ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

**Московский институт электроники и математики им. А.Н. Тихонова**

Саркисянц Юрий Григорьевич, группа СКБ221

Лабораторная работа №3
по дисциплине «Языки программирования»

студента образовательной программы
«Компьютерная безопасность»

Студент _____ Саркисянц Юрий
подпись

Москва 2022 г.

# Оглавление

# Ошибки

**Член else**

You entered wrong count of flags

**Член if (argc==1)**

The flag no found
You don't entered name_file
You entered wrong flag
Need to entered name for both files
You entered wrong flag
Entered Error

# Список файлов

## Файлы

Полный список файлов.

# Файлы

## Файл func.cpp

Файл с описанием функций для сортировке предложений в введенном тексте в порядке неубывания

```
#include <iostream>
#include <math.h>
#include <cstring>
#include <fstream>
```

### Функции

- void **FromUsage** ()
  *Функция, используемая при некорректном вводе флага –fromfile, выводит корректную последовательность флагов*

- void **ToUsage** ()
  *Функция, используемая при некорректном вводе флага –tofile, выводит корректную последовательность флагов*

- void **FromToUsage** ()
  *Функция, используемая при некорректном вводе флагов –fromfile и –tofile, выводит корректную последовательность флагов*

- void **CommonUsage** ()
  *Функция для ввода общего меню при некорректном вводе флагов*

- void **ToFile** (char *namefile)
  *Функция для вывода строки из консоли и ввода корректного ответа в указанный пользователем текстовый файл*

- void **FromFile** (char *namefile)
  *Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в консоль*

- void **FromAndToFile** (char *in_file, char *to_file)
  *Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в указанный пользователем текстовый файл*

- void **Nine** ()
  *Функция для ввода максимальной строки в обратном порядке слов в файл "nine.txt".*

---

### Подробное описание

Файл с описанием функций для сортировке предложений в введенном тексте в порядке неубывания

См. определение в файле **func.cpp**

---

## Функции

### void CommonUsage ()

Функция для ввода общего меню при некорректном вводе флагов

См. определение в файле **func.cpp** строка **33**

### void FromAndToFile (char * *in_file*, char * *to_file*)

Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в указанный пользователем текстовый файл

**Аргументы**

| | |
|---|---|
| *in_file* | название файла, которое получает функция для вывода из него данных |
| *to_file* | название файла, которое получает функция для ввода в него корректного ответа |

См. определение в файле **func.cpp** строка **175**

### void FromFile (char * *namefile*)

Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в консоль

**Аргументы**

| | |
|---|---|
| *namefile* | название файла, которое получает функция |

См. определение в файле **func.cpp** строка **109**

### void FromToUsage ()

Функция, используемая при некорректном вводе флагов –fromfile и –tofile, выводит корректную последовательность флагов

См. определение в файле **func.cpp** строка **27**

### void FromUsage ()

Функция, используемая при некорректном вводе флага –fromfile, выводит корректную последовательность флагов

См. определение в файле **func.cpp** строка **15**

### void Nine ()

Функция для ввода максимальной строки в обратном порядке слов в файл "nine.txt".

См. определение в файле **func.cpp** строка **238**

### void ToFile (char * *namefile*)

Функция для вывода строки из консоли и ввода корректного ответа в указанный пользователем текстовый файл

#### Аргументы

| | |
|---|---|
| *namefile* | название файла, которое получает функция |

См. определение в файле **func.cpp** строка **43**

### void ToUsage ()

Функция, используемая при некорректном вводе флага –tofile, выводит корректную последовательность флагов

