

МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ РФ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»  
Кафедра вычислительных систем

КУРСОВАЯ РАБОТА  
по дисциплине «Технологии разработки программного обеспечения»  
на тему  
«Программа символы: проверка сбалансированности скобок в программе на C++»

Выполнил: Ст.гр.ИП-211  
Сократов М.В.  
Проверил:  
Ст. преподаватель Токмашева Е. И

Новосибирск, 2023

## Содержание

Введение и постановка задачи .....	3
Техническое задание .....	4
Описание выполненного проекта .....	5
Личный вклад в проект .....	11
Приложение. Текст программы.....	11

## **Введение и постановка задачи**

Нами выбрана тема “Программа символы: проверка сбалансированности скобок в программе на C++”, потому что мне интересен процесс создания приложения, и я хотел бы лучше разобраться в технологии создания функций. В работе будут реализованы следующие функции:

1. Функция «меню».
2. Функция записи из файла расширения .cpp в файл расширения .txt.
3. Функция нахождения скобок в файле
4. Функция обнаружения не сбалансированных скобок
5. Вывод информации о правильности кода или о том, в каких строках не сбалансированные скобки

Для этого нужно решить следующие задачи:

1. Изучить язык.
2. Реализовать вышеперечисленные функции.

После реализации можно провести тесты приложения.

## Техническое задание

Реализовать программу.

Функционал: Ввод пользователем полного пути проверяемого файла или же ввод текста проверяемой программы самостоятельно.

Клиентская часть. Для всего использовать язык программирования C++.

1. Выбор пользователем одной из 3 вариантов: ввод полного пути к проверяемому файлу, ввод текста проверяемой программы самостоятельно, выход из программы.

1.1 Если был выбран первый вариант, то пользователь вводит в консоль полный путь к файлу.

1.2 Если был выбран второй вариант, то пользователь вводит в консоль код программы, которую он хочет проверить.

1.3 Если был выбран третий вариант, то программа завершает работу.

1.4 Если были введены некорректные данные, то программа выдаст ошибку и попросит ввести данные повторно.

2. Запись из файла расширения .crr в файл расширения .txt «если был выбран 1 вариант».

3. Считать файл .txt и найти в нём все скобки.

4. Повторное считывание файла и проверка на корректные скобки.

4.1 Вывод о сбалансированности всех скобок.

4.2 Вывод строк, в которых есть несбалансированные скобки.

5. Написать тесты.

Проверяемая программа не рассматривается с точки зрения правильности кода.

## Описание выполненного проекта

```
string window() {
    setlocale(LC_ALL, "RUS");
    char ch;
    string st, str;
    ofstream file("file.txt");
    while (1) {
        cout << "\nМеню :";
        cout << "\nВведите 1 для ввода полного адреса файла ";
        cout << "\nВведите 2 для ввода кода в консоль ";
        cout << "\nВведите 0 для выхода из программы\n";

        zn:
        cin >> ch;
        switch (ch) {
            case '1':
                cout << "\nВведите полный адрес файла(в конце нажать Enter -> ctrl+x -> Enter для Windows или Enter -> ctrl+d для Linux) : ";
                while (getline(cin, str)) {
                    st = st + str;
                }

                file.close();
                return st;
            case '2':
                cout << "\nВведите код в консоль(для окончания ввода нажмите Enter -> ctrl+x -> Enter для Windows или Enter -> ctrl+d для Linux): \n";
                while (getline(cin, st)) {
                    file << st << endl;
                }
                file.close();
                return "0";
            case '0':
                return "-1";
                break;
            default:
                cout << "\nВвод неверных данных, повторите попытку снова ";
                goto zn;
        }
    }
}
```

В данной части кода описана функция “Window” отвечающая за создания контекстного меню. Переменная `ch` типа `char` отвечает за выбор поля меню, переменные `st` и `str` отвечают за ввод текста, переменная `file` отвечает за работу с текстовым файлом “file.txt”.

В цикле `While` сначала выводятся параметры меню, потом пользователь вводит в консоль символ, который сохраняется в переменную `ch`, открывается файл “file.txt”.

