ММИНОБРНАУКИ 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: Взаимно-рекурсивные функции и процедуры. Синтаксический анализатор понятия скобок

Студент гр. 9384 _____ Нистратов Д.Г.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Разработать алгоритм обработки строки, с помощью взаимно-рекурсивной функции, для нахождения “скобки”.

Задание.

ВАРИАНТ 13.

Построить синтаксический анализатор для понятия скобки.

скобки::=A | скобка скобки

скобка::= (B скобки)

Выполнение работы.

Для реализации обработки строки, были написаны взаимно-рекурсивные функции brackets и braket:

```
bool brackets(std::string line, int& current_pos)
{
   std::cout << "<Скобки>" << std::endl;
   switch (line[current_pos])
   {
   case 'A':
      return true;
   case '(':
   {
      current_pos++;
      if (bracket(line, current_pos) == true)
         if (brackets(line, current_pos) == true)
            return true;
   }
   default:
         return false;
```

```
    }
}

bool bracket(std::string line, int& current_pos)
{
    std::cout << "<Скобка>" << std::endl;
    if (line[current_pos] == 'B')
    {
        current_pos++;
        if (brackets(line, current_pos) == true)
        {
            current_pos++;
            if (line[current_pos] == ')')
            {
                current_pos++;
                return true;
            }
        }
    }
    else
        return false;
}
```

На вход функции передается строка (string), а также значение, указывающее на позицию символа в строке. Передача происходит с помощью ссылки, для отслеживания позиции, при каждом проходе рекурсивного цикла.

Для отладки программы, были описаны вспомогательные строки вывода информации о заходе в рекурсию.

Функция read_type осуществляет запрос у пользователя о способе ввода данных. 1 – считывание данных из файла, 2 – считывание данных с консоли.

char read_type()

```
{
 std::cout << "Введите 1 для считывания из файла\nВведите 2 для считы
вания из консоли" << std::endl;
 char key = getchar();
 return key;
}
```

Считывание происходит построчно из файла input.txt, каждая строчка проверяется на соответствие синтаксиса (“(”, “)”, “A”, “B”), а также на верность “скобки”. Запись производиться в файл output.txt, с удалением последнего символа строки, если он не является EOF. Это необходимо для удаления символа переноса строки “/n/r” или “/n” (зависит от ОС).

```
//Считывание с файла

    std::ifstream in_file("input.txt");
    std::ofstream out_file("output.txt");
    std::string line;
    int i = 0;
    int line_number = 1;
    while (std::getline(in_file, line))
    {
       if (!synt_checker(line))
       {
          out_file << "Ошибка синтаксиса" << std::endl;
       }
       else
       {
          i = 0;
          if (line[line.length()-1] != EOF) line.pop_back();
          if (brackets(line, i) && line[i] == line.back()) out_file << "Выра
жение " << line << " - скобки" << std::endl;
```

```
                else out_file << "Выражение " << line << " -
не скобки" << std::endl;
                }
            }
            out_file.close();
            break;
        }
```

Считывание из консоли происходит с помощью метода стандартной библиотеки cin. Строка проверяется на соответсвие синтаксису (“(”, “)”, “A”, “B”), а также на верность “скобки”. Вывод производится в консоль.

```
            // Считывание с консоли

            std::string text;
            int i = 0;
            std::cout << "Введите значение" << std::endl;
            std::cin >> text;
            if (!synt_checker(text)){
                std::cout << "Ошибка синтаксиса" << std::endl;
                return;
            }
            if (brackets(text, i) && text[i] == text.back()) std::cout << "Выражен
ие " << text << " - скобки" << std::endl;
            else std::cout << "Выражение " << text << " -
не скобки" << std::endl;
            break;
```

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	(B)A	<Скобки> <Скобка> <Скобки> Выражение (B)A - не скобки	Выражения прошло 3 цикла рекурсии и вывело несоответствие “скобки”
2.	AAAA	<Скобки> Выражение AAAA - скобки	После 1 цикла рекурсии алгоритм завершил работу. Проверенное выражение - скобки
3.	AAA(B)SasddHH	Ошибка синтаксиса	Выражение содержит ошибку синтаксиса

Выводы.

В ходе выполнения лабораторный работы были изучены взаимно-рекурсивные функции и процедуры, а также, разработанная программа, выполняет синтаксический анализатор понятия скобок.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb1_13.cpp

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

bool bracket(std::string line, int& current_pos);
void solver(char read_type);

bool brackets(std::string line, int& current_pos)
{
   std::cout << "<Скобки>" << std::endl;
   switch (line[current_pos])
   {
   case 'A':
      return true;
   case '(':
   {
      current_pos++;
      if (bracket(line, current_pos) == true)
         if (brackets(line, current_pos) == true)
            return true;
   }
   default:
         return false;
   }
}

bool bracket(std::string line, int& current_pos)
{
```

```
            std::cout << "<Скобка>" << std::endl;
            if (line[current_pos] == 'B')
            {
                current_pos++;
                if (brackets(line, current_pos) == true)
                {
                    current_pos++;
                    if (line[current_pos] == ')')
                    {
                        current_pos++;
                        return true;
                    }
                }
            }
            return false;
        }

        char read_type()
        {
         std::cout << "Введите 1 для считывания из файла\nВведите 2 для считыва
ния из консоли" << std::endl;
         char key = getchar();
         return key;
        }

        bool synt_checker(std::string line)
        {
            // Syntaxis checker { A, B, (, ) }

            for(size_t i = 0; i < line.length()-1; i++)
```

```
        if (line[i] != 'A' && line[i] != 'B' && line[i] != '(' && line[i] != ')')
        {
            return false;
        }
    return true;
}

void solver(char read_type)
{
    switch (read_type)
    {
    case '1':
        {
            //Считывание с файла

            std::ifstream in_file("input.txt");
            std::ofstream out_file("output.txt");
            std::string line;
            int i = 0;
            int line_number = 1;
            while (std::getline(in_file, line))
            {
                if (!synt_checker(line))
                {
                    out_file << "Ошибка синтаксиса" << std::endl;
                }
                else
                {
                    i = 0;
                    if (line[line.length()-1] != EOF) line.pop_back();
```

```
                    if (brackets(line, i) && line[i] == line.back()) out_file << "Выраж
ение " << line << " - скобки" << std::endl;
                    else out_file << "Выражение " << line << " -
 не скобки" << std::endl;
                }
            }
            out_file.close();
            break;
        }
    case '2':
        {
            // Считывание с консоли

            std::string text;
            int i = 0;
            std::cout << "Введите значение" << std::endl;
            std::cin >> text;
            if (!synt_checker(text)){
                std::cout << "Ошибка синтаксиса" << std::endl;
                return;
            }
            if (brackets(text, i) && text[i] == text.back()) std::cout << "Выражени
e " << text << " - скобки" << std::endl;
            else std::cout << "Выражение " << text << " - не скобки" << std::endl;
            break;
        }
    default:
        std::cout<<"Числа нет в списке команд" << std::endl;
        break;
    }
```

```
}

int main()
{
    std::string text;
    solver(read_type());
}
```