**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

**КУРСОВАЯ РАБОТА**

**по дисциплине «Программирование»**

Тема: Обработка текстовых файлов

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Иванов Д.В. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент Иванов Д.В. | | |
| Группа 6303 | | |
| Тема работы: | | |
|  | | |
| Содержание пояснительной записки:   * Содержание * Введение * Описание функций, необходимых для работы программы * Примеры работы программы * Заключение * Список использованных источников * Приложение А. Исходный код программы | | |
|  | | |
| Дата выдачи задания: 28.05.2017 | | |
| Дата сдачи: | | |
| Дата защиты: | | |
| Студент |  | Иванов Д.В. |
| Преподаватель |  | Берленко Т.А. |

**Аннотация**

В данной работе была создана программа на языке программирования C, которая производит обработку csv-файлов посредством набора функций. Были созданы и описаны необходимые функции, позволяющие проверять корректность строк, содержащихся в файлах, проводить поиск определенных строк в файлах, сохранять данные из файлов. Помимо этого, была проведена работа над оптимизацией исходного кода программы для ускорения ее быстродействия и оптимального использования памяти и ресурсов клиента. Приведены примеры работы программы, а также полное описание исходного кода.

Оглавление

[Введение 1](#_Toc482281022)

[Функции для работы программы 2](#_Toc482281023)

[1.1. Функция для проверки корректности файлов. 2](#_Toc482281024)

[1.2. Функция для проверки корректности строки. 3](#_Toc482281025)

[1.3. Функция для определения параметра ошибки. 3](#_Toc482281026)

[1.4. Функция записи студента. 4](#_Toc482281027)

[1.5. Функция поиска студента в файле. 5](#_Toc482281028)

[1.6. Функция для закрытия файлов и освобождения памяти. 5](#_Toc482281029)

[Примеры работы программы 6](#_Toc482281030)

[Заключение 8](#_Toc482281031)

[Список использованных источников 9](#_Toc482281032)

[Приложение А 10](#_Toc482281033)

# Введение

Требуется написать программу, на вход которой подаются две таблицы, хранящиеся в csv файлах. Первая таблица содержит информацию о ФИО студента и его Github аккаунте, email'е и номере группы. Вторая таблица содержит фамилию, имя, количество баллов за экзамен. Программа должна находить количество студентов, которые получили максимальный балл и выводить результат на консоль.

Программа получает параметры из входного потока. Параметры:

* input\_file\_1 - csv файл
* input\_file\_2 - csv файл

В случае, если программа получила некорректные параметры, то:

* не создается выходного файла
* выводится сообщение об ошибке “Fail with <имя параметра>”.

# Функции для работы программы

## Функция для проверки корректности файлов.

1. int ValidOfFile(FILE \*file1, FILE \*file2, Students \*StudFromSec)
2. {
3. int i1 = 0, i2 = 0;
4. char \*string=(char \*)malloc(200);
5. if ((file1 == NULL)||(file2 == NULL))
6. {
7. if (file1 == NULL)
8. printf("input\_file\_1.csv doesn't exist\n");
9. if (file2 == NULL)
10. printf("input\_file\_2.csv doesn't exist\n");
11. return -3;
12. }
13. while (!feof(file1))
14. {
15. fscanf(file1, "%s", string);
16. if (string[0] != '\0')
17. {
18. if (ParametrOfError(ValidityOfString(string, 6), 6) == -1)
19. {
20. return -2;
21. }
22. i1++;
23. string[0] = '\0';
24. }
25. }
26. while (!feof(file2))
27. {
28. fscanf(file2, "%s", string);
29. if (string[0] != '\0')
30. {
31. if (ParametrOfError(ValidityOfString(string, 3), 3) == -1)
32. {
33. return -2;
34. }
35. if (SaveStud(string, StudFromSec, i2)==-1)
36. return -4;
37. i2++;
38. string[0] = '\0';
39. }
40. }
41. fclose(file1);
42. fclose(file2);
43. free(string);
44. if (i1 == i2)
45. return i1;
46. return -1;
47. }

Данной функции передают четыре параметра: два указателя на входные файлы, указатель на массив студентов, указатель на кол-во студентов в массиве. Функция возвращает:

* Количество студентов в файле, если оба файлы корректны
* -1, если количество студентов в файлах различается
* -2, если строка в файле не корректна
* -3, если один из файлов не существует
* -4, если студент во 2ом файле встречается дважды

## Функция для проверки корректности строки.

1. int ValidityOfString(char \*string, int ID)
2. {
3. char tmp[200];
4. unsigned int i = 0, count = 1;
5. while ((string[i] != '\n') && (i<strlen(string)))
6. {
7. int j = 0;
8. while ((string[i] != ';') && (string[i] != ',') && (i<strlen(string)))
9. {
10. tmp[j] = string[i];
11. i++;
12. j++;
13. }
14. tmp[j] = '\0';
15. if (tmp[0] == '\0')
16. return count;
17. if ((ID == 3) && (count == 3))
18. {
19. int score = atoi(tmp);
20. if ((score == 0) && (strcmp(tmp,"0")!=0))
21. return 3;
22. }
23. count++;
24. i++;
25. }
26. if ((count - 1) != ID)
27. return count;
28. else
29. return 0;
30. }

Функция получает строку, а также ID файла, из которого она была считана (Первый файл ID=6, второй - ID=3). Возвращает

* 0, если строка корректна
* Код параметра ошибки, если строка не корректна

## Функция для определения параметра ошибки.

1. int ParametrOfError(int ErrorCode, int ID)
2. {
3. switch (ErrorCode)
4. {
5. case 0: return 0;
6. case 1:
7. {
8. if (ID == 6)
9. printf("Failed with <Surname. File №1>\n");
10. else printf("Failed with <Surname. File №2>\n");
11. break;
12. }
13. case 2:
14. {
15. if (ID == 6)
16. printf("Failed with <Name. File №1>\n");
17. else printf("Failed with <Name. File №2>\n");
18. break;
19. }
20. case 3:
21. {
22. if (ID == 6)
23. return 0;
24. else printf("Failed with <Score>\n");
25. break;
26. }
27. case 4: printf("Failed with <Git>\n"); break;
28. case 5: printf("Failed with <Mail>\n"); break;
29. case 6: printf("Failed with <Group>\n"); break;
30. }
31. return -1;
32. }

Данной функции передается код ошибки, а также ID файла, в строке которого была найдена ошибка. Функция печатает сообщение, если ошибка есть (ErrorKod!=0) и возвращает -1, если ошибки нет, возвращает 0.

## Функция записи студента.

1. int SaveStud (char \*string, Students \*StudFromSec, int num)
2. {
3. char \*Surname = strtok(string, ";,");
4. char \*Name = strtok(NULL, ";,");
5. char \*Score = strtok(NULL, ";,");
6. if ((strcmp(Surname, StudFromSec[i].Surname) == 0) && (strcmp(Name, StudFromSec[i].Name) == 0))
7. return -1;
8. strcpy(StudFromSec[num].Surname, Surname);
9. strcpy(StudFromSec[num].Name, Name);
10. StudFromSec[num].Score =atoi(Score);
11. return 0;
12. }

Функция получает указатели на строку, на массив студентов и номер студента. Перемещаясь по массиву, она проверяет нет встречался ли поданный студент ранее. Если такой студент уже был считан возвращает -1, если нет – записывает фамилию, имя и балл студента в массив.

## Функция поиска студента в файле.

1. int Search(FILE \*In1, Students \*StudFromSec)
2. {
3. char StringFromFirst[200];
4. int count = 0, score=0;
5. while (!feof(In1))
6. {
7. fscanf(In1, "%s", StringFromFirst);
8. if (StringFromFirst[0] != '\0')
9. {
10. char \*Surname1 = strtok(StringFromFirst, ";,\n");
11. if (strcmp(Surname1, StudFromSec.Surname) == 0)
12. {
13. char \*Name1 = strtok(NULL, ";,\n");
14. if (strcmp(Name1, StudFromSec.Name) == 0)
15. {
16. count++;
17. }
18. }
19. }
20. StringFromFirst[0] = '\0';
21. }
22. if (count > 1)
23. return -2;
24. else
25. if (count == 1)
26. {
27. return StudFromSec.Score;
28. }
29. else return -1;
30. }

