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. 8304 _____ Птухов Д.А.

Преподаватель _____ Фирсов М.А.

Санкт-Петербург

2019

Цель работы.

Изучить основы рекурсии и составления эффективных алгоритмов.

Постановка задачи.

1) Разработать программу, использующую рекурсию;
2) Сопоставить рекурсивное и итеративное решение задачи;
3) Сделать вывод о целесообразности и эффективности рекурсивного подхода для решения данной задачи.

Вариант 14

Построение синтаксического анализатора для понятия *скобки*.

скобки::= A | (B *скобки скобки*)

Описание алгоритма.

Для решения поставленной задачи была реализована рекурсивная функция check, которая заменяет каждую подстроку “(B A A)” полученной строки на подстроку “A” и рекурсивно вызывает саму себя от измененной строки. Выход из рекурсии осуществляется посредством двух проверок. Первая – проверка на то, что полученная строка равна строке “A” (так как дальнейшая замена бесполезна) и возвращает значение True, вторая – проверка на подстроки (B A A) в полученной строке, если на одном из шагов таковой не было найдено, то функция возвращает значение False.

Спецификация программы.

Программа предназначена для синтаксического анализа выражения методом рекурсии.

Программа написана на языке C++. Входными данными является либо строка, либо путь до файла содержащего строку (строки). Выходными данными являются промежуточные значения входной после замены описанной выше и глубина рекурсии.

Описание функций.

1) Функция check_strElements.

Объявление функции:

```
bool check_strElements(std::string s);
```

Данная функция осуществляет проверку на отсутствие нешаблонных символов во входной строке. Если такие имеются возвращаемое значение – False, иначе – True. Обход по принятой строке осуществляется при помощи цикла for, проверка на нешаблонные символы осуществляется при помощи условного оператора if.

2) Функция check.

Объявление функции:

```
bool check(std::string& terminate, std::ostream& out, size_t n =
0);
```

Данная функция осуществляет проверку на то, удовлетворяет ли входная строка поставленной задаче. Функция принимает проверяемую строку, ссылку на поток вывода и имеет параметр n, который по умолчанию равен 0, он используется для подсчета глубины рекурсии. Сначала функция проверяет входную строку на равенство строке “A”, если это условие выполнилось, то функция завершается, возвращая True. Далее при помощи метода find класса std::string осуществляется поиск подстроки “B A A” (возвращаемое значение сохраняется в переменную it), если данная подстрока не была найдена (то есть функция вернула специальный параметр std::string::npos), то функция завершается, возвращая False. Если проверка последняя проверка не выполнилась, то подстрока “B A A” была найдена и индекс ее первого вхождения в строку terminate содержится в переменной it. Далее при помощи метода erase класса std::string из строки terminate удаляется найденная ранее подстрока “B A A”. Далее при помощи метода insert класса std::string осуществляется “вставка” подстроки “A” в строку terminate (места вставки и удаления определяются при помощи ранее найденного индекса). В конце функция рекурсивно вызывает саму себя от уже измененной строки.

Вывод.

Был получен опыт работы с рекурсией и с построением синтаксического анализатора. На мой взгляд, итеративное решение поставленной задачи более эффективно.

ПРИЛОЖЕНИЕ

1) ТЕСТИРОВАНИЕ:

Работа программы для строки (B (B A A) (B A (B A A))):

```
Choose input format:
        1)Enter string

        2) Read from file (Default file is located along the path:D:/LAB1_SOURCE/InputSource.txt
        If you want to change file location, you have to enter path as second argument
        Don't forget to change all '\' to '/'
1
Choose output format:
        1)Console

        2)File (Default file is located along the path: D:/LAB1_SOURCE/OutputSource.txt)
        If you want to change file location, you have to enter path as second argument
        Remember that debagging output will be saved with programm result
1
Enter string:
(B (B A A) (B A (B A A)))

Value of the check-string after the next function call : (B (B A A) (B A (B A A)))

Recursion depth: 0
------------------
Value of the check-string after the next function call : (B A (B A (B A A)))

Recursion depth: 1
------------------
Value of the check-string after the next function call : (B A (B A A))

Recursion depth: 2
------------------
Value of the check-string after the next function call : (B A A)

Recursion depth: 3
------------------
Value of the check-string after the next function call : A

Recursion depth: 4
------------------
Entered string: A
Result: true
```

Таблица результатов ввода/вывода тестирования программы

Входная строка	Вывод программы
A	True
(B A A)	True
(B A B)	False

B	False
(B A (B (B (B (B (B A A) A) (B A A)) A) A))	True
(B A (B A (B A (B A B A A))))	False
(B (B A) A)	False
(B A (B (B (B (B (B A A) A) (B A A)) A) B))	False
(B A (B (B (B (B (B A A) A) (B A A)) A)))	False
(B A (A (B (B (B (B A A) A) (B A A)) A) A))	False
(B C (B (B (B (B (B A A) A) (B A A)) A) A))	False
(B A (B (B (B (B (B (B A A) A) A) (B A A)) A) A))	True

