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:Рекурсия

Студент гр. 9304 _____ Краев Д.В.

Преподаватель _____ Фиалковский М.С.

Санкт-Петербург

2020

Цель работы.

Узнать, что такое рекурсия, и научиться использовать ее на практике.

Задание.

Построить синтаксический анализатор для параметризованного понятия скобки(Т), где Т–заданное конечное множество, а круглые скобки «(»и «)» не являются терминальными символами, а отражают зависимость опре-деляемого понятия от параметра Т.

скобки(Т)::=элемент(Т)|список(скобки(Т))

список(Е)::=N|[ряд(Е)]

ряд(Е)::=элемент(Е)|элемент(Е)ряд(Е)

Выполнение работы.

Для выполнения работы были созданы 4 функции: isBrackets, isArray, isList и isElem.

IsBrackets принимает на вход экземпляр класса string, содержащий строку, которую нужно проверить. Функция при помощи совершает обход по данной строке, проверяет каждый символ с помощью функций isList и isElem. Если хотя бы 1 символ не пройдет проверку, то функция isBrackets вернут строку «incorrect», в ином случае строку «correct».

IsList принимает на вход ссылку на итератор класса string. Если он находится на символе N, то функция возвращает true. Если итератор находится на символе [, то итератор переходит на следующий символ в строке и функция возвращает возвращаемое значение функции isArray, которой подали на вход итератор. Если итератор находится не на символе N и не на символе [, то функция возвращает false.

IsArray принимает на вход ссылку на итератор класса string. Функция совершает обход по строке до тех пор пока не встретит символ] или символ конца строки. Каждый она проверяет с помощью функций isElem и isList. Если обе функции возвращают false, то функция isArray возвращает false, в

ином случае продолжает обход. Если функция встретила символ конца строки, то она возвращает false. Если она встретила символ] , и до этого проверила как минимум 1 символ, то она вовращает true.

IsElem принимет на вход ссылку на итератор. И проверяет символ, на котором находится итератор, на вхождение в множество элементов. Если он не находится в данном множестве.

В функции main происходит ввод строки и проверка этой строки с помощью функции isBrackets.

Выводы

Изучено понятие рекурсия. Написан синтаксически анализатор для понятия скобки(T) с помощью рекурсии.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Файл: main.cpp

```
#include <iostream>
#include <string>

#define ELEMS "<>{}()"

std::string isBrackets(std::string str);
bool isArray(std::string::iterator &it);
bool isList(std::string::iterator &it);
bool isElem(std::string::iterator &it);



std::string isBrackets(std::string str){
	for(std::string::iterator it = str.begin();it != str.end();it++){
		if(!isElem(it) && !isList(it)){
			return "incorrect";
		}
	}
	return "correct";
}


bool isArray(std::string::iterator &it){
	int n = 0;
	while(*it != ']'){
		if(isElem(it) || isList(it)){
```

```
                it++;
            }else{
                return false;
            }
            if(*it == '\0'){
                return false;
            }
            n++;
    }
        if(n>=1){
            return true;
        }else{
            return false;
        }
}

bool isList(std::string::iterator &it){
    if(*it == 'N'){
            return true;
        }
        if(*it == '['){
            it++;
            return isArray(it);
        }
        return false;
}

bool isElem(std::string::iterator &it){
        std::string elems(ELEMS);
```

```
        for(int i = 0;i < elems.size();i++){
                if(*it == elems[i])
                        return true;
        }
        return false;
}


int main(){
        std::string str;
        std::cin >> str;
        std::cout << isBrackets(str) << '\n';
        return 0;
}
```

ПРИЛОЖЕНИЕ В

ТЕСТИРОВАНИЕ

Результаты тестирования представлены в таблице Б.1

Таблица Б.1 — Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	[[[]]]	incorrect
2.	<><>{}}{{}}{}{}))((((((	correct
3.	sofjsv3fj2230kcpwj2df2pof	incorrect
4.	<><><<<{{[>>]	corect
5.	<><><<<{{[>>]1	incorrect
6.	N	correct
7.	[[[[[[[]	incorrect
8.	[[[<><>><{}]]]	correct
9.	[NNNNN]())()({}{}<><>	correct
10.	Nadasdsad	incorrect