В конструкции `switch` с ключом `ch`, в зависимости от введенного символа, выполняются следующие действия: ввод в переменную `st` через переменную `str` полного пути к файлу, закрытие `file` и возврат значения ‘2’; ввод в файл “file.txt” текста через переменную `st`, закрытие файла и возврат значения ‘0’; возврат значения ‘-1’; при некорректно введенном символе выводится соответствующее сообщение и происходит возврат к флагу `zn`.

```
int wrote(string str)
{
    std::string line;
    std::ofstream out;
    std::ifstream in(str);
    out.open("file.txt");
    if (in.is_open())
    {
        while (std::getline(in, line))
        {
            out << line << std::endl;
        }
    }
    out.close();
    std::cout << "File C++ is writting in file.txt" << std::endl;
    in.close();
    return 0;
}
```

В данной части программы описана функция “wrote” отвечающая за запись из файла .cpp в file.txt, в качестве передаваемого параметра идет адрес файла .cpp записанный в переменную str. Переменная line типа string отвечает за перезапись текста, переменная out отвечает за работу с файлом “file.txt”, переменная in отвечает за работу с файлом .cpp.

Если файл открылся, то происходит считывание текста из in в строку line, а потом запись line в out. Далее оба файла закрываются.

```
int cheking() {
    ifstream file("file.txt");    // opening a file

    if (!file.is_open()) {
        cerr << "file can't be open" << endl;
        return 1;
    }

    int count = 0;
    char ch;
    // "{["
    int ct = 0; int ct1 = 0;
    //counting brackets not contained between the characters " and '
    while (file.get(ch)) {
        //Checking for the closeness of quotation marks
        if (ch == char(0x22)) {
            if (ct1 == 0)ct1++;
            else { ct1 = 0; }
        }
        if (ch == char(0x27)) {
            if (ct == 0)ct++;
            else { ct = 0; }
        }
        //Searching for brackets and entering them into the Skobki array
        if (ch == '{' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == '}' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == '[' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == ']' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == '(' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == ')' and ct == 0 and ct1 == 0) {
            count++;
        }
    }
    cout << "Quantity '{}[]': " << count << endl;

    file.close();
    return count;
}
```

В данной части программы описана функция “cheking”, в которой происходит подсчет скобок. Переменная count типа int отвечает за подсчёт скобок в файле, переменные ct и ct1 типа int отвечают за подсчёт одинарных и двойных кавычек, переменная ch типа char отвечает за считывание символов из файла.

В цикле while сначала идет проверка на близость кавычек к скобкам, а потом подсчет скобок, не стоящих внутри кавычек.

Далее идет вывод количества скобок, закрытии файла возвращение количества скобок через переменную count.

```

int searching(int n) {
    ifstream file("file.txt");    // opening a file

    if (!file.is_open()) {
        cerr << "file can't be open" << endl;
        return 1;
    }

    char ch;
    // "{"
    int chetstr = 0;
    int chet = 0; int ct = 0; int ctl = 0;
    //counting brackets not contained between the characters " and '

    string line;
    char* Skobki = new char[n];
    int* indexi = new int[n];
    while (file.get(ch)) {
        //Checking for the closeness of quotation marks
        if (ch == char(0x22)) {
            if (ctl == 0)ctl++;
            else { ctl = 0; }
        }
        if (ch == char(0x27)) {
            if (ct == 0)ct++;
            else { ct = 0; }
        }
        if (ch == '\\n') {
            ++chetstr;
        }
        if (ch == '{' and ct == 0 and ctl == 0) {
            Skobki[chet] = '{';
            indexi[chet] = chetstr;
            chet++;
        }
        if (ch == '}' and ct == 0 and ctl == 0) {
            Skobki[chet] = '}';
            indexi[chet] = chetstr; chet++;
        }
        if (ch == '[' and ct == 0 and ctl == 0) {
            Skobki[chet] = '[';
            indexi[chet] = chetstr; chet++;
        }
        if (ch == ']' and ct == 0 and ctl == 0) {
            Skobki[chet] = ']';
            indexi[chet] = chetstr; chet++;
        }
        if (ch == '(' and ct == 0 and ctl == 0) {
            Skobki[chet] = '(';
            indexi[chet] = chetstr; chet++;
        }
        if (ch == ')' and ct == 0 and ctl == 0) {
            Skobki[chet] = ')';
            indexi[chet] = chetstr; chet++;
        }
    }
}

