Міністерство освіти і науки України КПІ ім. Ігоря Сікорського

Кафедра ІПІ

ЗВІТ

з виконання лабораторної роботи № 1

з кредитного модуля

“Основи програмування-2. Методології програмування” Варіант № 21

Виконав:

студент 1-го курсу гр. ІП-22 ФІОТ

Патріюк Юрій Олексійович

Київ 2023

ПОСТАНОВКА ЗАДАЧІ

21. Створити текстовий файл. Слова у тексті відділені пробілами, комами, крапками; групи слів можуть бути обмежені фігурними дужками. Перевірити, чи є баланс дужок у кожному рядку даного тексту (без вкладеності дужок): для кожної дужки, яка відкривається, справа має бути дужка, що закривається. Рядки вхідного файлу, у яких порушений баланс дужок, переписати у новий текстовий файл. Відкоригувати баланс дужок у новому файлі шляхом видалення зайвих дужок. У кожному фрагменті тексту, обмеженому парою дужок (що залишилися), визначити кількість слів тексту, які містять хоча б одну цифру. Вставити цю величину на початок фрагменту (після відкриваючої дужки), відділивши її символом «-». Вивести вміст вхідного і створеного файлів.

ТЕКСТ ПРОГРАМИ

Вміст файлу main.cpp

#include <iostream>  
#include "functions.h"  
#include "functions\_c.h"  
  
int main(int argc, char \*argv[]) {  
 string name = "main.txt";  
 string secondName = "to\_correct.txt";  
 int mode = -1;  
 if(argc<3)  
 cout<<"Error"<<endl;  
 else  
 {  
 if(strcmp(argv[1],"-mode")==0)  
 {  
 if(strcmp(argv[2], "FilePointer")==0) mode = 0;  
 else if(strcmp(argv[2], "FileStream")==0) mode = 1;  
 else  
 cout<<"Error!"<<endl;  
 }  
 else  
 cout<<"Error!"<<endl;  
 }  
 if(mode == 1) {  
 WriteInFile(name);  
 ShowFile(name);  
 FindAndCopy(name);  
 ShowFile(secondName);  
 ClearFileFromExcessiveBraces(secondName);  
 ShowFile(secondName);  
 FindNumberInBraces(secondName);  
 cout<<"---------Results---------"<<endl;  
 ShowFile(name);  
 ShowFile(secondName);  
 }  
 else if(mode == 0) {  
 WriteInFile\_C(name.c\_str());  
 ShowFile\_C(name.c\_str());  
 FindAndCopy\_C(name.c\_str());  
 ShowFile\_C(secondName.c\_str());  
 ClearFileFromExcessiveBraces\_C(secondName.c\_str());  
 ShowFile\_C(secondName.c\_str());  
 FindNumberInBraces\_C(secondName.c\_str());  
 cout<<"---------Results---------"<<endl;  
 ShowFile\_C(name.c\_str());  
 ShowFile\_C(secondName.c\_str());  
 }  
 cout<<"---------Finish---------"<<endl;  
 system("pause");  
 return 0;  
}

Вміст файлу functions.h

using namespace std;  
  
void ShowFile(const string&);  
void WriteInFile(const string&);  
void FindAndCopy(const string&);  
void ClearFileFromExcessiveBraces(const string&);  
void FindNumberInBraces(const string&);

Вміст файлу functions\_c.h

#include <cstring>  
using namespace std;  
  
void ShowFile\_C(const char\*);  
void WriteInFile\_C(const char\*);  
void FindAndCopy\_C(const char\*);  
void ClearFileFromExcessiveBraces\_C(const char\*);  
void FindNumberInBraces\_C(const char\*);

