

Міністерство освіти і науки України  
Національний технічний університет України «Київський політехнічний  
інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1 з  
дисципліни «Основи програмування2».  
Основи комп'ютерних систем»

«Текстові файли»

Варіант 14

Виконав студент ПІ-13, Замковий Дмитро Володимирович  
(шифр, прізвище, ім'я, по батькові)

Перевірів Вечерковська Анастасія  
(прізвище, ім'я, по батькові)

## Лабораторна робота 1

### Текстові файли

**Мета заняття:** вивчити особливості і обробки текстових файлів даних.

### Завдання

Варіант 14: Створити текстовий файл, що містить програму на мові C++.  
Перевірити текст на рівну кількість відкритих і закритих дужок, вважаючи, що кожен оператор в програмі займає не більше одного рядка вихідного файлу.  
Вивести вміст файлу і результат аналізу

### Хід виконання

#### 1)Реалізація на мові програмування C++

CppLab.cpp:

```
#include "Header.h"

//Створити текстовий файл, що містить програму на мові C++. Перевірити текст
//на рівну кількість відкритих і закритих дужок, вважаючи, що кожен оператор в
//програмі займає не більше одного рядка вихідного файлу. Вивести вміст файлу і
//результат аналізу

int main()
{
    string path = "myFile.txt";
    f_read(path);
    f_write(path);
    f_check(path);
}
```

Header.h:

```
#pragma once
#include <iostream>
#include <string>
#include <fstream>
using namespace std;

void f_write(string path);
void f_read(string path);
void f_check(string path);
```

Source.cpp:

```
#include "Header.h"

int characters_per_line(string s1, string s2);

void f_read(string path) {
    ifstream f(path);
    if (f.is_open()) {
        cout << "Text in file:" << endl;
```

```

    string line;
    while (!f.eof())
    {
        getline(f, line);
        cout << line << endl;
    }
    f.close();
    cout << "End of file";
}
else
{
    cout << "Error: unable to read file";
}
}

void f_write(string path) {
    string mode;
    cout << "\n\nDo you want to clear the file?" << endl
        << "y - for yes" << endl << "n - for no" << endl;
    cin >> mode;
    while (mode != "y" and mode != "n")
    {
        cout << "ERROR: invalid value" << endl;
        cin >> mode;
    }

    if (mode == "y")
    {
        ofstream f(path, ofstream::out|ofstream::trunc);
        if (f.is_open()) {
            cout << "\nEnter text" << endl
                << "<ENTER> - to enter a new line" << endl
                << "<Ctrl+Z> - to complete the input" << endl;
            string line;
            while (getline(cin, line, '\26')) //<Ctrl+Z>
            {
                f << line;
            }
            f.close();
        }
        else
        {
            cout << "ERROR: could not open file";
        }
    }
    else
    {
        ofstream f(path, ofstream::app);
        if (f.is_open()) {
            string line;
            cout << "\nEnter text" << endl
                << "<ENTER> - to enter a new line" << endl
                << "<Ctrl+Z> - to complete the input" << endl;
            while (getline(cin, line, '\26')) //<Ctrl+Z>
            {
                f << line;
            }
            f.close();
        }
        else
        {
            cout << "ERROR: could not open file";
        }
    }
}

```

```
}

void f_check(string path)
{
    int braces[8] = {};
    string line;
    ifstream f(path);
    if (f.is_open())
    {
        cout << "Verified text: " << endl;
        while (!f.eof())
        {
            getline(f, line);
            braces[0] = characters_per_line(line, "(");
            braces[1] = characters_per_line(line, ")");
            braces[2] = characters_per_line(line, "{");
            braces[3] = characters_per_line(line, "}");
            braces[4] = characters_per_line(line, "[");
            braces[5] = characters_per_line(line, "]");
            braces[6] = characters_per_line(line, "<");
            braces[7] = characters_per_line(line, ">");

            bool ifTrue = braces[0] == braces[1] && braces[2] == braces[3] &&
braces[4] == braces[5] && braces[6] == braces[7];
            if (ifTrue)
            {
                cout << line << " - " << "All good!" << endl;
            }
            else
            {
                cout << line << " - " << "Somethink bad" << endl;
            }
        }
        f.close();
    }
    else
    {
        cout << "ERROR: unable to read file";
    }
}

int characters_per_line(string s1, string s2)
{
    int cnt = 0;
    for (int i = 0; i < s1.length(); ++i)
    {
        if (s1[i] == s2[0])
        {
            if (s1.substr(i, s2.length()) == s2)
            {
                ++cnt;
                i += s2.length() - 1;
            }
        }
    }
    return cnt;
}
```