См. определение в файле **func.cpp** строка **21**

## func.cpp

См. документацию.

```
00001
00005 #include <iostream>
00006 #include <math.h>
00007 #include <cstring>
00008 #include <fstream>
00009 using std::cout;
00010 using std::endl;
00011 using std::cin;
00015 void FromUsage(){
00016     cout << "For reading from file enter: [./nameprog --fromfile <namefile.txt>]"
<< endl;
00017 }
00021 void ToUsage(){
00022     cout << "For out to file enter: [./<nameprog> --tofile <namefile.txt>]" << endl;
00023 }
00027 void FromToUsage(){
00028     cout << "For reading from one file and out to another file enter: [./<nameprog>
--fromfile --tofile <file1> <file2>]" << endl;
00029 }
00033 void CommonUsage(){
00034     cout << "Write options in correct form:" << endl;
00035     FromUsage();
00036     ToUsage();
00037     FromToUsage();
00038 }
00043 void ToFile(char *namefile){
00044     std::ofstream out temp file ("temp file.txt"); // Создаем промежуточный файл
00045     std::ofstream result_file(namefile); // Создаем файл для записи результата
00046     std::ofstream maxSent ("maxSent.txt");
00047     char cur char = '0';
00048     char prev char = 'a';
00049     int curLen = 0;
00050     int maxLen = 0;
00051     int numSentences = 0;
00052     while (std::cin.get(cur char) && cur char != '\n') {
00053         if (cur char == '.') {
00054             numSentences += 1;
00055             maxLen = std::max(++curLen, maxLen);
00056             curLen = 0;
00057             prev_char = '.';
00058             out temp file << ".\n";
00059         }
00060         else if (cur char == ' ' && prev char == '.')
00061             continue;
00062
00063         else {
00064             out temp file << cur char;
00065             prev char = cur char;
00066             curLen++;
00067         }
00068     }
00069     out_temp_file.close();
00070     maxLen++;
00071     char sentences[numSentences][maxLen];  //Объявляем два массива, в одном будут
хранится предложения и их длины
00072     int lengths[numSentences]; //Во втором длины, но с связью с первым массивом
00073     std::ifstream in temp file ("temp file.txt"); //Из файла посимвольно записываем
предложения в массив, а также их длины
00074     for (int i = 0; i < numSentences; i++) { //Проходимся двойным циклом, если
предложение закончилось, то мы
00075         for (int k = 0; k < maxLen; k++) { // В массив длин записываем длину предл.
00076             in_temp_file.get(cur_char);
00077             if (cur char == '\n') {
00078                 sentences[i][k] = '\0';
00079                 lengths[i] = k;
00080                 break;
00081             }
00082             sentences[i][k] = cur_char;   // Продолжаем считывать в массив символы
с файла
00083         }
00084     }
00085     in temp file.close();
00086     int temp = 0;
```

```
00087     char temps[maxLen];
00088     for (int i = 0; i < numSentences;i++){
00089         for (int j = 0; j < numSentences - i - 1;j++){
00090             if (lengths[j]> lengths[j+1]){
00091                     temp = lengths[j];
00092                     lengths[j] = lengths[j+1];
00093                     lengths[j+1] = temp;
00094                     strcpy(temps,sentences[j]);
00095                     strcpy(sentences[j],sentences[j+1]);
00096                     strcpy(sentences[j+1],temps);
00097             }
00098         }
00099     }
00100     for (int i = 0; i < numSentences; i++)
00101         result file << sentences[i] << "   " << lengths[i] << endl;
00102     maxSent << sentences[numSentences-1] << endl;
00103     result_file.close();
00104 }
00109 void FromFile(char *namefile) {
00110     std::ofstream out temp file ("temp file.txt");
00111     std::ifstream input file (namefile);
00112     std::ofstream maxSent ("maxSent.txt");
00113     char cur_char = '0';
00114     char prev char = 'a';
00115     int curLen = 0;
00116     int maxLen = 0;
00117     int numSentences = 0;
00118     while (input_file.get(cur_char) && cur_char != '\n') {
00119         if (cur_char == '.') {
00120             numSentences += 1;
00121             maxLen = std::max(++curLen, maxLen);
00122             curLen = 0;
00123             prev char = '.';
00124             out_temp_file << ".\n";
00125         }
00126         else if (cur char == ' ' && prev char == '.')
00127             continue;
00128         else {
00129             out_temp_file << cur_char;
00130             prev_char = cur_char;
00131             curLen++;
00132         }
00133     }
00134     out temp file.close();
00135     input_file.close();
00136     maxLen++;
00137     char sentences[numSentences][maxLen];
00138     int lengths[numSentences];
00139     std::ifstream in temp file ("temp file.txt");
00140     for (int i = 0; i < numSentences; i++) {
00141         for (int k = 0; k < maxLen; k++) {
00142             in_temp_file.get(cur_char);