Функция получает указатель на первый файл и элемент структуры, содержащий данные о студенте. Ищет данного студента в первом файле. Возвращает

* Балл студента, если студент есть и единственен
* -1, если студент не найден
* -2, если в файле несколько студентов с такими данными

## Функция для закрытия файлов и освобождения памяти.

1. void BeforeTheEnd(FILE \*file1, FILE \*file2, Students \*StudFromSec)
2. {
3. fclose(file1);
4. fclose(file2);
5. free(StudFromSec);
6. }

Данной функции передаются указатели на файлы и на динамический массив. Функция закрывает файлы и освобождает память. Ничего не возвращает.

# Примеры работы программы

* Пример работы программы при корректных входных данных

|  |  |  |
| --- | --- | --- |
| input\_file\_1 | input\_file\_2 | Результат |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеева@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru,6303 | Сергеев,Вячеслав,60  Фадеев,Якун,57  Ширяева,Евпраксия,39  Авдеев,Лукий,54 | 1 |
| Степанова,Полина,Созоновна,Полина@github.ru,Степанова@mail.ru,6303  Мясникова,Екатерина,Федосеевна,Екатерина@github.ru,Мясникова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапова@mail.ru,6303  Герасимов,Игнатий,Парфеньевна,Игнатий@github.ru,Герасимова@mail.ru,6304  Кузьмин,Василий,Протасьевна,Василий@github.ru,Кузьмина@mail.ru,6304  Попов,Юлиан,Романович,Юлиан@github.ru,Попова@mail.ru,6304  Мартынова,Надежда,Егоровна,Надежда@github.ru,Мартынов@mail.ru,6304  Третьякова,Элеонора,Денисович,Элеонора@github.ru,Третьяков@mail.ru,6304 | Попов,Юлиан,32  Степанова,Полина,78  Третьякова,Элеонора,71  Мартынова,Надежда,78  Мясникова,Екатерина,78  Потапов,Вячеслав,32  Герасимов,Игнатий,20  Кузьмин,Василий,12 | 3 |

* Пример работы программы при отсутствии одного из входных файлов

|  |  |  |
| --- | --- | --- |
| input\_file\_1 | input\_file\_2 | Результат |
| //Отсутствует | Фадеев,Вячеслав,60  Авдеев,Лукий,54  Степанова,Полина,73  Третьякова,Элеонора,71 | input\_file\_1.csv doesn't exist |
| Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Ширяева,Евпраксия,Артёмовна,Евпраксия@github.ru,Ширяева@mail.ru,6305  Кузьмина,Василиса,Протасьевна,Василиса@github.ru,Кузьмина@mail.ru,6304 | //Отсутствует | input\_file\_2.csv doesn't exist |
| //Отсутствует | //Отсутствует | input\_file\_1.csv doesn't exist  input\_file\_2.csv doesn't exist |

* Примеры работы программы при отсутствии в файле одного из параметров (фамилия, имя, Git, email или группа для 1го файла; фамилия, имя или балл для 2го файла)

