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: Программирование рекурсивных алгоритмов

Студент гр. 9303 Молодцев Д.А.

Преподаватель Филатов Ар.Ю.

Санкт-Петербург
2020

Цель работы.

Создание рекурсивного алгоритма для анализа понятия «скобки», дальнейшее тестирование программы.

Задание.

Вариант 15

Построить синтаксический анализатор для понятия «скобки».

//скобки::=A|A(ряд_скобок)
//ряд_скобок::= скобки|скобки;ряд_скобок

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. Многие известные функции могут быть определены рекурсивно. Например факториал, который присутствует практически во всех учебниках по программированию, а также наибольший общий делитель, числа Фибоначчи, степенная функция и др.

Выполнение работы.

Для работы программы-анализатора, были реализованы следующие функции: bool is_bracket(string& analized_str), bool is_row_of_brackets(string& analized_str), void ErrorPrint(int error), void PrettyPrint(int is_start) и bool main_analizator(string& analized_str).

Int main() – в данной функции считывается введенное с консоли пользователем имя файла, в случае, если файл с таким именем существует, то будет происходить последовательное считывание строк из файла, к каждой из строк будет применяться функция main_analizator, также будет выводиться информация о том, является ли строка верной, или же нет.

bool is_bracket(string& analized_str) – одна из двух взаимно рекурсивных функций программы, принимает на вход анализируемую строку. Возвращает true или вызывает is_row_of_brackets в случае, если часть строки – это «скобки», иначе false. В функции последовательно проверяется: является ли элемент символом «A», «;» или «(», в случае, если встретился символ «(» проверяется наличие следующего символа, при наличии его равенство символу «A», далее вызывается рекурсия, для проверки понятий «ряд скобок», после же проверяется, замыкается ли выражение символом «)». При несоблюдении хотя бы одного из этих условий, функция вернет false. Также, при несовпадении после окончания работы функции значений текущего индекса элемента и длины, уменьшенной на единицу, выражение не будет считаться скобками.

bool is_row_of_brackets(string& analized_str) – одна из двух взаимно рекурсивных функций программы, принимает на вход анализируемую строку. Возвращает true или вызывает is_brackets в случае, если часть строки – это «ряд скобок», иначе false. В функции последовательно проверяется: является ли элемент символом «A»,«)», «;» или «(», в случае, если встретился символ «(» проверяется наличие следующего символа, при наличии его равенство символу «A», далее вызывается рекурсия, для проверки понятий «ряд скобок», после же проверяется, замыкается ли выражение символом «)». При несоблюдении хотя бы одного из этих условий, функция вернет false. Также, при несовпадении после окончания работы функции значений текущего индекса элемента и длины, уменьшенной на единицу, выражение не будет считаться рядом скобок.

Void PrettyPrint(int is_start) – печатает отступ из символов «#», количество символов равно глубине рекурсии.

void Error() – выводит сообщение об ошибке.

Так же, дабы избежать лишнего копирования, в программе используются 3 глобальные переменные типа static int: depth – глубина рекурсии, curr_ind – индекс текущего проверяемого символа, len – длина

Исходный код программы представлен в приложении А. Результаты тестирования включены в приложение Б

Выводы.

Была реализована программа, включающая в себя рекурсивную функцию is_bracket для синтаксического анализа выражения «скобки», функцию is_row_of_brackets для синтаксического анализа выражения «ряд

скобок». Также было проведено тестирование программы. Были удовлетворены следующие требования: информация на вход подается из файла, выводятся сообщения о начале и конце вызова функции Bracket с отступами, соответствующими глубине рекурсии, итог работы программы и все сообщения выводятся в консоль.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

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

using namespace std;

static int depth=0;
static int curr_ind=0;
static int len=0;

void PrettyPrint(int is_start);
void ErrorPrint(int error);
bool is_row_of_brackets(string &analized_str);
bool is_brackets(string &analized_str);
bool main_analizator(string analized_str);

int main() {
    string path;
    string analized_brack;
    cout<<"Enter file name:\n";
    getline(cin,path);
    fstream input_file(path);
    if(!input_file.is_open()){
        cout<<"Wrong filename.\n";
    }else{
        cout<<"File has opened.\n\n";
    }
    while(getline(input_file,analized_brack)){
        if(main_analizator(analized_brack)){
            cout<<"This brackets are right:
"<<analized_brack<<'\n';
        }else{
            cout<<"This brackets are wrong: "<<
analized_brack <<'\n';
        }
        depth=0;
        curr_ind=0;
        len=0;
    }
    return 0;
}

void ErrorPrint(int error){
    switch (error) {
        case 1:
            cout<<"Error 1: Wrong structure of brackets.\n";
```

