# PROJECT 2

# STACKS & EXPRESSIONS

**CSCI 220**
**DATA STRUCTURE 1**

**MAI PHAM**

**DEVELOPMENT ENVIRONMENT**
**MacOS – Xcode**
**Window 10 – MSVS 2017**

**TABLE OF CONTENTS**

# PROJECT NOTE

**OBJECTIVE:**

- Create a calculator class that convert infix expression to postfix expression, evaluate the postfix expression, and print out the result using a template stack class.

**SUMMARY:**

- There are many areas that I got struck with. The two most difficult parts are convert the infix to postfix and be able to evaluate the expression with multiply digits. There are two things I learned from this project, which are creating a class template and how to convert and evaluate the expression manually.

**EXTRA CREDIT:**

- I did extra credit 1. At first, I couldn't get the validation to work. Then I used the stack which help make it so much easier.

**CONCLUSION:**

- Overall, my project is completed and successfully include the main and the extra credit 1 requirement.

# OUTPUT

```
Project 2: Stacks & Expressions
Author: Mai Pham

Enter an arithmetic expression: 17 / ( 2 + 3 ) - 13
The infix expression is: 17 / ( 2 + 3 ) - 13
The postfix expression is: 17 2 3 + / 13 -
The answer is: -10

Enter an arithmetic expression: 5*2^3
The infix expression is: 5*2^3
The postfix expression is: 5 2 3 ^ *
The answer is: 40

Enter an arithmetic expression: 5*(2+3)  - 13
The infix expression is: 5*(2+3)  - 13
The postfix expression is: 5 2 3 + * 13 -
The answer is: 12

Enter an arithmetic expression: (5+2)/ ( 7 -  3)
The infix expression is: (5+2)/ ( 7 -  3)
The postfix expression is: 5 2 + 7 3 - /
The answer is: 1
```

```
Enter an arithmetic expression: (5+2)*3
The infix expression is: (5+2)*3
The postfix expression is: 5 2 + 3 *
The answer is: 21

Enter an arithmetic expression: 5 + 2 * 3
The infix expression is: 5 + 2 * 3
The postfix expression is: 5 2 3 * +
The answer is: 11

Enter an arithmetic expression: 5*(2+3
The infix expression is: 5*(2+3
The expression is invalid.

Enter an arithmetic expression: 5*2+3)
The infix expression is: 5*2+3)
The expression is invalid.

Enter an arithmetic expression: 5$4
The infix expression is: 5$4
The expression is invalid.

Enter an arithmetic expression: 0
Thank you for using the program.
Program ended with exit code: 0
```

# SOURCE CODE

## STACK TEMPLATE

```
//
//  Stack.h
//  Project 2
//
//  Created by Mai Pham on 10/7/17.
//  Copyright © 2017 Mai Pham. All rights reserved.
//

#ifndef Header_h
#define Header_h

#include <iostream>
#include <string>
using namespace std;

template <typename T>
class Stack
{
private:
    T arr[20];
    int current;
public:
    Stack()
```

```
    {
        current = 0;
    }
    void push(T t)
    {
        arr[current] = t;
        current++;
    }
    void pop()
    {
        current--;
    }
    T top()
    {
        return arr[current-1];
    }
    int size()
    {
        return current;
    }
    bool empty()
    {
        if(current == 0)
            return true;
        return false;
    }
    bool full()
    {
        if(current == 20)
            return true;
        return false;
    }
};
#endif /* Header_h */
```

## CALCULATOR
### Header File

```
//
//  Calculator.hpp
//  Project 2
//
//  Created by Mai Pham on 10/7/17.
//  Copyright © 2017 Mai Pham. All rights reserved.
//

#ifndef Calculator_hpp
#define Calculator_hpp

#include "Stack.h"
#include <iostream>
#include <string>
using namespace std;

class Calculator
{
private:
    string infix;
    string postfix;
    int answer;
public:
    Calculator(string exp);
    bool validate();
    void convert();
    void evaluate();
```

```
    int check(char c);
    string inFix();
    string postFix();
    int result();
};
#endif /* Calculator_hpp */
```

