Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления
Кафедра — интеллектуальных информационных технологий

К защите допустить:

Заведующий кафедрой ИИТ
Д.В. Шункевич

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к лабораторной работе
на тему:

Обратная польская запись

Студент гр. 321701 А. С. Астахов
Руководитель С.И. Матюшкин

Минск 2024

Цель: изучить правила формирования постфиксной записи арифметических выражений с использованием стека.
Задача: составить и отладить программу.

Индивидуальное задание

Написать программу формирования ОПЗ и расчета полученного выражения. Разработать удобный интерфейс ввода исходных данных и вывода результатов. Работу программы проверить на конкретном примере (табл. 5.1).

Таблица 5.1

Номер варианта	*Выражение*	*a*	*b*	*c*	*d*	*e*	*Результат*
1	$a / (b - c) \cdot (d + e)$	8.6	2.4	5.1	0.3	7.9	– 26.12

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

struct Stack {
    string info;
    Stack* next;
} *stack, *solution;

string resultList = "";
string inputList = "";

void inStack(Stack **p, string in) {
    Stack *t = new Stack;
    t -> info = in;
    t -> next = *p;

    *p = t;
}

string outStack(Stack **p) {
    string out;
    Stack* t = (*p);
    out = t -> info;

    *p = (*p) -> next;
    delete t;
    return out;
}

string firtstInStack(Stack *p) {
```

```
    if (p == NULL)
        return "(";
    else
        return p -> info;
}

void converToPrefixForm(Stack **p, string *inputString, string
*outString) {
    for (int i = 0; i <= (*inputString).length(); i++) {

         if (isdigit((*inputString)[i])) {
            string temp = "";

            for(int j = i; isdigit((*inputString)[j]) ||
(*inputString)[j] == '.'; j++) {
                temp.insert(temp.length(), 1, (*inputString)[j]);

                i = j;
            }

            (*outString) += temp;
        }

        else if ((*inputString)[i] == '(')
            inStack(&stack, "(");

        else if ((*inputString)[i] == ')') {
            while (firtstInStack(stack) != "(") {
                *outString += outStack(&stack);
            }
            string temp;

            outStack(&stack);
        }

        else if ((*inputString)[i] == '+' || (*inputString)[i] ==
'-' ||
            (*inputString)[i] == '*' || (*inputString)[i] == '/')
{

            string outStackElement = "";

            if ((*inputString)[i] == '+' || (*inputString)[i] ==
'-') {

                    while (firtstInStack(stack) != "(") {
                        string outStackElement = outStack(&stack);

                        (*outString).insert((*outString).length(),
outStackElement);

                    }
```

```
                string temp(1, (*inputString)[i]);

                inStack(&stack, temp);

            } else {
                    while (firtstInStack(stack) != "(" &&
firtstInStack(stack) != "+" && firtstInStack(stack) != "-") {
                        string outStackElement = outStack(&stack);

                        (*outString).insert((*outString).length(),
outStackElement);


                    }
                string temp(1, (*inputString)[i]);

                inStack(&stack, temp);

            }
        }


    }

    while (stack != NULL) {
        string outStackElement = outStack(&stack);

        (*outString).insert((*outString).length(),
outStackElement);
    }
}

double solveInPrefixForm(string str) {
    double result;

    for (int i = 0; i < str.length(); i++) {
        if (isdigit(str[i])) {
            string temp;

            temp.push_back(str[i]);
            temp.push_back(str[i+1]);
            temp.push_back(str[i+2]);

            inStack(&solution, temp);
            i = i + 2;
        }

        if (!isdigit(str[i])) {
            double firstOperand, secondOperand;

            secondOperand = stold(outStack(&solution));
            firstOperand = stold(outStack(&solution));
```

```
            if (str[i] == '+')
                inStack(&solution, to_string(firstOperand +
secondOperand));

            if (str[i] == '-')
                inStack(&solution, to_string(firstOperand -
secondOperand));

            if (str[i] == '*')
                inStack(&solution, to_string(firstOperand *
secondOperand));

            if (str[i] == '/')
                inStack(&solution, to_string(firstOperand /
secondOperand));
        }
    }

    result = stod(outStack(&solution));

    return result;
}

int main(int argc, const char * argv[]) {
    cout << "Введите пожалуйста выражение: " << endl;
    getline(cin, inputList);

    converToPrefixForm(&stack, &inputList, &resultList);
    cout << "Ваше выражение в ОПЗ: " << resultList << endl;

    cout << "Результат вычислений " <<
solveInPrefixForm(resultList) << endl;

    return 0;
                                                        }
```

```
Введите пожалуйста выражение:
8.6/(2.4-5.1)*(0.3+7.9)


Ваше выражение в ОПЗ: 8.62.45.1-/0.37.9+*
Результат вычислений -26.1185
```

Вывод: изучил правила формирования постфиксной записи арифметических выражений с использованием стека, составил и отладил программу.