### 2) Реалізація на мові програмування Python

main1.py:

```
import moduleLab1 as mod
```

```
def main():
    path = r'C:\Users\Dima\source\PC\OP\Lab1\text.txt'
    with open(path) as file:
        print('Previous text from file:',
              '=====', sep='\n')
        mod.output(path),
        print('=====')
        mod.config(path)
        print('\nVerified text')
        mod.check(path)

if __name__ == '__main__':
    main()
```

### moduleLab1.py

```
def file_input(path, mode):
    f = open(path, mode)
    line = ''
    key = 'q' # chr(26)
    while line != key:
        f.write(line + '\n')
        line = input()
    f.close()

def config(path):
    print('\nRecord:',
          '=====',
          'a - to append file',
          'w - to rewrite file',
          '=====', sep='\n')
    mode = input('Enter recording mode: ')
    while mode != 'a' and mode != 'w':
        print('ERROR: incorrect value')
        mode = input('Enter recording mode: ')

    print('\nEnter text:',
          '=====',
          'Press <ENTER> to end line',
          'Press <Ctrl+Z> to finish entering text',
          '=====', sep='\n')
    file_input(path, mode)

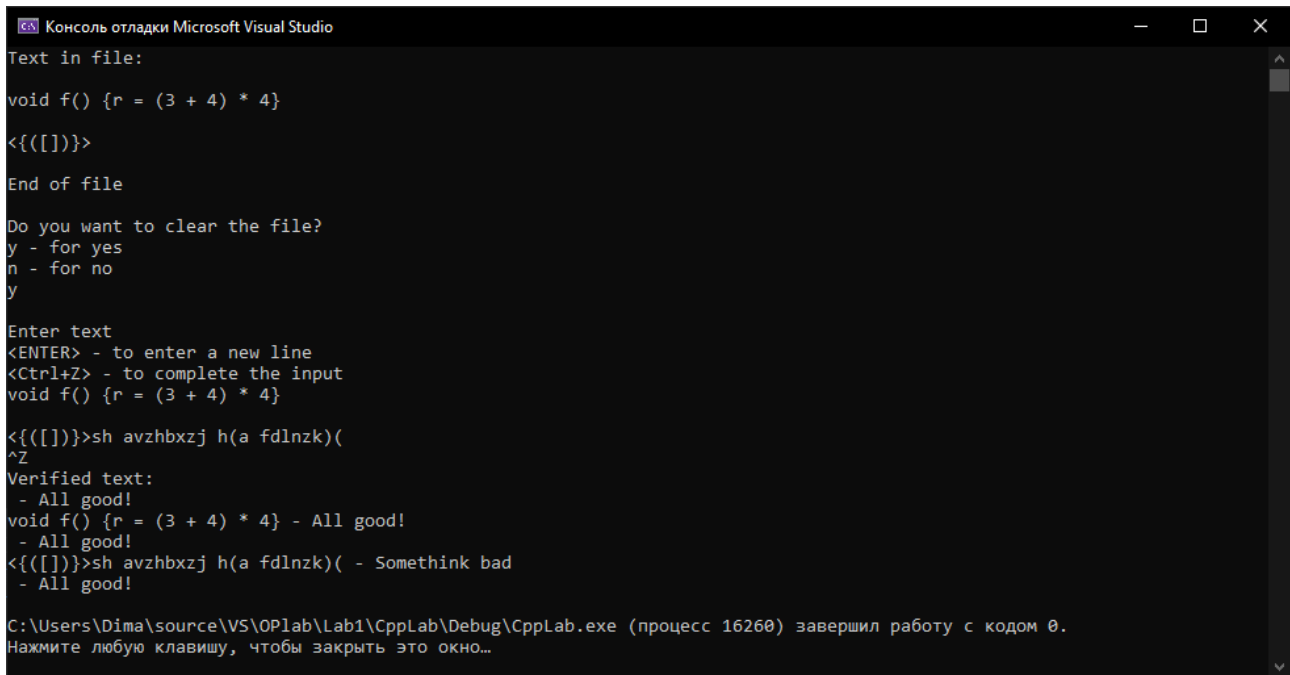
def output(path):
    f = open(path, 'r')
    print(f.read())
    f.close()

def check(path):
    f = open(path, 'r')
    lines = f.readlines()
    for i in range(0, len(lines)):
        line = lines[i]
        if line.count('(') == line.count(')') and line.count('{') ==
line.count('}') and line.count('[') == line.count(']') and line.count('<') ==
line.count('>'):
            print(line[:len(line) - 1], ' - ', 'All good!', sep='')
        else:
```

```
print(line[:len(line) - 1], ' - ', 'Somethink bad', sep='')  
f.close()
```