```

```

for (int i = 0; i < n - 1; i++) {
    for (int j = 1; j < n; j++) {
        if (Skobki[i] != '0') {
            if (Skobki[i] == '{' and Skobki[j] == '}') {
                Skobki[i] = '0';
                Skobki[j] = '0';
                break;
            }
            if (Skobki[i] == '[' and Skobki[j] == ']') {
                Skobki[i] = '0';
                Skobki[j] = '0';
                break;
            }
            if (Skobki[i] == '(' and Skobki[j] == ')') {
                Skobki[i] = '0';
                Skobki[j] = '0';
                break;
            }
        }
    }
}

int chetTrue = 0;
for (int i = 0; i < n; i++) {
    if (Skobki[i] != '0') {
        chetTrue++;
        if (chetTrue == 1) cout << "Kod ne norm" << endl << "Bad line:";
        cout << indexi[i];
    }
}

if (chetTrue == 0) cout << "Kod norm";
file.close();
return 0;
}

```

В данном фрагменте кода описана функция searching. Отвечающая за определение сбалансированности скобок. Переменная ch типа char отвечает за считывание символов из файла, переменные ct и ct1 типа int отвечают за подсчёт одинарных и двойных кавычек, переменная chetstr типа int отвечает за подсчет строк, переменная chet типа int отвечает за подсчет скобок, массивы Skobki и indexi типа int отвечают за расположение скобок в строках.

В цикле while идет распределение скобок по строкам. В первом цикле for идет проверка на сбалансированные скобки и зануление их. Во втором цикле for идет подсчет некорректных скобок и вывод строк, в которых они были обнаружены (если таковые имеются). В конце идет вывод сообщения, что в коде все скобки сбалансированы, если это так, и происходит закрытие файла.

```

int main() {
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    cout << "privet hozain, now i work" << endl;
    string sf = window();
    if (sf == "-1") {
        cout << "Bye Bye" << endl;
        return 0;
    }
    else if (sf != "0") {
        wrote(sf);
    }
    int countmemory = cheking();
    searching(countmemory);
}

```

В данном фрагменте кода описана заглавная функция main, в которой происходит вызов всех вторичных функций. Сначала идет вызов функции “window”, для вывода меню. Затем в зависимости от возвращаемого значения идет или закрытие программы, или запись из



файла .cpp с заданным пользователем адресом в файл file.txt. Далее в переменную countmemory записывается возвращаемое значение из функции cheking и в конце происходит вызов функции searching.

## Тесты:

Тесты затронули правильность работы алгоритма: были созданы три файла формата .txt в которых были записаны “коды программ”, и эти три файла проходили проверку поочерёдно, а затем выводился результат.

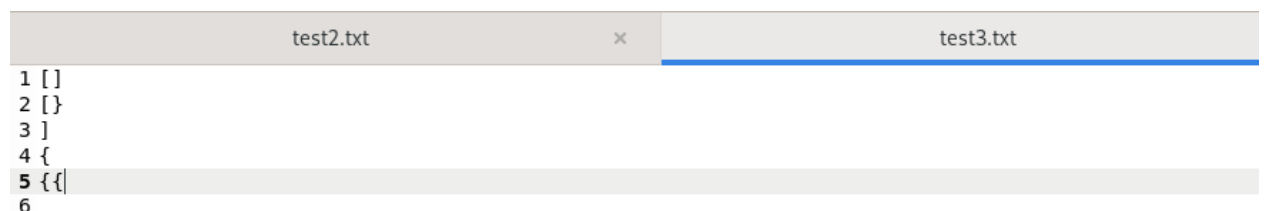
Содержимое файла test1.txt для первого теста



