) ngoặc

|  |
| --- |
| if (manhinh.Text[i] == ')')  {  str1 = "+"; // khai báo str1 ảo(tránh stack rỗng)  while (str1 != "(")  {    a = double.Parse((sh.Pop().ToString()));  str1 = st.Pop().ToString();  if (str1 == "(")  {  sh.Push(a);  break;  }  if (sh.Count == 0 && (str1 == "+" || str1 == "-")) b = 0;  else  if (sh.Count == 0 && (str1 == "\*" || str1 == "/"))  {  MessageBox.Show("Loi!");  break;  }  else  b = double.Parse((sh.Pop().ToString()));  switch (str1)  {  case "+":  ketQua = (a + b);  break;  case "-":  ketQua = (b - a);  break;  case "\*":  ketQua = (a \* b);  break;  case "/":  if (a == 0)  {  MessageBox.Show("Không thể chia cho 0", "Thông báo");  }  ketQua = (b / a);  break;  case "%":  if (a == 0)  {  MessageBox.Show("Không thể chia cho 0", "Thông báo");  }  ketQua = (b % a);  break;  }  sh.Push(ketQua);  if (str1 != "(")  str1 = st.Pop().ToString();  }  if (manhinh.Text[i] == ')' && (manhinh.Text[i] >= '0' && manhinh.Text[i] <= '9'))  st.Push("\*");    } |

* Xử lí các phép toán còn lại

|  |
| --- |
| while (st.Count > 0)  {  a = double.Parse(sh.Pop().ToString()); ;  str1 = st.Pop().ToString();  if (sh.Count == 0 && (str1 == "+" || str1 == "-")) b = 0;  else  if (sh.Count == 0 && (str1 == "\*" || str1 == "/"))  {  MessageBox.Show("Loi!");  manhinh.Text = "0";  return;  }    if (str1 == "+" || str1 == "-" || str1 == "\*" || str1 == "/" || str1 == "%")  {  b = double.Parse((sh.Pop().ToString()));  switch (str1)  {  case "+":  ketQua = (a + b);  break;  case "-":  ketQua = (b - a);  break;  case "\*":  ketQua = (a \* b);  break;  case "/":  if (a == 0)  {  MessageBox.Show("Không thể chia cho 0", "Thông báo");  }  ketQua = (b / a);  break;  case "%":  if (a == 0)  {  MessageBox.Show("Không thể chia cho 0", "Thông báo");  }  ketQua = (b % a);  break;  case "^":  ketQua = Math.Pow(b, a);  break;  }  sh.Push(ketQua);  }  else  if (str1 == "sin" || str1 == "cos" || str1 == "tan" || str1 == "cotg" || str1 =="sqrt")  {  switch (str1)  {  case "sin":  sh.Push(Math.Sin(a));  break;  case "cos":  sh.Push(Math.Cos(a));  break;  case "tan":  sh.Push(Math.Tan(a));  break;  case "cotg":  sh.Push(1.00/Math.Tan(a));  break;  case "sqrt":  sh.Push(Math.Sqrt(a));  break;  }  }  }  manhinh.Text = sh.Pop().ToString(); |

**6. Cài đặt và kiểm thử**

***6.1. Cài đặt***

Ứng dụng Caculator được cài đặt trên Visual Studio 2017, chạy trên Window10 x64 bit.

***6.2. Kiểm thử***

Các trường hợp đã thử:

TH1: 5 \* 3 – (3 + 5)/2 = 11

TH2: 9%4 dư 1

TH3: cos1 + 2 = 2.54

TH4: 5 - - 1 = 6

**7. Kết luận**

Nhóm đã hoàn được 90% mục tiêu đã đề ra.

***7.1. Ưu điểm và nhược điểm***

**7.1.1. Ưu điểm**

* Xử lý được nhiều phép toán một lúc, giúp giảm thời gian tính toán của người sử dụng.

**7.1.2. Nhược điểm**

* Chưa xử lý được nhập dữ liệu từ bàn phím
* Chưa có các phép toán như: tính giai thừa, logarit, giải phương trình bậc 1, bậc 2, phân số, hàm mũ.

***7.2. Hướng phát triển***

Hướng phát triển của đồ án Calculator là:

* Cần bổ sung thêm chức năng nhập dữ liệu từ bán phím.
* Thêm chức năng quay lại phép tính trước.
* Thêm chức năng tính toán như: tính giai thừa, logarit, giải phương trình bậc 1, bậc 2, phân số, lũy thừa, hàm mũ.

**8. Tài liệu tham khảo**

[1] https://www.stdio.vn/articles/ung-dung-stack-bieu-thuc-trung-to-infix-470