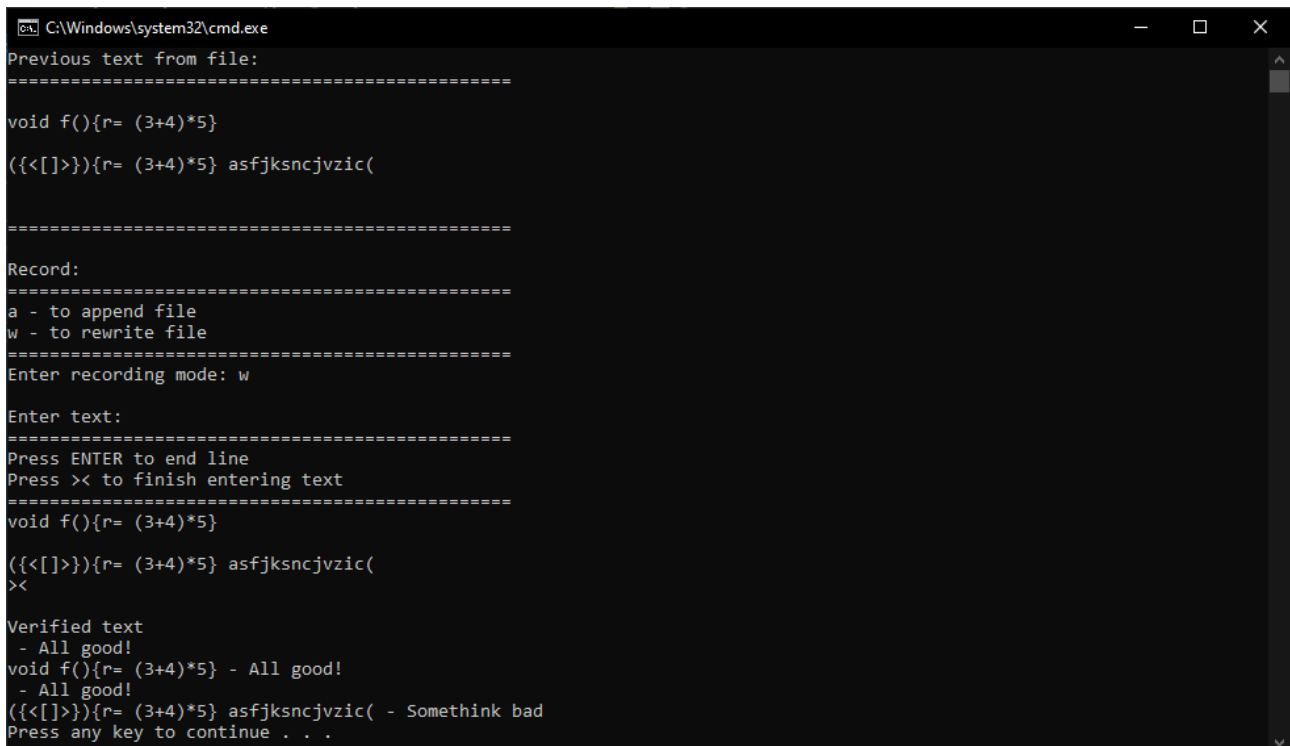
## Результати

C++:



```
Консоль отладки Microsoft Visual Studio  
Text in file:  
void f() {r = (3 + 4) * 4}  
<{([ ])}>  
End of file  
Do you want to clear the file?  
y - for yes  
n - for no  
y  
Enter text  
<ENTER> - to enter a new line  
<Ctrl+Z> - to complete the input  
void f() {r = (3 + 4) * 4}  
<{([ ])}>sh avzhbxzj h(a fdlnzk)(  
^Z  
Verified text:  
- All good!  
void f() {r = (3 + 4) * 4} - All good!  
- All good!  
<{([ ])}>sh avzhbxzj h(a fdlnzk)( - Somethink bad  
- All good!  
C:\Users\Dima\source\VS\OPlab\Lab1\Cpplab\Debug\Cpplab.exe (процесс 16260) завершил работу с кодом 0.  
Нажмите любую клавишу, чтобы закрыть это окно...
```

Python:



```
C:\Windows\system32\cmd.exe  
Previous text from file:  
=====  
void f(){r= (3+4)*5}  
(({<[ ]>}){r= (3+4)*5} asfjksncjvzic(  
=====  
Record:  
=====  
a - to append file  
w - to rewrite file  
=====  
Enter recording mode: w  
Enter text:  
=====  
Press ENTER to end line  
Press >< to finish entering text  
=====  
void f(){r= (3+4)*5}  
(({<[ ]>}){r= (3+4)*5} asfjksncjvzic(  
><  
Verified text  
- All good!  
void f(){r= (3+4)*5} - All good!  
- All good!  
(({<[ ]>}){r= (3+4)*5} asfjksncjvzic( - Somethink bad  
Press any key to continue . . .
```

## Висновки

В ході даної лабораторної роботи ми вивчили особливості і обробки текстових файлів та застосували теоретичні відомості на практиці.