00143             if (cur char == '\n') {
00144                 sentences[i][k] = '\0';
00145                 lengths[i] = k;
00146                 break;
00147             }
00148             sentences[i][k] = cur char;
00149         }
00150     }
00151     in temp file.close();
00152     int temp = 0;
00153     char temps[maxLen];
00154     for (int i = 0; i < numSentences;i++){
00155         for (int j = 0; j < numSentences - i - 1;j++){
00156             if (lengths[j]> lengths[j+1]){
00157                     temp = lengths[j];
00158                     lengths[j] = lengths[j+1];
00159                     lengths[j+1] = temp;
00160                     strcpy(temps,sentences[j]);
00161                     strcpy(sentences[j],sentences[j+1]);
00162                     strcpy(sentences[j+1],temps);
00163             }
00164         }
00165     }
00166     for (int i = 0; i < numSentences; ++i)
00167         cout << sentences[i] << "   " << lengths[i] << endl;
```

```
00168     maxSent << sentences[numSentences-1] <<  endl;
00169 }
00175 void FromAndToFile(char *in file, char *to file){
00176     std::ofstream out temp file("temp file.txt");
00177     std::ifstream input_file (in_file);
00178     std::ofstream result_file(to_file);
00179     char cur char = '0';
00180     char prev char = 'a';
00181     int curLen = 0;
00182     int maxLen = 0;
00183     int numSentences = 0;
00184     while (input_file.get(cur_char) && cur_char != '\n') {
00185         if (cur char == '.') {
00186             numSentences += 1;
00187             maxLen = std::max(++curLen, maxLen);
00188             curLen = 0;
00189             prev_char = '.';
00190             out_temp_file << ".\n";
00191         }
00192         else if (cur char == ' ' && prev char == '.')
00193             continue;
00194         else {
00195             out_temp_file << cur_char;
00196             prev char = cur char;
00197             curLen++;
00198         }
00199     }
00200     out_temp_file.close();
00201     input_file.close();
00202     maxLen++;
00203     char sentences[numSentences][maxLen];
00204     int lengths[numSentences];
00205     std::ifstream in temp file ("temp file.txt");
00206     for (int i = 0; i < numSentences; i++) {
00207         for (int k = 0; k < maxLen; k++) {
00208             in temp file.get(cur char);
00209             if (cur char == '\n') {
00210                 sentences[i][k] = '\0';
00211                 lengths[i] = k;
00212                 break;
00213             }
00214             sentences[i][k] = cur char;
00215         }
00216     }
00217     in_temp_file.close();
00218     int temp = 0;
00219     char temps[maxLen];
00220     for (int i = 0; i < numSentences;i++){
00221         for (int j = 0; j < numSentences - i - 1;j++){
00222             if (lengths[j]> lengths[j+1]){
00223                     temp = lengths[j];
00224                     lengths[j] = lengths[j+1];
00225                     lengths[j+1] = temp;
00226                     strcpy(temps,sentences[j]);
00227                     strcpy(sentences[j],sentences[j+1]);
00228                     strcpy(sentences[j+1],temps);
00229             }
00230         }
00231     }
00232     for (int i = 0; i < numSentences; i++)
00233         result file << sentences[i] << "   " << lengths[i] << endl;
00234 }
00238 void Nine(){
00239     std::ifstream input_file("maxSent.txt");
00240     std::ofstream temp("tempfile.txt");
00241     char cur char = '0';
00242     char prev char = 'a';
00243     int curLen = 0;
00244     int maxLen = 0;
00245     int numWords = 1;
00246     while (input file.get(cur char)){
00247         if (cur char == ' '){
00248             numWords+=1;
00249             maxLen = std::max(++curLen, maxLen);
00250             curLen = 0;
00251             temp << "\n";
00252         }
```

```
00253         else {
00254             temp << cur char;
00255             curLen++;
00256     }
00257     }
00258     input_file.close();
00259     temp.close();
00260     maxLen++;
00261     char words[numWords][maxLen];
00262     std::ifstream tempor ("tempfile.txt");
00263     std::ofstream res_file("nine.txt");
00264     for (int i = 0; i< numWords;i++){
00265         for (int j = 0; j< maxLen;j++){
00266             tempor.get(cur char);
00267             if (cur char == '\n') {
00268                     words[i][j] = '\0';
00269                     break;
00270                 }
00271             if (cur char!='.'){
00272 
00273                 words[i][j] = cur char;
00274                 }
00275     }
00276     }
00277     char token;
00278     for (int i = numWords-1;i>=0;i--){
00279         int j = 0;
00280         while (token = words[i][j++]){
00281             res_file << token;
00282         }
00283         res file << " ";
00284     }
00285     tempor.close();
00286 }
```

