**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Алгоритмы и структуры данных»**

**Тема: «Рекурсивное программирование. Синтаксический анализатор»**

Студент гр. 7382 Дрозд А. С.

Преподаватель Фирсов М. А.

Санкт-Петербург

2018

**Задание.**

Построить синтаксический анализатор для определяемого далее понятия

*константное\_выражение*.

*константное\_выражение*::=*ряд\_цифр*|

*константное\_выражение* *знак\_операции* *константное\_выражение*

*знак\_операции*::=+ | - | \*

*ряд\_цифр*::=*цифра* | *цифра ряд\_цифр*

**Пояснение задания.**

На вход программе подаётся последовательность символов. Итогом завершения программы должно быть выражение, дающее понять является ли последовательность символов константным выражением.

**Описание алгоритма.**

Программа считывает строку, затем посимвольно начинает проверку строки, является ли она константным выражением или нет. Посимвольная проверка различает 2 варианта событий: предыдущий символ является рядом цифр или предыдущий символ является знаком. Для первого символа строки считается, что предыдущий символ был знаком, это делается для упрощения проверки первого символа, т.к. он должен быть цифрой. В зависимости от варианта событий символ проходит разные проверки.

**Описание функций**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Выходной параметр** | **Входные данные** | **Описание** |
| **indent** | - | **Int –** количество рекурсий | Выводит несколько пробелов, равное количеству рекурсий. |
| **fun** | int | **Char \* –** текущий, проверяемый на корректность символ;  **Int –** значение предыдущего символа;  **Int –** ключ, дающий понять требуются ли пояснения;  **Int –** глубина рекурсии | Посимвольно проверяет является ли строка константным выражением. |

**Тестирование.**

|  |  |
| --- | --- |
| Исходное выражение: | Результат: |
| 1  123+45-8 | Выражение является конcтантным выражением |
| 1  +456\*8 | Выражение не является конcтантным выражением |
| 1  5654-4564\*11+55 | Выражение является конcтантным выражением |
| 1  46+gf45-798 | Выражение не является конcтантным выражением |
| 1  -46578+77 | Выражение не является конcтантным выражением |

**Вывод.**

В процессе выполнения лабораторной работы были усвоены методы использования рекурсии, а так же написана программа с использованием метода рекурсии.

**Приложение 1. Код программы.**

* **[test.sh](https://github.com/makometr/AiSD/pull/37/files" \l "diff-240c2792fd92b595432e18629f6e16b6)**

#!/bin/bash

gcc ./Sourse/lab1.c -o Lab1

echo -e '\_\_\_\_\_\_\_\nTest 1:'

cat ./Tests/test1.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab1 < ./Tests/test1.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 2:'

cat ./Tests/test2.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab1 < ./Tests/test2.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 3:'

cat ./Tests/test3.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab1 < ./Tests/test3.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 4:'

cat ./Tests/test4.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab1 < ./Tests/test4.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 5:'

cat ./Tests/test5.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab1 < ./Tests/test5.txt

* **main[.](https://github.com/makometr/AiSD/pull/37/files" \l "diff-240c2792fd92b595432e18629f6e16b6)c**

#include <stdio.h>

void indent(int I) //функция для отступов

{

while(i!=0)

{

printf(" ");

i--;

}

}

int fun(char\* a,int n,int key,int dep)// рекурсивная функция,помогающая определить, является ли строка константным выражением

// а - текущий проверяемый на корректность символ

// n - значение предыдущего символа

// key - ключ, дающий понять требуются ли пояснения

// dep - счетчик глубины рекурсии

{

int i=dep;

if(\*a=='\n')// встреча конца строки означает конец рекурсии

return 1;

if(n==1) //n==1 значит что предыидущий символ был знаком

{

if(\*a>47 && \*a<58) // если символ является цифрой

{

if(key==1)//если нужны пояснения

{

indent(i);

printf("Найдена цифра %c.Следующим символом должна быть цифра,знак или конец строки\n",\*a);

}

i=fun(a+1,2,key,dep+1);

if(key==1)//если нужны пояснения

{

indent(dep);

printf("Рекрсия закрыта\n");

}

return i;

}

else

{

if(key==1)//если нужны пояснения

{

indent(dep);

printf("Найдена ошибка.Символ %c.Завершение рекурсий.\n",\*a);

}

return 2;

}

}

if(n==2) //n==2 значит что предыидущий символ был числом

{

if((\*a>47 && \*a<58) || \*a==42 || \*a==43 || \*a==45) //если символ является цифрой или знаком

{

if(\*a>47 && \*a<58) // если символ является цифрой

{

if(key==1)//если нужны пояснения

{

indent(i);

printf("Найдена цифра %c.Следующим символом должна быть цифра,знак или конец строки\n",\*a);

}

i=fun(a+1,2,key,dep+1);

if(key==1)//если нужны пояснения

{

indent(dep);

printf("Рекрсия закрыта\n");

}

return i;

}

else// если символ является знаком

{

if(key==1)//если нужны пояснения

{

indent(i);

printf("Найден знак %c.Следующим символом должна быть цифра или конец строки\n",\*a);

}

i=fun(a+1,1,key,dep+1);

if(key==1)//если нужны пояснения

{

indent(dep);

printf("Рекрсия закрыта\n");

}

return i;

}

}

else

{

if(key==1)//если нужны пояснения

{

indent(dep);

printf("Найдена ошибка.Символ %c.Завершение рекурсий.\n",\*a);

}

return 2;

}

}

}

int main()

{

int key; // ключ, отвечающий за пояснения

char a='1'; // символ, необходимый для беспроблемного считывания строки

int result; // переменная, в которую выведется результат рекурсии

char str[100]; //строка, для считывания данных

printf("Введите ключ: 1- если нужны пояснения.В обратном случае введите любое число кроме 1.\n");

scanf("%d",&key);

while(a!='\n')// цикл нужен чтобы в строку в самом начале не попал \n

{

scanf("%c",&a);

}

printf("Введите выражение\n");

fgets(str,100,stdin);

result=fun(str,1,key,0);

if(result==1)

printf("Выражение является конcтантным выражением");

else

printf("Выражение не является конcтантным выражением");

}