**Практическое занятие №1 (Анализ сложности алгоритмов)**

Язык: C#  
Цель задания – познакомиться со структурой данных Стек.

**Задание 1.**  
public class ListStack

{

List<string> list = new List<string>();

public void Push(string value)

{

list.Add(value);

}

public string Pop()

{

if (!isEmpty())

{

var result = list[list.Count - 1];

list.RemoveAt(list.Count - 1);

return result;

}

else return "Стек пуст, нечего вытаскивать";

}

public bool isEmpty()

{

return list.Count == 0;

}

public string Top()

{

if (!isEmpty())

return list[list.Count - 1];

else return "Стек пуст, нет верхнего элемента";

}

public void Print()

{

Console.Write("В стеке: ");

foreach (var element in list)

Console.Write(element + " ");

Console.WriteLine();

}

}

**Задание 2.**

ListStack stack = new ListStack();

string operations = File.ReadAllText("input.txt");

for (int i = 0; i < operations.Length; i++)

{

char symb = operations[i];

switch (symb)

{

case '1':

string element = "";

for (int j = i + 2; j < operations.Length; j++)

{

if (char.IsWhiteSpace(operations[j]))

break;

element += operations[j];

}

stack.Push(element);

Console.WriteLine("команда Push(" + element + ") выполнена");

break;

case '2': Console.WriteLine("команда Pop: " + stack.Pop()); break;

case '3': Console.WriteLine("команда Top: " + stack.Top()); break;

case '4':

Console.WriteLine("команда IsEmpty: " + (stack.isEmpty() ?

"Стек пуст" : "Стек не пуст"));

break;

case '5': stack.Print(); break;

}

}

**Задание 3.**

|  |  |  |
| --- | --- | --- |
| Входная строка | Время на считывание (в мс) | Время на выполнение(в мс) |
| 5 1,PrI 2 | 1,0002 | 24,0915 |
| 1,555 1,oikuj 4 5 | 0,7232 | 23,1263 |
| 4 5 1,7 5 2 | 0,7161 | 17,7408 |
| 5 4 2 3 1,oikj 1,52 | 1,4416 | 25,2615 |
| 3 4 1,56 1,7 1,cat 2 5 | 0,8505 | 57,0767 |
| 1,fghj 1,io 2 3 5 4 1,852 2 | 0,6586 | 28,6358 |
| 5 2 4 1,frr 1,rr 3 4 2 5 | 0,8965 | 33,4624 |
| 1,55 5 4 2 1,hj 4 2 1,1 3 | 0,6176 | 25,2494 |
| 5 2 3 4 2 5 4 3 1,fghj 1,852 5 3 | 2,8865 | 51,3378 |
| 1,fgh 1,852 2 3 5 4 2 1,fg 1,85 2 3 4 5 | 0,7494 | 81,9554 |

**Задание 4**  
Сложность: *О(n).*

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace AlgLab2

{

class Program

{

public static int GetPriority(string op)

{

if (op == "^" || op == "cos" || op == "sin" || op == "ln" || op == "sqrt")

return 1;

else if (op == "\*" || op == ":" || op == "/")

return 2;

else return 3;

}

public static string TranslateInfixToPostfix(StreamReader inputFile, int length, string[] allSymbol,

string[] operations)

{

Stack<string> stack = new Stack<string>();

string output = "";

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

{

if (operations.Any(operation => operation == allSymbol[i]))

{

if (stack.Count == 0)

stack.Push(allSymbol[i]);

else

{

int pr1 = GetPriority(stack.Peek());

int pr2 = GetPriority(allSymbol[i]);

if (pr1 > pr2)

stack.Push(allSymbol[i]);

else if (stack.Peek() != "(" && stack.Peek() != ")")

{

output = output + stack.Pop() + ' ';

stack.Push(allSymbol[i]);

}

else

stack.Pop();

}

}

else if (allSymbol[i] == "(")

{

i++;

var list = new List<string>();

while (allSymbol[i] != ")")

{

list.Add(allSymbol[i]);

i++;

}

output += TranslateInfixToPostfix(inputFile, list.Count, list.ToArray(), operations);

}

else

{

if (allSymbol[i].ToCharArray().All(char.IsDigit))

output = output + allSymbol[i] + ' ';

else

{

string line = inputFile.ReadLine();

string[] l = line.Split(new char[] { ' ' },

StringSplitOptions.RemoveEmptyEntries);

output = output + l[l.Length - 1] + ' ';

}

}

}

while (stack.Count != 0)

{

if (stack.Peek() != "(" && stack.Peek() != ")")

output = output + stack.Pop() + ' ';

else

stack.Pop();

}

return output;

}

public static int HowMachNumber(string op)

{

if (op == "ln" || op == "cos" || op == "sin" || op == "sqrt")

return 1;

else return 2;

}

public static double CalculateTheValue(string input, string[] operations)

{

Stack<double> stack = new Stack<double>();

double number = 0;

double output = 0;

string[] allSymbol = input.Split(new char[] { ' ' },

StringSplitOptions.RemoveEmptyEntries);

int lengthInputString = allSymbol.Length;

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

{

if (operations.Any(operation => operation == allSymbol[i]))

{

if (HowMachNumber(allSymbol[i]) == 1)

{

number = stack.Pop();

if (allSymbol[i] == "sin")

{

number = Math.Sin((number \* Math.PI) / 180);

stack.Push(number);

}

else if (allSymbol[i] == "cos")

{

number = Math.Cos((number \* Math.PI) / 180);

stack.Push(number);

}

else if (allSymbol[i] == "sqrt")

{

number = Math.Sqrt(number);

stack.Push(number);

}

else

{

number = Math.Log(number);

stack.Push(number);

}

output += number;

}

else

{

double first = stack.Pop();

double second = stack.Pop();

if (allSymbol[i] == "^")

number = Math.Pow(second, first);

else if (allSymbol[i] == "\*")

{

number = second \* first;

stack.Push(number);

}

else if (allSymbol[i] == ":" || allSymbol[i] == "/")

{

number = second / first;

stack.Push(number);

}

else if (allSymbol[i] == "+")

{

number = second + first;

stack.Push(number);

}

else

{

number = second - first;

stack.Push(number);

}

}

}

else stack.Push(Convert.ToDouble(allSymbol[i]));

}

output = stack.Pop();

return output;

}

static void Main()

{

StreamReader inputFile = new StreamReader("input.txt");

Stack<string> stack = new Stack<string>();

string[] operations = { "ln", "cos", "sin", "sqrt", "^", "+", "-", "\*", ":", "/" };

string[] allSymbols = inputFile.ReadLine().Split(new char[] { ' ' },

StringSplitOptions.RemoveEmptyEntries);

int lengthString = allSymbols.Length;

string postFix = TranslateInfixToPostfix(inputFile, lengthString, allSymbols, operations);

Console.WriteLine(postFix);

Console.WriteLine(CalculateTheValue(postFix, operations));

}

}

}

**Расчет сложности алгоритма:**

Определим сложность данного алгоритма путем подсчета инструкций.

В самом начале имеем несколько элементарных операций, они занимают константное время. Получим величину C1 - какая-то константа.

Далее видим операцию Split, которая зависит от длины строки, разбиение которой мы выполняем. Длина строки – n.  
На данном этапе получаем n+C1.

Дальше у нас цикл от 0 до n (длины строки), в каждой итерации цикла у нас выполняется сколько-то элементарных операций, следовательно, получим n\*C2.

Итого сложность равна n+C1+n\*C2.

По правилу вычисления сложности константами можно пренебречь, т.к. если n стремиться к бесконечности, то величины C будут пренебрежительно малы.

Итого сложность будет равна O(n).