Содержимое файла test2.txt для второго теста

```
1#include "String.h"
2#include <fstream>
3#include <iostream>
4#include <string>
5
6using namespace std;
7
8string window()
9{
10    setlocale(LC_ALL, "RUS");
11    char ch;
12    string st;
13    string str;
14    ofstream file("file.txt");
15    while (1) {
16        cout << "\nМеню .:";
17        cout << "\nВведите 1 для ввода полного адреса файла ";
18        cout << "\nВведите 2 для ввода кода в консоль ";
19        cout << "\nВведите 0 для выхода из программы\n";
20        zn:
21        cin >> ch;
22        switch (ch) {
23            case '1':
24                cout << "\nВведите полный адрес файла(не забудьте заменить 1'\\' "
25                    << "на 2'\\\\' и в конце нажать Enter -> ctrl+x -> Enter для "
26                    << "Windows или Enter -> ctrl+d -> Enter) : ";
27                while (getline(cin, str)) {
28                    st = st + str;
29                }
30                cout << st;
31                file.close();
32                return st;
33            case '2':
34                cout << "\nВведите код в консоль(для окончания ввода нажмите Enter "
35                    << "-> ctrl+x -> Enter для Windows или Enter -> ctrl+d -> "
36                    << "Enter):\n";
37                while (getline(cin, st)) {
38                    file << st << endl;
39                }
40                file.close();
41                return "0";
42            case '0':
43                return "0";
44            break;
45            default:
46                cout << "\nВвод неверных данных, повторите попытку снова ";
47                goto zn;
48        }
49    }
50}
```

Содержимое файла test3.txt для третьего теста



## Результат выполнения тестов

```
TEST 1/3 searching:test1 Quantity '{}[]': 0
Your code is good
[OK]
TEST 2/3 searching:test2 Quantity '{}[]': 0
Your code is good
[OK]
TEST 3/3 searching:test3 Quantity '{}[]': 0
Your code is good
[OK]
RESULTS: 3 tests (3 ok, 0 failed, 0 skipped) ran in 0.2 ms
make[1]: выход из каталога «/home/sokrat1798/cw-ip-211_searching_unbalanced_brackets»
make runap
make[1]: вход в каталог «/home/sokrat1798/cw-ip-211_searching_unbalanced_brackets»
./app
```

## Личный вклад в проект:

Написал функцию “window”, частично участвовал в написании остальных функций. Работа с документацией.

## Приложения. Текст программы.

Main.cpp

```
#include <fstream>
#include <string>
#include <iostream>
#include <Windows.h>
#include <string>
#include <search.h>

using namespace std;
int searching(int n);
int cheking();
int wrote(string str);
string window();
int main() {
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    cout << "privet hozain, now i work" << endl;
    string sf = window();
    if (sf == "-1") {
        return 0;
        cout << "Bye Bye" << endl;
    }
    else if (sf != "0") {
        wrote(sf);
    }
    int countmemory = cheking();
    searching(countmemory);
}
```

Search.cpp

```
#include <fstream>
#include <string>
#include <iostream>
#include <string>
using namespace std;
int cheking() {
    ifstream file("file.txt");    // opening a file

    if (!file.is_open()) {
        cerr << "file can't be open" << endl;
        return 1;
    }

    int count = 0;
    char ch;
    // "{"
    int ct = 0; int ct1 = 0;
    //counting brackets not contained between the characters " and '
    while (file.get(ch)) {
        //Checking for the closeness of quotation marks
        if (ch == char(0x22)) {
            if (ct1 == 0)ct1++;
            else { ct1 = 0; }
        }
    }
```

```

        if (ch == char(0x27)) {
            if (ct == 0) ct++;
            else { ct = 0; }
        }
        //Searching for brackets and entering them into the Skobki array
        if (ch == '{' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == '}' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == '[' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == ']' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == '(' and ct == 0 and ct1 == 0) {
            count++;
        }
        if (ch == ')' and ct == 0 and ct1 == 0) {
            count++;
        }
    }
    cout << "Quantity '{}[]': " << count << endl;

    file.close();
    return count;
}

int searching(int n) {
    ifstream file("file.txt");    // opening a file

    if (!file.is_open()) {
        cerr << "file can't be open" << endl;
        return 1;
    }
    char ch;
    // "{"
    int chetstr = 0;
    int chet = 0; int ct = 0; int ct1 = 0;
    //counting brackets not contained between the characters " and '

    char* Skobki = new char[n];
    int* indexi = new int[n];
    while (file.get(ch)) {
        //Checking for the closeness of quotation marks
        if (ch == char(0x22)) {
            if (ct1 == 0) ct1++;
            else { ct1 = 0; }
        }
        if (ch == char(0x27)) {
            if (ct == 0) ct++;
            else { ct = 0; }
        }
        if (ch == '\n') {
            ++chetstr;
        }
        if (ch == '{' and ct == 0 and ct1 == 0) {
            Skobki[chet] = '{';
            indexi[chet] = chetstr;
            chet++;
        }
        if (ch == '}' and ct == 0 and ct1 == 0) {
            Skobki[chet] = '}';