Вміст файлу functions.cpp

#include <iostream>  
#include "functions.h"  
#include <fstream>  
  
//Обрахунок кількості слів з числами  
int WordsWithNumbers(const string& str)  
{  
 int counter = 0;  
 bool numCheck = false;  
 for(int i = 0; i<str.length(); i++)  
 {  
 if(isdigit(str[i]))  
 numCheck = true;  
 if((str[i] == ' ' || i==str.length()-1) && numCheck)  
 {  
 counter++;  
 numCheck=false;  
 }  
 }  
  
 return counter;  
}  
  
//Вивід вмісту файлу  
void ShowFile(const string& name){  
 ifstream iFile(name);  
 string line;  
 cout<<"Content of "<<name<<endl;  
 while(!iFile.eof())  
 {  
 getline(iFile,line);  
 cout<<'#'<<line<<endl;  
 }  
 cout<<endl;  
 iFile.close();  
}  
//Запис у файл  
void WriteInFile(const std::string& name)  
{  
 ofstream outFile(name);  
 cout<<"Click double enter to finish writing"<<endl;  
 string line=" ";  
 while(!line.empty()) {  
 cout<<"#";  
 getline(cin, line);  
 if(!line.empty())  
 outFile << line << endl;  
 }  
 outFile.close();  
  
}  
//Пошук неправильних рядків та запис у окремий файл  
void FindAndCopy(const string& name)  
{  
 ifstream inFile(name);  
 ofstream additionalFile("to\_correct.txt");  
 string line;  
 while(!inFile.eof())  
 {  
 bool is\_incorrect = false;  
 bool is\_opened = false;  
 getline(inFile,line);  
 if(!line.empty()) {  
 int i = -1;  
 int size = (int) line.length();  
 while (!is\_incorrect && i < size) {  
 ++i;  
 if (line[i] != '{' && line[i] != '}')  
 continue;  
 if (line[i] == '{' && !is\_opened)  
 is\_opened = true;  
 else if (line[i] == '{' && is\_opened || line[i] == '}' && !is\_opened)  
 is\_incorrect = true;  
 else if (line[i] == '}' && is\_opened)  
 is\_opened = false;  
 }  
 if (is\_incorrect || is\_opened)  
 additionalFile << line << endl;  
 }  
 }  
 inFile.close();  
 additionalFile.close();  
}  
//зачистка рядка від зайвих фігурних дужок  
void ClearFileFromExcessiveBraces(const string& name)  
{  
 ifstream iFile(name);  
 string oFileName = "temp.txt";  
 ofstream oFile(oFileName);  
 string line;  
  
 while(!iFile.eof()) {  
 getline(iFile, line);  
 if(!line.empty()) {  
 bool is\_opened = false;  
 int i = 0,  
 open\_position = -1,  
 size = (int) line.length();  
 while (i < size) {  
 if (line[i] == '{' && !is\_opened) {  
 is\_opened = true;  
 open\_position = i;  
 } else if (line[i] == '{' && is\_opened) {  
 line.erase(open\_position, 1);  
 size = (int) line.length();  
 is\_opened = false;  
 i -= 2;  
 } else if (line[i] == '}' && !is\_opened) {  
 line.erase(i, 1);  
 size = (int) line.length();  
 i--;  
 } else if (line[i] == '}' && is\_opened)  
 is\_opened = false;  
 ++i;  
 }  
 if (is\_opened)  
 line.erase(open\_position);  
 oFile << line << endl;  
 }  
  
 }  
 iFile.close();  
 remove(name.c\_str());  
 oFile.close();  
 rename(oFileName.c\_str(),name.c\_str());  
}  
//пошук слів з числами та запис кількості у фігурних дужках  
void FindNumberInBraces(const string& name)  
{  
 ifstream iFile(name);  
 string oFileName = "temp.txt";  
 ofstream oFile(oFileName);  
 string line;  
  
 while(!iFile.eof()) {  
 getline(iFile, line);  
 if(!line.empty()) {  
 int open\_position = 0, close\_position = 0;  
 while (open\_position != string::npos && close\_position != string::npos) {  
 open\_position = (int) line.find('{', open\_position);  
 close\_position = (int) line.find('}', close\_position);  
  