# Файл func.h

## Функции

- void **CommonUsage** ()
  *Функция для ввода общего меню при некорректном вводе флагов*

- void **FromUsage** ()
  *Функция, используемая при некорректном вводе флага –fromfile, выводит корректную последовательность флагов*

- void **ToUsage** ()
  *Функция, используемая при некорректном вводе флага –tofile, выводит корректную последовательность флагов*

- void **FromToUsage** ()
  *Функция, используемая при некорректном вводе флагов –fromfile и –tofile, выводит корректную последовательность флагов*

- void **FromFile** (char *namefile)
  *Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в консоль*

- void **FromAndToFile** (char *in_file, char *to_file)
  *Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в указанный пользователем текстовый файл*

- void **ToFile** (char *namefile)
  *Функция для вывода строки из консоли и ввода корректного ответа в указанный пользователем текстовый файл*

- void **Nine** ()
  *Функция для ввода максимальной строки в обратном порядке слов в файл "nine.txt".*

---

## Функции

### void CommonUsage ()

Функция для ввода общего меню при некорректном вводе флагов

См. определение в файле **func.cpp** строка **33**

### void FromAndToFile (char * *in_file*, char * *to_file*)

Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в указанный пользователем текстовый файл

**Аргументы**

| | |
|---|---|
| *in_file* | название файла, которое получает функция для вывода из него данных |
| *to_file* | название файла, которое получает функция для ввода в него корректного ответа |

См. определение в файле **func.cpp** строка **175**

### void FromFile (char * *namefile*)

Функция для вывода строки из файла, указанного пользователем и ввода корректного ответа в консоль

**Аргументы**

| | |
|---|---|
| *namefile* | название файла, которое получает функция |

См. определение в файле **func.cpp** строка **109**

### void FromToUsage ()

Функция, используемая при некорректном вводе флагов –fromfile и –tofile, выводит корректную последовательность флагов

См. определение в файле **func.cpp** строка **27**

### void FromUsage ()

Функция, используемая при некорректном вводе флага –fromfile, выводит корректную последовательность флагов

См. определение в файле **func.cpp** строка **15**

### void Nine ()

Функция для ввода максимальной строки в обратном порядке слов в файл "nine.txt".

См. определение в файле **func.cpp** строка **238**

### void ToFile (char * *namefile*)

Функция для вывода строки из консоли и ввода корректного ответа в указанный пользователем текстовый файл

**Аргументы**

| | |
|---|---|
| *namefile* | название файла, которое получает функция |

См. определение в файле **func.cpp** строка **43**

### void ToUsage ()

Функция, используемая при некорректном вводе флага –tofile, выводит корректную последовательность флагов

См. определение в файле **func.cpp** строка **21**

## func.h

См. документацию.
```
00001 #ifndef FUNCTIONS
00002 #define FUNCTIONS
00003 void CommonUsage();
00004 void FromUsage();
00005 void ToUsage();
00006 void FromToUsage();
00007 void FromFile(char* namefile);
00008 void FromAndToFile(char* in file, char* to file);
00009 void ToFile(char* namefile);
00010 void Nine();
00011 #endif
```

# Файл main.cpp

Главный файл с реализацией вводов флагов и названий файлов пользователем

```
#include <iostream>
#include "func.h"
#include <cstring>
```

## Функции

- * **if** (argc==1)
- **CommonUsage** ()
  *Функция для ввода общего меню при некорректном вводе флагов*

## Переменные

- **else**

---

## Подробное описание

Главный файл с реализацией вводов флагов и названий файлов пользователем

См. определение в файле **main.cpp**

---

## Функции

### CommonUsage ()

Функция для ввода общего меню при некорректном вводе флагов

См. определение в файле **func.cpp** строка **33**

### else if (argc == 1)

Массив типа const содержащий флаги, которые должны вводиться пользователем

```
 const char* flags[2] = {"--fromfile","--tofile"};

const char* flags[2] = {"--fromfile","--tofile"};
int main(int argc, char*argv[]){
```

Если был введен 2 флага, то нужно проверить его на соответствие с объявленными в начале:

**Ошибка:**

The flag no found

```
if (argc == 2){}
```

Если же флагов было 3, то

```
else if (argc == 3){}
```

Если же ,было введено 5 флагов

```
else if (argc == 5){}
```

Если введенный флаг совпадает с –tofile или –namefile, то нужно потребовать введения файла с помощью функции ToUsage или FromUsage соответственно. При этом вводится ошибка:

**Ошибка:**

You don't entered name_file

```
 if (!strcmp(argv[1],flags[1])){
    cerr << "You don't entered name file" << endl;
    ToUsage();
}
else if (!strcmp(argv[1],flags[0])){
    cerr << "You don't entered name file" << endl;
    FromUsage();
}
```

Если флаг не совпадает с верхними, то вызывается меню с помощью ф-ии CommonUsage и выводится ошибка:

**Ошибка:**

You entered wrong flag

```
 else {
    cerr << "You entered wrong flag" << endl;
    CommonUsage();
}
```