|  |  |  |
| --- | --- | --- |
| input\_file\_1 | input\_file\_2 | Результат |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  ,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru,6303 | Сергеев,Вячеслав,60  Фадеев,Якун,57  Ширяева,Евпраксия,39  Авдеев,Лукий,54 | Failed with <Surname. File №1> |
| Фадеев,,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru,6303 | Сергеев,Вячеслав,60  Фадеев,Якун,57  Ширяева,Евпраксия,39  Авдеев,Лукий,54 | Failed with <Name. File №1> |
| Степанова,Полина,Созоновна,Полина@github.ru,Степанова@mail.ru,6303  Мясникова,Екатерина,Федосеевна,,Мясникова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапов@mail.ru,6303  Кузьмин,Василий,Протасьевна,Василий@github.ru,Кузьмин@mail.ru,6304  Попов,Юлиан,Романович,Юлиан@github.ru,Попов@mail.ru,6304 | Попов,Юлиан,32  Степанова,Полина,78  Мясникова,Екатерина,78  Потапов,Вячеслав,32  Кузьмин,Василий,12 | Failed with <Git> |
| Степанова,Полина,Созоновна,Полина@github.ru,Степанова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапова@mail.ru,6303  Герасимов,Игнатий,Парфеньевна,Игнатий@github.ru,Герасимов@mail.ru,6304  Кузьмин,Василий,Протасьевна,Василий@github.ru,,6304  Попов,Юлиан,Романович,Юлиан@github.ru,Попов@mail.ru,6304 | Попов,Юлиан,32  Степанова,Полина,78  Потапов,Вячеслав,32  Герасимов,Игнатий,20  Кузьмин,Василий,12 | Failed with <Mail> |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru | Сергеев,Вячеслав,60  Фадеев,Якун,57  Ширяева,Евпраксия,39  Авдеев,Лукий,54 | Failed with <Group> |
| Степанова,Полина,Созоновна,Полина@github.ru,Степанова@mail.ru,6303  Мясникова,Екатерина,Федосеевна,Екатерина@github.ru,Мясникова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапов@mail.ru,6303  Кузьмин,Василий,Протасьевна,Василий@github.ru,Кузьмин@mail.ru,6304 | Степанова,Полина,78  ,Екатерина,78  Потапов,Вячеслав,32  Кузьмин,Василий,12 | Failed with <Surname. File №2> |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru,6303 | Сергеев,Вячеслав,60  Фадеев,Якун,57  Ширяева,,39  Авдеев,Лукий,54 | Failed with <Name. File №2> |
| Мясникова,Екатерина,Федосеевна,Екатерина@github.ru,Мясникова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапов@mail.ru,6303  Кузьмин,Василий,Протасьевна,Василий@github.ru,Кузьмин@mail.ru,6304 | Мясникова,Екатерина  Потапов,Вячеслав,32  Кузьмин,Василий,12 | Failed with <Score> |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru,6303 | Фадеев,Якун,57  Ширяева,Евпраксия,39  Авдеев,Лукий,ball | Failed with <Score> |

* Пример работы программы при наличии во входных файлах нескольких студентов с одним именем и фамилией.

|  |  |  |
| --- | --- | --- |
| input\_file\_1 | input\_file\_2 | Результат |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Авдеев,Лукий,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru | Сергеев,Вячеслав,60  Фадеев,Якун,57  Ширяева,Евпраксия,39  Авдеев,Лукий,54 | Failed with <One name for a few students. File №1> |
| Степанова,Полина,Созоновна,Полина@github.ru,Степанова@mail.ru,6303  Мясникова,Екатерина,Федосеевна,Екатерина@github.ru,Мясникова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапов@mail.ru,6303  Кузьмин,Василий,Протасьевна,Василий@github.ru,Кузьмин@mail.ru,6304 | Степанова,Полина,78  Мясникова,Екатерина,78  Степанова,Полина,32  Кузьмин,Василий,12 | Failed with <One name for a few students. File №2> |

* Пример работы программы при разном количестве студентов во входных файлах.

|  |  |  |
| --- | --- | --- |
| input\_file\_1 | input\_file\_2 | Результат |
| Мясникова,Екатерина,Федосеевна,Екатерина@github.ru,Мясникова@mail.ru,6303  Потапов,Вячеслав,Павлович,Вячеслав@github.ru,Потапов@mail.ru,6303 | Степанова,Полина,78  Мясникова,Екатерина,78  Потапов,Вячеслав,32  Кузьмин,Василий,12 | Failed with <Different Numbers of Students> |
| Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303  Ширяева,Евпраксия,Артёмович,Евпраксия@github.ru,Ширяева@mail.ru,6303 | Сергеев,Вячеслав,60  Авдеев,Лукий,54 | Failed with <Different Numbers of Students> |

* Пример работы программы при отсутствии студента из 2го файла в 1ом

|  |  |  |
| --- | --- | --- |
| input\_file\_1 | input\_file\_2 | Результат |
| Фадеев,Якун,Сергеевич,Якун@github.ru,Фадеев@mail.ru,6303  Авдеев,Лукий,Созонович,Лукий@github.ru,Авдеев@mail.ru,6303  Сергеев,Вячеслав,Андреевич,Вячеслав@github.ru, Сергеев@mail.ru,6303 | Сергеев,Вячеслав,60  Ширяева,Евпраксия,39  Авдеев,Лукий,54 | Failed with <Student isn't found> |

# 

# Заключение

В ходе работы была создана программа обработки двух csv-файлов, выполняющая необходимые действия. Также для работы программы были созданы и описаны все необходимые функции и структуры.

