МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по практической работе №0
по дисциплине «Алгоритмы и структуры данных»
Тема: Программирование рекурсивных алгоритмов

Студент(ка) гр. 7383 _____ Чемова К.А.

Преподаватель _____ Размочаева Н.В.

Санкт-Петербург
2018

Содержание:

Цель работы.

Познакомиться с основными понятиями и приемами рекурсивного программирования, получить навыки программирования рекурсивных процедур и функций на языке программирования C++.

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.

Постановка задачи.

Реализовать основанную на рекурсивном алгоритме программу – синтаксический анализатор понятия скобки:

скобки::=квадратные | круглые

квадратные:: = [[квадратные] (круглые)] | B

круглые::=((круглые) [квадратные]) | A

Реализация.

Были реализованы булевы функции Bracket(), Round(), Square() для проверки выражения на соответствие понятию *скобки*. Функция main() выводит меню для выбора способа ввода данных: 1 – для ввода с клавиатуры и 2 - для использования данных из файла (при невозможности открыть файл выводится

сообщение об ошибке). Далее вызывается Bracket(), в которой происходит вызов Square() и Round(). Последние две функции определены через самих себя и через друг друга, что делает их рекурсивными. Square() проверяет выражение на соответствие выражения понятию квадратные скобки, Round(), в свою очередь, на – круглые скобки. После проверки, выводится сообщение о результате работы программы. Если выражение оказалось скобками, то выводится само выражение и фраза «Это скобки». В противном случае, выводится сообщение об ошибке и выражение до того места, в котором было найдено не соответствие.

Тестирование.

Программа собрана в операционной системе Ubuntu 16.04 LTS с использованием компилятора g++. В других ОС и компиляторах тестирование не проводилось.

Выводы.

В ходе лабораторной работы было изучено понятие рекурсии и освоены основные принципы рекурсивного программирования, написана программа, позволившая овладеть навыки программирования рекурсивных процедур и функций на языке программирования C++.