## Implementation File

```
//
//  Calculator.cpp
//  Project 2
//
//  Created by Mai Pham on 10/7/17.
//  Copyright © 2017 Mai Pham. All rights reserved.
//

#include "Calculator.h"
#include "Stack.h"
#include <iostream>
#include <string>
#include <math.h>
#include <sstream>
using namespace std;

Calculator::Calculator(string exp)
{
    infix = exp;
    postfix = "";
    answer = 0;
}
bool Calculator::validate()
{
    Stack<char> s;

    for (int i = 0; i < infix.length(); i++)
    {
        if (infix[i] == ' ')
            continue;
        if(infix[i] >= '0' && infix[i] <= '9')
            continue;
        if (check(infix[i]) > 0)
            continue;
        if (infix[i] == '(')
            s.push(infix[i]);
        else if (infix[i] == ')')
        {
            if (s.top() == '(')
                s.pop();
            else
                s.push(infix[i]);
        }
        else
            s.push(infix[i]);
    }
    if (s.empty())
        return true;
    else
        return false;
}
void Calculator::convert()
{
    Stack<char> s;

    for (int i = 0; i < infix.length(); i++)
```

```
    {
        if (infix[i] == ' ')
            continue;
        else if(infix[i] >= '0' && infix[i] <= '9')
        {
            postfix += infix[i];
            while(infix[i+1] >= '0' && infix[i+1] <= '9')
            {
                postfix += infix[i+1];
                i++;
            }
            postfix += ' ';
        }
        else if (infix[i] == '(')
            s.push(infix[i]);
        else if (infix[i] == ')')
        {
            while (!s.empty() && s.top() != '(')
            {
                postfix += s.top();
                postfix += ' ';
                s.pop();
            }
            s.pop();
        }
        else
        {
            while(!s.empty() && check(infix[i]) <= check(s.top()))
            {
                postfix += s.top();
                postfix += ' ';
                s.pop();
            }
            s.push(infix[i]);
        }
    }
    while (!s.empty())
    {
        postfix += s.top();
        postfix += ' ';
        s.pop();
    }
}
void Calculator::evaluate()
{
    Stack<int> s;

    istringstream stream(postfix);
    string n;
    int m;

    while(stream >> n)
    {
        if (n == " ")
            continue;
        else if (isdigit(n[0]))
        {
            m = stoi(n);
            s.push(m);
        }
        else
        {
            int value2 = s.top();
            s.pop();
            int value1 = s.top();
```

```
            s.pop();
            switch (n[0])
            {
                case '+':
                    s.push(value1 + value2);
                    break;
                case '-':
                    s.push(value1 - value2);
                    break;
                case '*':
                    s.push(value1 * value2);
                    break;
                case '/':
                    s.push(value1 / value2);
                    break;
                case '%':
                    s.push(value1 % value2);
                    break;
                case '^':
                    s.push(pow(value1, value2));
                    break;
                default:
                    break;
            }
        }
    }
    answer = s.top();
}
int Calculator::check(char c)
{
    switch (c)
    {
        case '+':
        case '-':
            return 1;

        case '*':
        case '/':
        case '%':
            return 2;

        case '^':
            return 3;
    }
    return -1;
}
string Calculator::inFix()
{
    return infix;
}
string Calculator::postFix()
{
    return postfix;
}
int Calculator::result()
{
    return answer;
}
```

**MAIN**

```
//
//  main2.cpp
//  Project 2
//
```

```
//  Created by Mai Pham on 10/7/17.
//  Copyright © 2017 Mai Pham. All rights reserved.
//

#include "Calculator.h"
#include "Stack.h"
#include <iostream>
#include <string>
using namespace std;

int main()
{
    cout << "Project 2: Stacks & Expressions\n";
    cout << "Author: Mai Pham\n\n";
    string expression;

    cout << "Enter an arithmetic expression: ";
    getline(cin,expression);

    while (expression != "0")
    {
        cout << "The infix expression is: " << expression << endl;
        Calculator cal(expression);
        if (cal.validate() == true)
        {
            cal.convert();
            cal.evaluate();
            cout << "The postfix expression is: " << cal.postFix() << endl;
            cout << "The answer is: " << cal.result() << endl << endl;
        }
        else
            cout << "The expression is invalid." << endl << endl;

        cout << "Enter an arithmetic expression: ";
        getline(cin,expression);
    }
    cout << "Thank you for using the program." << endl;
    return 0;
}
```