

**МИНОБРНАУКИ РОССИИ  
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ  
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)  
Кафедра МОЭВМ**

**ОТЧЕТ  
по лабораторной работе №1  
по дисциплине «Алгоритмы и структуры данных»  
ТЕМА: РЕКУРСИЯ**

**Студент гр. 7381**

\_\_\_\_\_

**Кушкочева А.О.**

**Преподаватель**

\_\_\_\_\_

**Фирсов М. А.**

**Санкт-Петербург  
2018**

### **Цель работы.**

Ознакомиться с основными методами использования рекурсии и написать программу с использованием рекурсии.

### **Задание.**

Вариант10.

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*

### **Основные теоретические положения.**

Рекурсия — определение, описание, изображение какого-либо объекта или процесса внутри самого этого объекта или процесса, то есть ситуация, когда объект является частью самого себя.

### **Пояснение задания:**

На вход подается строка. Задача состоит в том, чтобы определить удовлетворяет ли она определенному понятию «константное выражение». Реализовать рекурсивную функцию проверки.

### **Описание алгоритма:**

В начале работы происходит ввод данных (строка), после чего происходит проверка на то, является ли константным выражением. Если введенные данные некорректны, то программа выводит соответствующую ошибку и завершает работу.

Функция **void error(int i, int deep);**

### Аргументы:

int i – номер ошибки.

int deep – глубина рекурсии.

### Описание:

Данная функция вызывается в тот момент, когда нужно вывести сообщение об ошибке . Вывод зависит от того, какой аргумент подали в int i,

если i=1, то выводится «ERROR: invalid first character!», если 2- «ERROR: requires a number of digits!», если 3- «ERROR: extra characters in the input line!»

### Функция **int main()**

#### Описание:

В главной функции программы происходит выделение блока динамической памяти под массив типа char, который будет задействован в ходе программы. Именно в этот массив идет считывание возможного константного выражения. Далее происходит вызов функции analiz, которая определяет является ли строка константным выражением или нет в зависимости от того, что вернула функция analiz. Если 1, то является, если 0-нет.

### Функция **int analiz(char\* str, int deep);**

#### Аргументы:

char\* str-возможное константное выражение

int deep-глубина рекурсии

#### Описание:

Анализируем полученную строку по одному элементу. Для начала идет проверка нулевого элемента массива на то-является ли цифрой или нет. Если проверка прошла успешно, то рекурсивно вызывается функция analiz, при этом глубина увеличивается на единицу, если нет, то вызывается функция error с номером ошибки 1 и работа функции прекращается, возвращая 0. В дальнейшем происходит такая же проверка элемента массива на цифру, с некоторыми отличиями : при неуспешном исходе проверяем на знак операции. Если это знак операции, то вызываем функцию check\_next, если нет, то это не цифра и не знак операции, а это значит, что принятая строка не константное выражение, следовательно, прерываем функцию, возвращая 0.

### Функция **int check\_next(char\* str, int deep);**

char\* str-возможное константное выражение

int deep-глубина рекурсии

#### Описание:

В условии задания сказано, что после знака операции обязательно должна следовать цифра. В данной функции проверяем ее выполнимость.

### Тестирование программы:

Файлы с тестовыми данными, находящиеся в директории **Tests** и имеющие названия вида **testn.txt** ( $1 \leq n \leq 6$ ), проверяют функционал и работоспособность написанной программы.

Ввод	Результат выполнения программы
Test1: 123+4*57-6	123+4*57-6 It is a constant expression!
Test2: 2-23*78-238	2-23*78-238 It is a constant expression!
Test3: -12*34	- ERROR: invalid first character! It's not a constant expression!
Test4: 442+a63-6	442+a ERROR: extra characters in the input line! It's not a constant expression!
Test5: 460-23*6+73-	460-23*6+73- ERROR: requires a number of digits! It's not a constant expression!
Test6: 123	123 It is a correct constant expression!

### Выводы.

В ходе выполнения лабораторной работы получены, а также закреплены знания по теме «рекурсия».

## ИСХОДНЫЙ КОД:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

void error(int i, int deep){
    for( int i = 0; i <= deep; i++)
        printf(" ");

    printf("ERROR: ");

    switch (i){
        case 1:
            printf("invalid first character!\n");
            printf("onstant expression analyzer!\n");
            break;
        case 2:
            printf("requires a number of digits!\n ");
            break;
        case 3:
            printf("extra characters in the input line!\n");
            break;
        default:
            break;
    }
}

int check_next(char* str, int deep){
    if(str[deep] == '\0'){
        error(2, deep);
        return 0;
    }

    else
        return 1;
}

int analiz(char* str, int deep){
    int flag = 1;

    if(str[deep] == '\0'){
        return 1;
    }
}
```

```

for(int i = 0; i < deep; i++)
    printf(" ");
printf(">The incoming character: %c", str[deep]);

if( !isdigit(str[0]) ){
    printf("\n");
    error(1, deep);
    printf("<\n");
    return 0;
}

if( isdigit(str[deep]) ){
    printf("\n");
    analiz(str, deep+1);
}
else{
    if( str[deep] == '+' || str[deep] == '-' || str[deep] == '*' ){
        printf("   \\\\ Waiting for a series of numbers.\n");
        if( check_next(str, deep+1) )
            analiz(str, deep+1);
        else{
            flag = 0;
            deep--;
        }
    }

    else{
        printf("\n");
        flag = 0;
        error(3, deep);
    }
}

for( int i = 0; i < deep; i++)
    printf(" ");
printf("<\n");

if(flag)
    return 1;
else
    return 0;
}

int main(){
    printf("Constant expression analyzer:\n");

    int len_str = 100;

```

```

char* str=(char*)malloc(sizeof(char)*len_str);
scanf("%s", str);

if( analiz(str, 0) )
    printf("It's a constant expression!\n");
else
    printf("It's not a constant expression! ");

return 0;
}

```

## Приложение А. Файл compile.sh

```

#!/bin/bash
gcc ./Source/main.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
echo -e "
echo -e '_____\nTest 6:'
cat ./Tests/Test6.txt

```

```
echo -e '_____\nTesting:\n'  
./Lab1 < ./Tests/Test6.txt
```