 if (open\_position != string::npos && close\_position != string::npos) {  
 string temp = line.substr(open\_position + 1, close\_position - open\_position - 1);  
 int amount = WordsWithNumbers(temp);  
 if (amount != 0) {  
 line.insert(open\_position + 1, '-' + to\_string(amount) + '-');  
 close\_position += 4;  
 } else  
 close\_position++;  
 open\_position = close\_position;  
 }  
 }  
 oFile << line << endl;  
 }  
 }  
 iFile.close();  
 remove(name.c\_str());  
 oFile.close();  
 rename(oFileName.c\_str(),name.c\_str());  
}

Вміст файлу functions\_c.cpp

#include <iostream>  
#include "functions\_c.h"  
#define MAX\_LINE\_LENGTH 500  
  
//видалення елементу за індексом  
void RemoveElementByIndex(char\* array,int index)  
{  
 int i = index;  
 while(array[i]!=0)  
 {  
 array[i] = array[i+1];  
 i++;  
 }  
}  
//пошук позиції вказаного символу  
int FindPosition(const char\* array, int start\_position,char elem)  
{  
 if(strlen(array)<start\_position)  
 return -1;  
 else {  
 int i = start\_position;  
 while (array[i] != elem && array[i] != '\n')  
 i++;  
 if (array[i] == elem)  
 return i;  
 else  
 return -1;  
 }  
}  
//копіювання масиву з позиції довжиною length  
void Copy(const char\* array, char\* buff, int start\_position, int length)  
{  
 int i = 0;  
 while(i<length)  
 {  
 buff[i] = array[start\_position+i];  
 i++;  
 }  
  
}  
//Обрахунок кількості слів з числами  
int WordsWithNumbers(const char\* str)  
{  
 int counter = 0;  
 bool numCheck = false;  
 for(int i = 0; i<strlen(str); i++)  
 {  
 if(isdigit(str[i]))  
 numCheck = true;  
 if((str[i] == ' ' || i == strlen(str)-1) && numCheck)  
 {  
 counter++;  
 numCheck=false;  
 }  
 }  
 return counter;  
}  
//вставлення на позицію кількості слів з числами  
void InsertAmount(char\* array, int position, int amount)  
{  
 int i = -1;  
 char result[MAX\_LINE\_LENGTH] = "";  
 char temp[MAX\_LINE\_LENGTH]="";  
 Copy(array, temp,position, (int)strlen(array)-position);  
 while (i!=position)  
 {  
 i++;  
 if(i==position)  
 {  
 strcat(result,("-"+to\_string(amount)+"-").c\_str());  
 }  
 else {  
 result[i] = array[i];  
 }  
 }  
  
 strcat(result,temp);  
 strcpy(array,result);  
  
  
}  
  
//Вивід вмісту файлу  
void ShowFile\_C(const char\* name)  
{  
 FILE\* file;  
 file = fopen(name,"r");  
 char line[MAX\_LINE\_LENGTH];  
 cout<<"Content of "<<name<<endl;  
 while(fgets(line,MAX\_LINE\_LENGTH,file))  
 {  
 cout<<'#'<<line;  
 }  
 cout<<endl;  
 fclose(file);  
}  
//Запис у файл  
void WriteInFile\_C(const char\* name)  
{  
 FILE\* file;  
 file = fopen(name,"w");  
 char line[MAX\_LINE\_LENGTH] = " ";  
 cout<<"Click double enter to finish writing"<<endl;  
 while(strcmp(line,"")!=0) {  
 cout<<"#";  
 gets(line);  
 if(strcmp(line,"")!=0) {  
 fputs(line, file);  
 fputs("\n", file);  
 }  
 }  
 fclose(file);  
}  
//Пошук неправильних рядків та запис у окремий файл  
void FindAndCopy\_C(const char\* name)  
{  
 FILE\* main, \*to\_correct;  
 main = fopen(name,"r");  
 to\_correct = fopen("to\_correct.txt","w");  
 char line[MAX\_LINE\_LENGTH] = " ";  
 while(fgets(line,MAX\_LINE\_LENGTH,main))  
 {  
 bool is\_incorrect = false;  
 bool is\_opened = false;  
  