2) ИСХОДНЫЙ КОД:

```
#include <iostream> // std::cin, std::cout, std::ostream
#include <string> // std::string и сопутствующие функции
#include <fstream> // std::ofsrteam

//Функция, ведущая диалог с пользователем
void Dialog();

//Функция, проверяющая отсутсвие нешаблонных символов в принятой строке
bool check_strElements(std::string s);

//Функция, которая возвращает извлеченный из принять строки путь до файла сохранения
std::string make_path(std::string var, std::string default_path);

//Функция, вызывающая для полученной строки s функцию check
void call(std::string s, std::ostream& const out);

//Рекурсивная функция, осуществляющая проверку введенной пользователем строки
bool check(std::string& terminate, std::ostream& out, size_t n = 0);

//Функция преобразующая значение типа bool в значение типа std::string
std::string bool_to_string(bool a);


void Dialog()
{
	//of - output format
	int  of;
	// ivar - input variant, ovar - output variant
	std::string ivar, ovar;
```

```
	//out - поток вывод, зависящий от выбора пользователя
	std::ofstream out;

	//Считывание введенной пользователем команды, использование std::getline обусловлено возможным наличием пробельных символов в считываемой строке
	std::getline(std::cin, ivar);

	std::cout << "Choose output format:\n\t1)Console\n\n\t2)File (Default file is located along the path: D:/LAB1_SOURCE/OutputSource.txt)\n\t"
		"If you want to change file location, you have to enter path as second argument\n\t"
		"Remember that debagging output will be saved with programm result\n";
	std::getline(std::cin, ovar);

	//создание стандартного потока вывода
	if (ovar[0] == '1')
		of = 0;
	//создание файлового потока вывода с сопутствующей проверкой
	else if (ovar[0] == '2')
	{
		out.open(make_path(ovar, "D:/LAB1_SOURCE/OutputSource.txt"));
		if (!(out.is_open()))
		{
			std::cout << "Uncorrect path!!!";
			return;
		}
		of = 1;
	}
	//иные случаи
	else
	{
		std::cout << "Goodbye" << std::endl;
		return;
	}

	//считывание входных данных через консоль
	if (ivar[0] == '1')
	{
		std::string s;

		std::cout << "Enter string:\n";
		std::getline(std::cin, s);
		std::cout << std::endl;

		//проверка на отсутствие "лишних" символов
		if (!check_strElements(s))
		{
			std::cout << "Entered string: " + s + "\nResult: False - Uncorrect symbols";
			return;
		}

		//выбор потока вывода
		if (!of)
			call(s, std::cout);
		else
		{
			call(s, out);
			out.close();
		}
	}
	//считывание входных данных через файл
	else if (ivar[0] == '2')
	{
```

```
			std::string data_;

			//открытие потока
			std::ifstream f(make_path(ivar, "D:/LAB1_SOURCE/InputSource.txt"));

			//проверка на то, что поток действительно был открыт
			if (!(f.is_open()))
			{
				std::cout << "Uncorrect path!!!";
				return;
			}

			//считывание данных из файла
			std::getline(f, data_);

			//проверка на отсутствие "лишних" символов
			if (!check_strElements(data_))
			{
				std::cout << "Entered string: " + data_ + "\nResult: False - Uncorrect
symbols";
				return;
			}

			//выбор потока вывода
			if (!of)
				call(data_, std::cout);
			else
			{
				call(data_, out);
				out.close();
			}
		}
		//иные случаи
		else
		{
			std::cout << "Goodbye!";
			return;
		}

}

bool check_strElements(std::string s)
{
		for (char i : s)
			if (i != 'A' && i != 'B' && i != '(' && i != ')' && i != ' ')
				return 0;
		return 1;
}

std::string make_path(std::string var, std::string default_path)
{
		std::string path;
		std::string data_;

		//запись полученного пути в переменную path
		for (auto i = ++var.begin(); i != var.end(); i++)
			if (*i != ' ')
				path += *i;

		//проверка на то, что путь был введен
		if (path.empty())
			path = default_path;
		return path;
}
```

```
void call(std::string s, std::ostream& const out)
{
	std::string message = "Entered string: " + s;
	out << message << std::endl << "Result: " + bool_to_string(check(s, out)) <<
std::endl;
	out << "----------------------------------------------------------------------------
----------------------" << std::endl;
}

bool check(std::string& terminate, std::ostream& const out, size_t n)
{
	//Выход из рекурсии, если дальнейшая замена подстроки (B A A) на подстроку A
бесполезно
	if (terminate == "A")
		return 1;

	//Нахождение очередной подстроки (B A A)
	size_t it = terminate.find("(B A A)");

	//Проверка на наличие вышеуказанной подстроки
	if (it == std::string::npos)
		return 0;

	//Отладочные выводы
	out << "Value of the check-string after the next function call : " << terminate <<
std::endl << std::endl;
	out << "Recursion depth: " << n << std::endl << "------------------" << std::endl;

	//Удаление подстроки (B A A) из ввденной пользователем строки
	terminate.erase(it + terminate.begin(), it + terminate.begin() + 7);

	//Вставка на место удаленной подстроки символа A
	terminate.insert(it + terminate.begin(), 'A');

	//Рекурсивный вызов функции от измененной строки
	return check(terminate, out, ++n);
}

std::string bool_to_string(bool a)
{
	return a ? "True" : "False";
}

int main()
{
	std::cout << "Choose input format:\n\t1)Enter string\n\n\t2) Read from file (Default
file is located along the path:"
		"D:/LAB1_SOURCE/InputSource.txt)\n\tIf you want to change file location, you
have to enter path as second argument\n\t"
		"Don't forget to change all \'\\\' to \'/\'\n";
	//Чтение входной строки (использование getline обусловлено возможностью присутсвия
пробелов во входной строке)
	Dialog();

	return 0;

}
```