МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА МО ЭВМ

ОТЧЕТ
по лабораторной работе No1
по дисциплине «Алгоритмы и структуры данных»
Тема: Программирование рекурсивных алгоритмов

Студент гр. 7383 _____ Рудоман В.А.

Преподаватель _____ Размочаева Н.В.

Санкт-Петербург

2018

Содержание

1. Цель работы

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

2. Задание

Вариант 16.

Построить синтаксический анализатор для понятия *скобки*.

скобки::=A | B | (*скобки скобки*)

3. 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.

4. Ход работы

Программа посимвольно считывает входные данные, а именно последовательность символов ASCII. Исходная строка выводится на консоль до запуска программы. По результату обработки исходная строка выводится вплоть до ошибочного символа, если таковой был найден, и не выводится, если она верна. Соответствующие сообщения об ошибке или корректности считанной строки присутствуют в коде. В ходе выполнения программы на консоль отображается глубина рекурсии.

Функция **bool bracket (ifstream &infile, char ch)**

Рекурсивная функция.

Принимает на вход адрес файла и символ типа char. Возвращает значение типа bool (true, false).

5. Описание

На вход программе подается текст из файла. Считывается первый символ. Если он неверные, выводится сообщение об ошибке. В противном случае программа продолжает работу. Вызывается функция **bracket.** Функция проверяет символ и выводит его на экран. Если символом является «A» или «B», возвращается значение **true**. Если символом является «(», программа продолжает работу. Считывается следующий символ. Вызывается функция **bracket**. Если вернувшееся значение **true**, считывается следующий символ и снова вызывается функция **bracket**. В противном случае выводится сообщение об ошибке. Если после открывающей скобки «(» следовало 2 символа «A» и «B», то программа проверяет, является ли следующий символ закрывающей скобкой «)».

Если является, то возвращается значение **true**, в противном случае **false** и сообщение об ошибке. Если после выполнения программы остались лишние символы, программа сообщит об ошибке. Если на вход программе подается выражение A или B, без последующих символов, программа считает это выражение верным.

6. Тестирование программы

Программа собрана и протестирована в ОС macOS Sierra версия 10.12.6. С использование компилятора g++. В других ОС и компиляторах программа тестирование не проходила.

Входные данные	Вывод	Результат
A	A ЭТО СКОБКИ	ВЕРНО
((AB)A)	((AB)A) ЭТО СКОБКИ	ВЕРНО
(ABB	(ABB Error № 1 ОЖИДАЛСЯ :)	ВЕРНО
AB	AB Error № 5 ЛИШНИЙ СИМВОЛ	ВЕРНО

7. Вывод

В ходе лабораторной работы были получены навыки работы с рекурсивными функциями. Был реализован синтаксический анализатор скобок. В ходе тестирования ошибок выявлено не было. В тестовом случаее 3 программа вывела номер ошибки и предложила ожидаемый символ. Рекурсивные функции позволяют упростить написание и повысить читабельность кода, рекурсия удобна в использовании при работе с повторяющимися типами данных.

ПРИЛОЖЕНИЕ А

Исходный код

```
#include <iostream>
#include <fstream>
#include <cstdio>
#include <cstring>

using namespace std ;

bool bracket (ifstream &infile, char ch);
void Error (short k);
bool forFile (ifstream &infile, bool check, char ch);


int main ( ){

      cout << "АНАЛИЗАТОР СКОБОК" << endl;
      cout << "ДЛЯ АНАЛИЗА СКОБОК ИЗ test.txt НАЖМИТЕ '1'" << endl;
      cout << "ДЛЯ ПЕРЕЗАПИСИ И АНАЛИЗА СКОБОК ИЗ test.txt НАЖМИТЕ '2'" <<
endl;
      cout << "ДЛЯ ПРЕРЫВАНИЯ ПРОГРАММЫ НАЖМИТЕ '0'" << endl;

      bool exit = true, check;
      char ch;
      char arr[100];
      int forSwitch;

      FILE* fp;
      cin >> forSwitch;

      switch (forSwitch){
            case 1:{
                  ifstream infile ("test.txt");
                  forFile(infile, check, ch);
                  break;
            }
            case 2:{
                  fp = fopen("test.txt", "w");
```

```
                if (!fp)
                        return 0;
                cin >> arr;
                fputs(arr,fp);
                fclose(fp);
                ifstream infile ("test.txt");
                forFile(infile, check, ch);
                break;
        }
        case 0:{
                cout << "ВСЕГО ДОБРОГО" << endl;
                return 0;
        }
        default: {
                cout<<"НЕВЕРНЫЙ ВВОД"<<endl;
                break;
        }
    }
    return 0;
}


bool bracket (ifstream &infile, char ch) {
    static string tab;
    bool forCheck;
    if(ch == 'A' || ch == 'B')
        return true;

        if(ch == '(') {
            tab.push_back('\t');
            if(infile >> ch) {
                cout << tab << ch << endl;
                forCheck = bracket(infile, ch);
            }
            else {
                Error(2);
                return false;
            }
            if (forCheck) {
                if (infile >> ch)
                    cout << tab << ch << endl;
                forCheck = bracket(infile, ch);
```

```
                }
                else {
                      Error(2);
                      return false;
                }

                tab.pop_back();
                if(forCheck) {
                      if(infile >> ch) {
                            cout << tab << ch << endl;
                            if (ch != ')'){
                                  Error(1);
                                  return false;
                            }
                            return (ch == ')');
                      }
                      else {
                            Error(2);
                            return false;
                      }
                }
          }

     else {
          Error(6);
          return false;
     }

     return true;
}

bool forFile (ifstream &infile, bool check, char ch) {
                if (infile >> ch){
                      cout << ch << endl;
                      if ((ch == 'A') || (ch == '(') || (ch == 'B'))
                            check = bracket (infile, ch);

                      else{
                            Error(3);
                            return 0;
                      }
                }
```

```
                else{
                      Error(4);

                }
                if (infile >> ch) {
                      cout << ch;
                      if (ch != EOF) {
                            Error(5);
                            return 0;
                      }
                }
                cout << endl;
                if (check)
                      cout<<"ЭТО СКОБКИ"<<endl;
                return true;
}


void Error (short k){

      cout << endl << "   Error № " << k << endl;

      switch (k) {

      case 1: cout << "::ОЖИДАЛСЯ: ) ::" << endl; break;

      case 2: cout << "::ОЖИДАЛИСЬ: A, B, ( ::" << endl; break;

      case 3: cout << "::НЕВЕРНЫЙ СИМВОЛ::" << endl; break;

      case 4: cout << "::ПУСТОЙ ФАЙЛ::" << endl; break;

      case 5: cout << "::ЛИШНИЙ СИМВОЛ::" << endl; break;

      case 6 : cout << "ERROR..." << endl; break;

      };
}
```