```

```

        indexi[chet] = chetstr; chet++;
    }
    if (ch == '[' and ct == 0 and ct1 == 0) {
        Skobki[chet] = '[';
        indexi[chet] = chetstr; chet++;
    }
    if (ch == ']' and ct == 0 and ct1 == 0) {
        Skobki[chet] = ']';
        indexi[chet] = chetstr; chet++;
    }
    if (ch == '(' and ct == 0 and ct1 == 0) {
        Skobki[chet] = '(';
        indexi[chet] = chetstr; chet++;
    }
    if (ch == ')' and ct == 0 and ct1 == 0) {
        Skobki[chet] = ')';
        indexi[chet] = chetstr; chet++;
    }
}
//Search for paired brackets and their pairwise removal from the Skobki array
for (int i = 0; i < n - 1; i++) {
    for (int j = i; j < n; j++) {
        if (Skobki[i] != '0') {
            if (Skobki[i] == '{' and Skobki[j] == '}') {
                Skobki[i] = '0';
                Skobki[j] = '0';
                break;
            }
            if (Skobki[i] == '[' and Skobki[j] == ']') {
                Skobki[i] = '0';
                Skobki[j] = '0';
                break;
            }
            if (Skobki[i] == '(' and Skobki[j] == ')') {
                Skobki[i] = '0';
                Skobki[j] = '0';
                break;
            }
        }
    }
}
int chetTrue = 0;
for (int i = 0; i < n; i++) {
    if (Skobki[i] != '0') {
        chetTrue++;
        if (chetTrue == 1) cout << "Kod ne norm" << endl << "Bad line:";
        cout << indexi[i];
    }
}
if (chetTrue == 0) cout << "Kod norm";
file.close();
return 0;
}

```

Console.cpp

```

#include <iostream>
#include <string>
#include <fstream>

using namespace std;

string window() {

```

```

setlocale(LC_ALL, "RUS");
char ch;
string st, str;
ofstream file("file.txt");
while (1) {
    cout << "\nМеню :";
    cout << "\nВведите 1 для ввода полного адреса файла ";
    cout << "\nВведите 2 для ввода кода в консоль ";
    cout << "\nВведите 0 для выхода из программы\n";

zn:
    cin >> ch;
    switch (ch) {
        case '1':
            cout << "\nВведите полный адрес файла(в конце нажать Enter ->
ctrl+x -> Enter для Windows или Enter -> ctrl+d для Linux) : ";
            while (getline(cin, str)) {
                st = st + str;
            }

            file.close();
            return st;
        case '2':
            cout << "\nВведите код в консоль(для окончания ввода нажмите
Enter -> ctrl+x -> Enter для Windows или Enter -> ctrl+d для Linux): \n";
            while (getline(cin, st)) {
                file << st << endl;
            }
            file.close();
            return "0";
        case '0':
            return "-1";
            break;
        default:
            cout << "\nВвод неверных данных, повторите попытку снова ";
            goto zn;
    }
}
}

```

writing.cpp

```

#include <fstream>
#include <string>
#include <iostream>
#include <Windows.h>
#include <string>

using namespace std;

int wrote(string str)
{
    std::string line;
    std::ofstream out;
    std::ifstream in(str);
    out.open("file.txt");
    if (in.is_open())
    {
        while (std::getline(in, line))
        {
            out << line << std::endl;
        }
    }
    out.close();
    std::cout << "File C++ is writting in file.txt" << std::endl;
    in.close();
}

```

```
return 0;}
```