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

```
/* Вариант 21
   Построить синтаксический анализатор для понятия скобки.
   скобки::=квадратные | круглые
   квадратные::=[[кваратные](круглые)] | B
   круглые::=((круглые)[квадратные]) | A
*/

#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;

bool Bracket(ifstream &infile);
bool Round(ifstream &infile, char ch);
bool Square(ifstream &infile, char ch);
void Error(short k);

int main() {

   int n;
   bool br;
   char data[1000];
   FILE *f;
   cout<<"\nВас приветствует анализатор скобок!\n\nВыберите способ
ввода данных:\n1 - Для ввода данных с клавиатуры\n2 - Для
использования данных из файла\n0 - Для выхода из программы"<<endl;
   cin>>n;

   switch(n) {

   case 0:

      return 0;

   case 1:
   {
      /*       f = fopen("test.txt", "w"); //создание файла
            cin»data;
            fputs(data, f);
            fclose(f);
```

```
        ifstream infile("test.txt");
        if(!infile) cout«"Входной файл не может быть открыт!"«endl;
        else {
           br = Bracket(infile);
           cout«endl;
           if (br) cout«"Это скобки"«endl;
           else cout«"Это НЕ скобки!"«endl;
        }
   */

   f = fopen("test.txt", "w");
   cin>>data;
   fputs(data,f);
   fclose(f);

   ifstream infile("test.txt");
   if(!infile) {
      cout<< "Входной файл не открыт!"<<endl;
      return 0;
   }

   br = Bracket(infile);
   infile.close();

   cout<<endl;

   if (br) cout<<"Это скобки!"<<endl;
   else cout<<"Это НЕ скобки!"<<endl;

   remove("test1.txt"); // удаление файла
   return 0;
}
case 2:
{
   ifstream infile("test1.txt");
   if(!infile) {
      cout<< "Входной файл не открыт!"<<endl;
      break;
   }

   br = Bracket(infile);
   infile.close();

   cout<<endl;
```

```
        if (br) cout<<"Это скобки!"<<endl;
        else cout<<"Это НЕ скобки!"<<endl;

        return 0;
    }
    }
}

bool Square(ifstream &infile, char ch) {

// квадратные::=[[кваратные](круглые)] | B
// ch - текущий символ входной строки

    if (ch == 'B') {
        cout<<ch;
        return true;
    }

    else if (ch == '[') {
        cout<<ch;
        infile>>ch;
        if (ch == '[') {
            cout<<ch;
            infile>>ch;
            if (Square(infile,ch)) {
                infile>>ch;
                if (ch == ']') {
                    cout<<ch;
                    infile>>ch;
                    if (ch == '(') {
                        cout<<ch;
                        infile>>ch;
                        if (Round(infile, ch)) {
                            infile>>ch;
                            if (ch == ')') {
                                cout<<ch;
                                infile>> ch;
                                if (ch == ']') {
                                    cout<<ch;
                                    return true;
                                }
                                else {
                                    Error(2);
                                    return false;
                                }
```

```
                    }
                    else {
                        Error(4);
                        return false;
                    }
                }
                else return false;
            }
            else {
                Error(3);
                return false;
            }
        }
        else {
            Error(2);
            return false;
        }
    }
    else return false;
}
else {
    Error(1);
    return false;
}
}
else {
    return false;
}
}

bool Round(ifstream &infile, char ch) {

// круглые::=((круглые)[квадратные]) | A
// ch - текущий символ входной строки

    if (ch == 'A') {
        cout<<ch;
        return true;
    }

    else if (ch == '(') {
        cout<<ch;
        infile>>ch;
        if (ch == '(') {
            cout<<ch;
```

```
    infile>>ch;
    if (Round(infile, ch)) {
        infile>>ch;
        if (ch == ')') {
            cout<<ch;
            infile>>ch;
            if (ch == '[') {
                cout<<ch;
                infile>>ch;
                if (Square(infile, ch)) {
                    infile>>ch;
                    if (ch == ']') {
                        cout<<ch;
                        infile>>ch;
                        if (ch == ')') {
                            cout<<ch;
                            return true;
                        }
                        else {
                            Error (4);
                            return false;
                        }
                    }
                    else {
                        Error(2);
                        return false;
                    }
                }
                else return false;
            }
            else {
                Error(1);
                return false;
            }
        }
        else {
            Error(4);
            return false;
        }
    }
    else return false;
}
else {
    Error(3);
```

```
            return false;
        }
    }
    else {
        //      Error(3);
        return false;
    }
}

bool Bracket(ifstream &infile) {

// скобки::= квадратные | круглые

    char ch;

    if (infile>>ch) {
        if (Square(infile, ch)) return true;
        else if (Round(infile, ch)) return true;
        else return false;
    }
    else Error(0);

}

void Error(short k) {

    cout<< endl << "Ошибка №" << k << ": ";

    switch(k) {
    case 0:
        cout<< "Пустая входная строка." <<endl;
        break;
    case 1:
        cout<< "Не хватает '['." <<endl;
        break;
    case 2:
        cout<< "Не хватает ']'." <<endl;
        break;
    case 3:
        cout<< "Не хватает '('."<<endl;
        break;
    case 4:
        cout<< "Не хватает ')'." <<endl;
        break;
    }
```

```
    return;
}
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ ПРОГРАММЫ

Входные данные	**Выходные данные**
A	A Это скобки!
B	B Это скобки!
((A)[B])	((A)[B]) Это скобки!
[[B](A)]	[[B](A)] Это скобки!
((A)[[[B](A)]])	((A)[[[B](A)]]) Это скобки!
(((((()))))))	((((((Ошибка №3: Не хватает '('. Это НЕ скобки!
((A))	((A) Ошибка №1: Не хватает '['. Это НЕ скобки!