МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Алгоритмы и структуры данных»

Тема: Рекурсия

Студентка гр. 9304 _____ Селезнёва А.В.

Преподаватель _____ Фиалковский М.С.

Санкт-Петербург

2020

Цель работы.

Ознакомиться с основными понятиями и приёмами рекурсивного программирования, получить навыки программирования рекурсивных процедур и функций на языке программирования C++.

Задание.

Вариант – 7.

Построить синтаксический анализатор для понятия *вещественное число*.

вещественное_число::= целое_число . целое_без_знака |
целое_число . целое_без_знакаEцелое число |
целое_числоEцелое_число

целое_без_знака::=цифра | цифра целое_без_знака

целое_число::=целое_без_знака | + целое_без_знака | −целое_без_знака

Выполнение работы.

На вход программа получает строку, которая содержит число – оно может быть как вещественным, так и любым другим.

В начале программы объявлена переменная *str* типа *string*. В случае, если количество аргументов командной строки больше одного, то в переменную *str* записывается строка, идущая следующим аргументом (после вызова функции). Иначе, в str записывается строка из стандартного потока ввода. В конце функции *main()* вызывается лямбда функция *Is_real_number* и выводится результат ее работы.

В лямбда захвате передаются ссылка на строку *str* и ссылки на переменные *existence_E*, *existence_point*, *real*; в параметры передаются ссылка на переменную *i* и сама функция для рекурсивного вызова. В функции проверяется, является ли элемент строки цифрой; расположение знаков ‘+’ и ‘-’ (они могут находиться только в начале числа и после элемента ‘E’); количество элементов ‘.’ и E (допустимо только одно вхождение), их расположение относительно друг друга – сразу после точки не может стоять E, а после E –

точка (проверяется в функции *check_point()*), а также расположение в целом – E и ‘.’ не могут стоять в конце строки. После каждого пройденного условия *i* увеличивается на единицу и вызывается функция *Is_real_number*.

В функцию *check_point()* типа *bool* передается подстрока строки *str*. Она проверяет, встречается ли в подстроке элемент ‘.’: если да, возвращает *true*, если нет – *false*.

Тестирование.

Тестирование проводится путем ввода в консоль различных случаев входных данных.

Результаты тестирования находятся в приложении Б.

Выводы.

Ознакомилась с основными понятиями и приемами рекурсивного программирования, получила навыки программирования рекурсивных процедур и функция на языке C++.

Разработала программу, рекурсивно проверяющую, являются ли входные данные вещественным числом.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: a_ds_1.cpp

```
#include <iostream>
#include <string>
#include <cctype>

bool check_point(std::string s){
    for( int l = 0; l < s.size(); ++l) {
        if(s[l] == '.'){
            return true;
        }
    }
    return false;
}


int main(int argc, char* argv []) {
    bool existence_E = false;
    bool existence_point = false;
    bool real = true;
    int i = 0;
    std::string str;
    auto Is_real_number = [&real, &str, &existence_E,
&existence_point](int& i, auto&& Is_real_number)->void{
        if((str[i] == '+' || str[i] == '-') && i !=
(str.size()-1) && ( i == 0 || str[i-1] == 'E') ) {
            ++i;
            Is_real_number(i, Is_real_number);
        }
        else if(isdigit(str[i]) != 0 ) {
            if(i == (str.size()-1)){}
            else {
            ++i;
            Is_real_number(i, Is_real_number);
            }
        }
        else if(str[i] == '.' && i != 0 && existence_point ==
false && str[i+1] != 'E' && i != (str.size()-1) ){
            existence_point = true;
            ++i;
            Is_real_number(i, Is_real_number);
        }
        else if(str[i] == 'E' && i != 0 && existence_E ==
false && check_point(str.substr(i)) == false && i != (str.size()-
1) ){
            existence_E = true;
            ++i;
            Is_real_number(i, Is_real_number);
        }
```

```
        else
        {
            real = false;
        }

    };

    if(argc < 2) {
        getline(std::cin, str, '\n');
    }
    else{
        int m = 1;
        for(int n = 0; argv[m][n]; ++n){
            str.push_back(argv[m][n]);
        }
    }
    Is_real_number(i, Is_real_number);

    if(real == true) {
        std::cout << "The real number\n";
    }else {
        std::cout << "This is not real number\n";
    }
    return 0;
}
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ

Таблица Б – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Результат проверки
1.	12.3	The real number	correct
2.	12.-3	This is not real number	correct
3.	+12.3	The real number	correct
4.	12E-1	The real number	correct
5.	12E.1	This is not real number	correct
6.	12E12.	This is not real number	correct
7.	+12.3E1	The real number	correct
8.	.12	This is not real number	correct
9.	E12	This is not real number	correct
10.	+12.-3E1	This is not real number	correct