```
            break;
        case 2:
            cout<<"Error 2: Wrong structure of row of brack-
ets.\n";
            break;
        default:
            break;
    }
}

void PrettyPrint(int is_start) {
    if(is_start==1) {
        for (int i = 0; i < depth; i++) {
            cout << "#";
        }
        cout << "Start\n";
        depth++;
    }else if(is_start==2){
        depth--;
        for (int i = 0; i < depth; i++) {
            cout << "#";
        }
        cout << "End\n";
    }
    curr_ind++;
}

bool main_analizator(string analized_str){
    len = analized_str.length();
    if(analized_str[0]!='A'){
        ErrorPrint(1);
        return false;
    }else{
        return is_brackets(analized_str);
    }
}

bool is_row_of_brackets(string &analized_str){
    if(depth==0 && curr_ind==len){
        return true;
    }else if (analized_str[curr_ind]=='A'){
        PrettyPrint(0);
        return is_row_of_brackets(analized_str);
    }else if(analized_str[curr_ind]==';'){
        if(analized_str[curr_ind+1]=='A'){
            PrettyPrint(0);
            return is_row_of_brackets(analized_str);
        }else{
            ErrorPrint(2);
            return false;
        }
```

```
    }else if(analized_str[curr_ind]=='('){
        PrettyPrint(1);
        return is_row_of_brackets(analized_str);
    }else if(analized_str[curr_ind]==')'){
        PrettyPrint(2);
        return is_brackets(analized_str);
    }else{
        ErrorPrint(1);
        return false;
    }
}

bool is_brackets(string &analized_str){
    if(depth==0 && curr_ind==len){
        return true;
    }
    else if(analized_str[curr_ind]=='A'){
        PrettyPrint(0);
        return is_brackets(analized_str);
    }else if(analized_str[curr_ind]=='('){
        if(analized_str[curr_ind+1]=='A'){
            PrettyPrint(1);
            return is_row_of_brackets(analized_str);
        }else{
            ErrorPrint(1);
            return false;
        }
    }else if(analized_str[curr_ind]==')'){
        PrettyPrint(2);
        return is_brackets(analized_str);
    }else if(analized_str[curr_ind]==';'){
        if(analized_str[curr_ind+1]=='A'){
            PrettyPrint(0);
            return is_row_of_brackets(analized_str);
        }else{
            ErrorPrint(1);
            return false;
        }
    }else {
        ErrorPrint(1);
        return false;
    }
}
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ

Файл со входными данными: input.txt

Таблица Б.1 - Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарии
1.	AAAAA	These brackets are right: AAAAA	Программа работает нормально.
2.	A(A;A(A))	Start #Start #End End These brackets are right: A(A;A(A))	Программа работает нормально.
3.	A()	Error 1: Wrong structure of brackets. This brackets are wrong: A()	Программа работает нормально. Выражение не соответствует требованиям.
4.	A(A(A(AAA;A(A))))	Start #Start ##Start ###Start ###End ##End	Программа работает нормально.

		#End End These brackets are right: A(A(A(AAA;A(A))))	
5.	AAAAAAAAAAAA A;	Error 1: Wrong structure of brackets. These brackets are wrong: AAAAAAAAAAAA A;	Программа работает нормально. Выражение не соответствует требованиям
6.	Part of the team, part of the ship!	Error 1: Wrong structure of brackets. These brackets are wrong: Part of the team, part of the ship!	Программа работает нормально. Выражение не соответствует требованиям.
7.	A(A(A(A;A(A(A(A;A(A(A(A;A(A))))))) (A(A(A;A(A)))))))	Start #Start ##Start ###Start ####Start #####Start ######Start #######Start ########Start #########Start #########End ########End #######End ######End #####End ####End ###End ###Start ####Start #####Start	Программа работает нормально.

		######Start ######End #####End ####End ###End ##End #End End	
8.	;A;A	Error 1: Wrong structure of brackets. These brackets are wrong: ;A;A	Программа работает нормально.
9.	A(A(A;A)))	Start #Start #End End End Error 1: Wrong structure of brackets. These brackets are wrong: A(A(A;A)))	Программа работает нормально. В выражении лишняя закрывающая скобка.

9.	(A(B(BA(BAA))A)(BA(BAA)))	!Start String contains extraor incorrect characters! Correct format of string is: brackets := A | B(brackets brackets) !End This is not parenthesis: (A(B(BA(BAA))A)(BA(BAA)))	Программа работает нормально. Ошибка: недопустимый символ A после открывающей скобки.
10. (тест на некорректное имя файла)	file	Wrong filename.	Программа работает нормально.