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: ПРОГРАММИРОВАНИЕ РЕКУРСИВНЫХ АЛГОРИТМОВ

Студент гр. 8381 _____ Нгуен Ш. Х.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2019

Цель работы.

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

Основные теоретические положения.

Рекурсивным называется объект, содержащий сам себя или определенный с помощью самого себя.

Мощность рекурсии связана с тем, что она позволяет определить бесконечное множество объектов с помощью конечного высказывания. Точно так же бесконечные вычисления можно описать с помощью конечной рекурсивной программы. Рекурсивные алгоритмы лучше всего использовать, когда решаемая задача, вычисляемая функция или обрабатываемая структура данных определены с помощью рекурсии.

Если процедура (функция) P содержит явное обращение к самой себе, она называется прямо рекурсивной. Если P содержит обращение к процедуре (функции) Q, которая содержит (прямо или косвенно) обращение к P, то P называется косвенно рекурсивной.

Задание

Построить синтаксический анализатор для понятия скобки.

скобки::=A | (B скобки скобки)

Ход работы

1. Произведён анализ задания и выделена рекурсивная функция **bracket** и разработана её рекурсивная часть.
2. Разработана программа, содержащая в себе:

2.1.Функция, которая принимает каждый символ входной символьной строки **arr**, сохраняется в char **a**:

```
void symbol(string arr, char& a){
        a = arr[pos];
        pos++;
}
```

2.2. Функцию **bracket** для обработки строки. Флаг **FlagErr** ввода хранит значение типа ошибки. Символ строки **arr** берется по очереди и сохраняется в переменной char **a**. Если **a = 'A'**, то **FlagErr** = 0, в противном случае **a = '('** продолжайте получать больше символов из строкового аргумента **arr** и проверять, будет ли следующий символ **a = 'B'**, если так рекурсивная функция вызывается 2 раза , а неудовлетворенные случаи возвращают разные ошибки через переменную **FlagErr**.

2.3. Функция main для инициализации массива **arr** , символ **a**, чтобы получить каждый символ **arr** и флаг **FlagErr** хранит значения, чтобы сообщить об ошибках. Функция Bracket() вызывается и значение флага **FlagErr** установлено.

Вывод

В ходе лабораторной работы были изучены основы работы с рекурсивными функциями. А также реализована рекурсивный вызов функции.

Тестирование

Input	Output
AIt's a bracket.	Its a bracket.
AB	Error: Extra characters in input string.
(BAA)	Its a bracket.
(B(BAA)(BA(BAA)))	Its a bracket.
(B(BAA)(BA(BAA)))A	Error: Extra characters in input string.

ПРИЛОЖЕНИЕ А

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
int pos = 0;

void symbol(string arr, char& a){
   a = arr[pos];
   pos++;
}

bool bracket(string arr,char& a, int& FlagErr){
      symbol(arr,a);
      if(a != '\0'){
            if(a == 'A')
                  FlagErr = 0;
            else if(a == '('){
                  symbol(arr,a);
                  if(a!='\0' && a == 'B'){
                              if(bracket(arr, a, FlagErr)){
                                    if(bracket(arr,a,FlagErr)){
                                          symbol(arr,a);
                                          if(a!='\0' && a == ')'){
                                                FlagErr = 0;
                                          } else FlagErr = 1;
                                    } else FlagErr = 2;
                              } else FlagErr = 3;
                  } else FlagErr = 4;
            } else FlagErr = 5;
      } else FlagErr = 5;
      if(FlagErr == 0)
            return true;
      else
            return false;
}


int main(){
      string arr;
      char a;
```

```
    int FlagErr;
    cin>>arr;
    bracket(arr,a,FlagErr);
    symbol(arr,a);
    if(a!='\0')
        FlagErr = 6;

    switch(FlagErr){
        case 0:
            cout<<"It's a bracket."<<endl;
            break;
        case 1:
            cout<<"It's not a bracket"<<endl;
            cout<<"Error: Missing '('."<<endl;
            break;
        case 2:
            cout<<"It's not a bracket"<<endl;
            cout<<"Second bracket error."<<endl;
            break;
        case 3:
            cout<<"It's not a bracket"<<endl;
            cout<<"First bracket error."<<endl;
            break;
        case 4:
            cout<<"It's not a bracket"<<endl;
            cout<<"Error: Missing 'B'."<<endl;
            break;
        case 5:
            cout<<"It's not a bracket"<<endl;
            cout<<"Error: Missing 'A' or '('."<<endl;
            break;
        case 6:
            cout<<"It's not a bracket"<<endl;
            cout<<"Error: Extra characters in input string."<<endl;
            break;
    }
    return 0;
}
```