Name: Mohamed Khaled Gamil Ismail

Section: 3

Seat No.: 33813

Data Structure

Assignment 1

# Supported Operators:

Binary: +, -, \*, /, ^

Unary: ~(unary negate), sin, cos, sqrt

# Code:

#include <iostream>

#include <string>

#include <sstream>

#include <stack>

#include <windows.h>

using namespace std;

HANDLE hConsole;

bool is\_oper(int size, string str[], string s);

bool contain\_oper(int size, string str[], string s);

void print\_error();

int main()

{

system("COLOR F0");

hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

string binary\_oper[] = { "^","\*", "/", "+", "-" };

string uniary\_oper[] = { "sin","cos", "sqrt", "~" };

stack<string> stack;

string expr;

cout << "Enter postfix expression with spaces between operands and operators: " << endl;

getline(cin,expr);

istringstream is(expr);

string term;

while (getline(is, term, ' '))

{

if ((is\_oper(5, binary\_oper, term) || is\_oper(4, uniary\_oper, term)) && stack.empty())

{

print\_error();

return 1;

}

else if (!(is\_oper(5, binary\_oper, term) || is\_oper(4, uniary\_oper, term)))

{

stack.push(term);

}

else if (is\_oper(5, binary\_oper, term))

{

if (stack.empty())

{

print\_error();

return 1;

}

string op2 = stack.top();

stack.pop();

if (stack.empty())

{

print\_error();

return 1;

}

string op1 = stack.top();

stack.pop();

if (contain\_oper(5, binary\_oper, op1))

{

op1 = "(" + op1 + ")";

}

if (contain\_oper(5, binary\_oper, op2))

{

op2 = "(" + op2 + ")";

}

string newOp = op1 + " " + term + " " + op2;

stack.push(newOp);

}

else if (is\_oper(4, uniary\_oper, term))

{

if (stack.empty())

{

print\_error();

return 1;

}

string op = stack.top();

stack.pop();

string newOp;

if (term == "~")

{

if (contain\_oper(5, binary\_oper, op))

{

op = "(" + op + ")";

}

newOp = term + op;

}

else

{

newOp = term + "(" + op + ")";

}

stack.push(newOp);

}

}

string infix = stack.top();

if (!(contain\_oper(5, binary\_oper, infix) || contain\_oper(4, uniary\_oper, infix)))

{

print\_error();

return 1;

}

SetConsoleTextAttribute(hConsole, 250);

cout << "Infix expression: " << infix << endl;

SetConsoleTextAttribute(hConsole, 240);

return 0;

}

bool is\_oper(int size, string str[], string s)

{

for (int i = 0; i < size; i++)

{

if (str[i] == s) return true;

}

return false;

}

bool contain\_oper(int size, string str[], string s)

{

for (int i = 0; i < size; i++)

{

if (s.find(str[i]) != string::npos) return true;

}

return false;

}

void print\_error()

{

SetConsoleTextAttribute(hConsole, 252);

cout << "Invalid postfix expression" << endl;

SetConsoleTextAttribute(hConsole, 240);

}

# Screenshots:

