Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

(ФГАОУ ВО «ЮФУ»)

Институт компьютерных технологий и информационной безопасности

Кафедра системного анализа и телекоммуникаций

**ПРОВЕРИЛ** **ВЫПОЛНИЛ**

Лапшин В. С. МастерскихА.С

« » февраля 2024 г. « » 2024 г.

**ЛАБОРАТОРНАЯ РАБОТА №7**

по дисциплине

«Алгоритмизация и программирование»

на тему

«Работа со стеком»

Оглавление

[Техническое задание 3](#_Toc159270881)

[Выполнение работы 4](#_Toc159270882)

[Результаты работы программы 5](#_Toc159270883)

[Вывод 6](#_Toc159270884)

[Листинг программы 7](#_Toc159270885)

# 

# Техническое задание

**Цель работы**

Целью данной лабораторной работы является выработка практических навыков работы со стеком на примере построения обратной польской записи арифметических и логических выражений.

**Задание**

Сделать программу, которая будет преобразовывать данное инфиксное выражение в формат обратной польской записи при помощи алгоритма Дейкстры.

Примеры выражений, которые нужно преобразовать:

1. 5 + 7–5 + 3 / (4–5 \* 32) + 1
2. 1 + 8 \* 4 / 4 - (4 + 3) + (54–3)
3. 3 \* (32–1) / 5 - (4–1)
4. 5–4 + 1 \* 3–7 / (4–1) \* (6 + 22)
5. 7–4 + (6–5) \* 5–4 + 1
6. 8 / 5–44 \* (6 + 4–3) + 43
7. 7 + 4 \* 3–1 \* (4 / 9–1 + 7) \* 4
8. 5 + 1–5 / 8 - (5 \* 8 + 6 / 93)
9. 7 + 4–3 + 1 \* (4 / 5 + 1) \* 2–54
10. 1 + 5 / 4 \* (3 + 1) \* 5 - 8 \* (42 - 33)

# Выполнение работы

Изображение выглядит как текст, диаграмма, План, зарисовка

Автоматически созданное описание

# Результаты работы программы

Input:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Output:

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

Проверка результата:

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

# Вывод

Проделав данную лабораторную работу, я узнал:

1) что такое алгоритм Дейкстры

2) что такое Обратная Польская Запись

3) что такое стек, как с ним работать, функции стека

# Листинг программы

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int const MAX = 100;

int stack[MAX];

int tos = 0;

void push(int i) {

if (tos >= MAX) {

cout << "stack is full";

return;

}

stack[tos] = i;

tos++;

}

int pop() {

tos--;

if (tos < 0) {

cout << "stack is empty";

return 0;

}

return stack[tos];

}

int priority(char pr) {

switch (pr) {

case '+':

return 1;

case '-':

return 1;

case '\*':

return 2;

case '/':

return 2;

case '^':

return 3;

case '&':

return 4;

case '|':

return 4;

case '!':

return 5;

default:

return 0;

}

}

string RPE(const string& strok) {

string OPZ;

for (char i : strok) {

if (i == ' ') {

continue;

}

if (isalnum(i) || i == ' ') { // Если символ - операнд или пробел

if (i != ' ') {

OPZ += i;

}

}

else if (i == '(') { // Если символ - открывающая скобка

push(i);

}

else if (i == ')') { // Если символ - закрывающая скобка

while (tos != -1 && stack[tos] != '(') {

OPZ += stack[tos];

pop();

}

pop(); // Удаляем открывающую скобку из стека

tos += 1;

}

else { // Если символ - оператор

if (isalnum(i) || i == ' ') { // Если символ - операнд или пробел

if (i != ' ') {

OPZ += i;

OPZ += ' '; // Добавляем пробел после операнда

}

}

else { // Если символ - оператор

if (tos == 0 || stack[tos - 1] == '(' || priority(stack[tos - 1]) < priority(i)) {

push(i);

}

else {

while (tos != 0 && priority(stack[tos - 1]) >= priority(i)) {

OPZ += stack[tos - 1]; // Добавляем оператор из стека в ОПЗ

OPZ += ' '; // Добавляем пробел после оператора

pop(); // Удаляем его из стека

}

push(i); // Помещаем текущий оператор в стек

}

}

}

}

while (tos != 0) {

OPZ += " "; // Добавляем пробел

OPZ += stack[tos - 1]; // Добавляем оператор или операнд

pop(); // Удаляем его из стека

}

return OPZ;

}

int main()

{

ifstream fin;

fin.open("inputData.txt");

ofstream fout;

fout.open("outputData.txt");

if (!fin.is\_open()) {

cout << "unfind file!";

}

else {

string strok;

getline(fin, strok);

string opz = RPE(strok);

fout << opz << '\n';

return 0;

}

fout.close();

fin.close();

}