Объявляем и инициализируем массив типа char, куда будет введен третий флаг

```
char *namefile = argv[2];
```

Если второй аргумент это –tofile, то мы учитываем третий аргумент в качестве названия файла и выполним функцию nine для доп.задания

```
 if (!strcmp(argv[1],flags[1]))  {
    ToFile(namefile);
    Nine();
}
```

Если же пользователь ввел флаги –tofile и –fromfile подряд, то выйдет ошибка и меню с правильным вводом

**Ошибка:**

Need to entered name for both files

```
else if ((((!strcmp(argv[1],flags[0])) && (!strcmp(argv[2],flags[1]))) ||
      ((!strcmp(argv[1],flags[1])) && (!strcmp(argv[2],flags[0])))){
      cerr << "Need to entered name for both files" << endl;;
      FromToUsage();
      }
```

Если же пользователь ввел первым флаг –fromfile, то мы учитываем второй аргумент в качестве файла

```
else if (!strcmp(argv[1],flags[0]))     FromFile(namefile);
```

Если же были введены ошибочные флаги, то будет выведена ошибка

**Ошибка:**

You entered wrong flag

```
else cerr << "You entered wrong flag" << endl;
```

Объявляем и инициализируем 2 массива типа char, в одном будет содержаться название входного файла, во втором выходного

```
 char *in file = argv[2];
char *to_file = argv[4];
```

Если же было введено 5 флагов, при этом первые и третий согласно требованию меню это флаги –fromfile и –tofile соотвественно, то выполняем **FromAndToFile()**

```
 if ((!strcmp(argv[1],flags[0])) && (!strcmp(argv[3], flags[1]))) {
    FromAndToFile(in file,to file);
}
```

А иначе вывести ошибку и вывести меню:

**Ошибка:**

Entered Error

```
 else {
cerr << "Entered Error" << endl;
FromToUsage();
}
```

См. определение в файле **main.cpp** строка **29**

---

## Переменные

### else

```
Инициализатор{
        cerr << "You entered wrong count of flags" << endl
```

Если было введено некорректное число флагов то выводится ошибка и меню:

**Ошибка:**

You entered wrong count of flags

```
 else {
    cerr << "You entered wrong count of flags" << endl;
    CommonUsage();
}
```

См. определение в файле **main.cpp** строка **161**

## main.cpp

```
См. документацию.00001
00005 #include <iostream>
00006 #include "func.h"
00007 #include <cstring>
00008
00009 using std::cout;
00010 using std::cin;
00011 using std::cerr;
00012 using std::endl;
00015
00016 const char* flags[2] = {"--fromfile","--tofile"};
00021 int main(int argc, char*argv[]){
00029     if (argc == 1){
00030         cerr << "The flag no found" << endl;
00031     }
00037     else if (argc == 2){
00050         if (!strcmp(argv[1],flags[1])){
00051             cerr << "You don't entered name_file" << endl;
00052             ToUsage();
00053         }
00054
00055         else if (!strcmp(argv[1],flags[0])){
00056             cerr << "You don't entered name_file" << endl;
00057             FromUsage();
00058         }
00067         else {
00068             cerr << "You entered wrong flag" << endl;
00069             CommonUsage();
00070         }
00071     }
00076     else if (argc == 3){
00081         char *namefile = argv[2];
00089         if (!strcmp(argv[1],flags[1]))  {
00090             ToFile(namefile);
00091             Nine();
00092         }
00102         else if (((!strcmp(argv[1],flags[0])) && (!strcmp(argv[2],flags[1]))) ||
00103                 ((!strcmp(argv[1],flags[1])) && (!strcmp(argv[2],flags[0])))){
00104                 cerr << "Need to entered name for both files" << endl;;
00105                 FromToUsage();
00106                 }
00111         else if (!strcmp(argv[1],flags[0]))     FromFile(namefile);
00117         else cerr << "You entered wrong flag" << endl;
00118     }
00123     else if (argc == 5){
00129         char *in_file = argv[2];
00130         char *to file = argv[4];
00137         if ((!strcmp(argv[1],flags[0])) && (!strcmp(argv[3], flags[1]))) {
00138             FromAndToFile(in file,to file);
00139         }
00148         else {
00149         cerr << "Entered Error" << endl;
00150         FromToUsage();
00151         }
00152     }.
00161     else {
00162         cerr << "You entered wrong count of flags" << endl;
00163         CommonUsage();
00164
00165
00166
00167 }
00168 }
```

# Алфавитный указатель

INDEX