 int i = -1;  
 while (!is\_incorrect && line[i]!='\n') {  
 ++i;  
 if (line[i] != '{' && line[i] != '}')  
 continue;  
 if (line[i] == '{' && !is\_opened)  
 is\_opened = true;  
 else if (line[i] == '{' && is\_opened || line[i] == '}' && !is\_opened)  
 is\_incorrect = true;  
 else if (line[i] == '}' && is\_opened)  
 is\_opened = false;  
 }  
 if (is\_incorrect || is\_opened)  
 fputs(line,to\_correct);  
  
 }  
 fclose(main);  
 fclose(to\_correct);  
}  
//зачистка рядка від зайвих фігурних дужок  
void ClearFileFromExcessiveBraces\_C(const char\* name)  
{  
 FILE\* oFile, \*tFile;  
 oFile = fopen(name,"r");  
 char oFileName[] = "temp.txt";  
 tFile = fopen(oFileName,"w");  
 char line[MAX\_LINE\_LENGTH] = " ";  
  
 while(fgets(line,MAX\_LINE\_LENGTH,oFile)) {  
 if(strcmp(line,"")!=0) {  
 bool is\_opened = false;  
 int i = 0,  
 open\_position = -1;  
 while (line[i] != '\n') {  
 if (line[i] == '{' && !is\_opened) {  
 is\_opened = true;  
 open\_position = i;  
 } else if (line[i] == '{' && is\_opened) {  
 RemoveElementByIndex(line,open\_position);  
 is\_opened = false;  
 i -= 2;  
 } else if (line[i] == '}' && !is\_opened) {  
 RemoveElementByIndex(line,i-1);  
 i--;  
 } else if (line[i] == '}' && is\_opened)  
 is\_opened = false;  
 ++i;  
 }  
 if (is\_opened)  
 RemoveElementByIndex(line,open\_position);  
 fputs(line, tFile);  
 memset(line,0,sizeof(line));  
 }  
  
 }  
 fclose(oFile);  
 remove(name);  
 fclose(tFile);  
 rename(oFileName,name);  
}  
//пошук слів з числами та запис кількості у фігурних дужках  
void FindNumberInBraces\_C(const char\* name)  
{  
 FILE\* oFile, \*tFile;  
 oFile = fopen(name,"r");  
 char oFileName[] = "temp.txt";  
 tFile = fopen(oFileName,"w");  
 char line[MAX\_LINE\_LENGTH] = " ";  
  
 while(fgets(line,MAX\_LINE\_LENGTH,oFile)) {  
 if(strcmp(line,"")!=0) {  
 int open\_position = 0, close\_position = 0;  
 while (open\_position != -1 && close\_position != -1) {  
 open\_position = FindPosition(line,open\_position,'{');  
 close\_position = FindPosition(line,close\_position,'}');  
  
 if (open\_position != -1 && close\_position != -1) {  
 char temp[MAX\_LINE\_LENGTH]="";  
 Copy(line,temp,open\_position+1,close\_position-open\_position-1);  
 int amount = WordsWithNumbers(temp);  
 if (amount != 0) {  
 InsertAmount(line,open\_position+1,amount);  
 close\_position += 4;  
 } else  
 close\_position++;  
 open\_position = close\_position;  
  
 }  
 }  
 fputs(line,tFile);  
 }  
 }  
 fclose(oFile);  
 remove(name);  
 fclose(tFile);  
 rename(oFileName,name);  
}

РЕЗУЛЬТАТ ТЕСТУВАННЯ

