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

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

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



**BÁO CÁO ĐỒ ÁN**

**CẤU TRÚC DỮ LIỆU & GIẢI THUẬT**

**SVTH : Ngô Thanh Tài**

**MSSV : 16110201**

**SVTH : Trần Tuấn Kiệt**

**MSSV : 16110131**

**TP. Hồ Chí Minh, tháng 12 năm 2017**

# MỤC LỤC

[MỤC LỤC 3](#_Toc500714979)

[DANH MỤC CÁC HÌNH 4](#_Toc500714980)

[NỘI DUNG 5](#_Toc500714981)

[1. MÔ TẢ PROJECT 5](#_Toc500714982)

[Sản phẩm đồ án: 5](#_Toc500714983)

[Phần mềm dùng để: 5](#_Toc500714984)

[2. MÔ TẢ QUÁ TRÌNH LÀM 5](#_Toc500714985)

[Cách thức xây dựng phần mềm: 5](#_Toc500714986)

[Thiết kế giao diện: 5](#_Toc500714987)

[Thiết kế lớp: 6](#_Toc500714988)

[Mô tả phương thức của mỗi lớp: 6](#_Toc500714989)

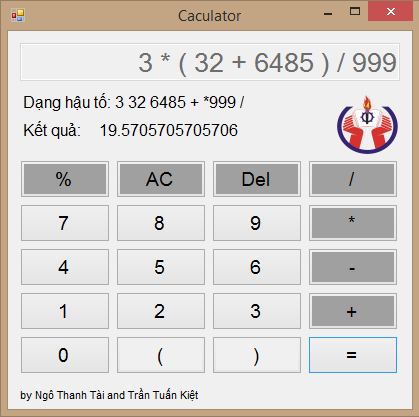
[Lớp Node.cs: 6](#_Toc500714990)

[Lớp Stack.cs: 8](#_Toc500714991)

Lớp Calculator.cs………………………………………………………………….8

[TÀI LIỆU THAM KHẢO 9](#_Toc500714992)

# DANH MỤC CÁC MÀN HÌNH



# 

# NỘI DUNG

## MÔ TẢ PROJECT

### Sản phẩm đồ án:

Làm ứng dụng Calculator ứng dụng Stack.

### Phần mềm dùng để:

Giúp người dùng tính toán và cho biết dạng hậu tố của các dãy phép tính.

## MÔ TẢ QUÁ TRÌNH LÀM

### Cách thức xây dựng phần mềm:

Ứng dụng các giải thuật liên quan đến Node và Stack để xử lý 1 chuỗi các phép tính đúng thành dạng hậu tố sau đó tính kết quả.

### Thiết kế giao diện:

|  |  |  |  |
| --- | --- | --- | --- |
| TT | (Tên màn hình trên 1 dòng; dưới dòng này là hinh chụp màn hình) | Người thiết kế & giải thích ngắn gọn các quyết định thiết kế khi thiết kế màn hình (Người thiết kế ở trên 1 dòng. Sau dòng đó là phần giải thích. | Mục đích chính của màn hình |
| 1 |  | Ngô Thanh Tài  Giải thích thiết kế:  Thiết kế đơn giản, gồm các nút số và phép tính cơ bản kèm theo các nút hỗ trợ các thao tác xoá và tính phần trăm và textbox để hiện dãy phép tính. | Hiển thị dãy phép tính người dùng đã nhập cũng như kết quả và dạng hậu tố của nó. Ngoài ra còn hỗ trợ người dùng 2 phương thức nhập trực tiếp từ bàn phím hoặc click vào các button tương ứng. |

### Thiết kế lớp:

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Tên lớp | Tên các SV phụ trách viết  (Nếu là code lấy từ ngồn khác thì phải chỉ rõ lấy từ ngồn nào). | Mục đích chính của lớp trong chương trình |
| 1 | Node.cs | Trần Tuấn Kiệt | Khởi tạo, khai báo cấu trúc 1 node của Stack và các hàm liên quan. |
| 2 | Stack.cs | Trần Tuấn Kiệt | Khởi tạo, khai báo cấu trúc của 1 Stack và các hàm liên quan khác. |
| 3 | Calculator.cs | Ngô Thanh Tài | Chứa các hàm xử lí chuỗi phép tính đưa vào.  Hàm hauTo dùng để chuyển chuỗi phép tính sang dạng hậu tố.  Hàm Calculate dùng để tính kết quả từ dạng hậu tố của phép tính. |

### Mô tả phương thức của mỗi lớp:

#### Lớp Node.cs:

class Node

{

String info;

Node pNext;

public Node()

{

info = "";

pNext = null;

}

public void createNode(string s)

{

info = s;

pNext = null;

}

public void Next(Node x)//dung trong push

{

pNext = x;

}

public string getInfo()

{

return info;

}

public Node getNext()

{

return pNext;

}

}

#### 

#### Lớp Stack.cs:

class Stack

{

Node pTop;

public Stack()

{

pTop = new Node();

pTop.createNode("#");

}

public void Push(string s)

{

Node p = new Node();

p.createNode(s);

if (pTop == null)

pTop = p;

else

{

p.Next(pTop);//pNext cua p tro den pTop

pTop = p;

}

}

public string Pop()

{

if (pTop.getInfo() != "#")

{

string s;

s = pTop.getInfo();

pTop = pTop.getNext();

return s;

}

else return "#";

}

public string getTop()

{

return pTop.getInfo();

}

}

#### Lớp Calculator.cs:

private void hauTo(List<string> IN,List<String>OUT)

{

string x;

for (int i = 0; IN[i] != "\0"; i++)

{

if (isNumb(IN[i]) == 1)

{

OUT.Add(IN[i]);

}

else

if (IN[i] == "(")

S.Push("(");

else

if (IN[i] == ")")

{

while (isEmpty() == 0)//

{

x = S.Pop();

if (x != "(")

{

OUT.Add(x);

}

else break;

}

}

else

{

while (isEmpty() == 0 && level(IN[i]) <= level(S.getTop()))

{

OUT.Add(S.Pop());

}

S.Push(IN[i]);

}

}

while (isEmpty() == 0)

{

OUT.Add(S.Pop());

}

OUT.Add("\0");

}

string calculate(List<string> OUT)

{

double x1, x2;

double kq;

int i = 0;

while (i < OUT.Count && OUT[i] != "\0")

{

while (OUT[i] == " ")

{

i++;

}

if (isNumb(OUT[i]) == 1)

{

double num = 0;

while (isNumb(OUT[i]) == 1)

{

num = num \* 10 + Convert.ToDouble(OUT[i]) ;

i++;

}

S.Push(Convert.ToString((num)));

}

else

{

x1 = Convert.ToDouble(Convert.ToDouble(S.Pop()));

x2 = Convert.ToDouble(Convert.ToDouble(S.Pop()));

switch (OUT[i])

{

case "+":

kq = x1 + x2;

break;

case "-":

kq = x2 - x1;

break;

case "\*":

kq = x1 \* x2;

break;

case "/":

kq = x2 / x1;

break;

case "%":

kq = x2 % x1;

break;

default:

return "0";

}

S.Push(Convert.ToString(kq));

}

i++;

}

return S.Pop();

}

TÀI LIỆU THAM KHẢO

-Internet

-Tập vở