# Список использованных источников

1. Язык программирования СИ / Керниган Б., Ритчи Д. СПб.: Издательство "Невский Диалект", 2001. 352 с.

2. UNIX. Программное окружение / Керниган Б., Пайк Р. СПб.: Символ Плюс, 2003. 416 с.

# Приложение А

**Исходный код программы**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale.h>

#define MAX\_NUM 1000

typedef struct Students

{

char Surname[30];

char Name[30];

int Score;

} Students;

int ValidityOfString(char \*string, int ID)

{

char tmp[200];

unsigned int i = 0, count = 1;

while ((string[i] != '\n') && (i<strlen(string)))

{

int j = 0;

while ((string[i] != ';') && (string[i] != ',') && (i<strlen(string)))

{

tmp[j] = string[i];

i++;

j++;

}

tmp[j] = '\0';

if (tmp[0] == '\0')

return count;

if ((ID == 3) && (count == 3))

{

int score = atoi(tmp);

if ((score == 0) && (strcmp(tmp,"0")!=0))

return 3;

}

count++;

i++;

}

if ((count - 1) != ID)

return count;

else

return 0;

}

int ParametrOfError(int ErrorCode, int ID)

{

switch (ErrorCode)

{

case 0: return 0;

case 1: if (ID == 6) printf("Failed with <Surname. File №1>\n"); else printf("Failed with <Surname. File №2>\n"); break;

case 2: if (ID == 6) printf("Failed with <Name. File №1>\n"); else printf("Failed with <Name. File №2>\n"); break;

case 3: if (ID == 6) return 0; else printf("Failed with <Score>\n"); break;

case 4: printf("Failed with <Git>\n"); break;

case 5: printf("Failed with <Mail>\n"); break;

case 6: printf("Failed with <Group>\n"); break;

}

return -1;

}

int SaveStud (char \*string, Students \*StudFromSec, int num)

{

char \*Surname = strtok(string, ";,");

char \*Name = strtok(NULL, ";,");

char \*Score = strtok(NULL, ";,");

if (num>1)

for (int i=0; i<num; i++)

if ((strcmp(Surname, StudFromSec[i].Surname) == 0) && (strcmp(Name, StudFromSec[i].Name) == 0))

return -1;

strcpy(StudFromSec[num].Surname, Surname);

strcpy(StudFromSec[num].Name, Name);

StudFromSec[num].Score =atoi(Score);

return 0;

}

int Search(FILE \*In1, Students StudFromSec)

{

char StringFromFirst[200];

int count = 0, score=0;

while (!feof(In1))

{

fscanf(In1, "%s", StringFromFirst);

if (StringFromFirst[0] != '\0')

{

char \*Surname1 = strtok(StringFromFirst, ";,\n");

if (strcmp(Surname1, StudFromSec.Surname) == 0)

{

char \*Name1 = strtok(NULL, ";,\n");

if (strcmp(Name1, StudFromSec.Name) == 0)

{

count++;

}

}

}

StringFromFirst[0] = '\0';

}

if (count > 1)

return -2;

else

if (count == 1)

{

return StudFromSec.Score;

}

else return -1;

}

int ValidOfFile(FILE \*file1, FILE \*file2, Students \*StudFromSec)

{

int i1 = 0, i2 = 0;

if ((file1 == NULL)||(file2 == NULL))

{

if (file1 == NULL)

printf("input\_file\_1.csv doesn't exist\n");

if (file2 == NULL)

printf("input\_file\_2.csv doesn't exist\n");

return -3;

}

char \*string = (char \*)malloc(200);

while (!feof(file1))

{

fscanf(file1, "%s", string);

if (string[0] != '\0')

{

if (ParametrOfError(ValidityOfString(string, 6), 6) == -1)

{

return -2;

}

i1++;

string[0] = '\0';

}

}

while (!feof(file2))

{

fscanf(file2, "%s", string);

if (string[0] != '\0')

{

if (ParametrOfError(ValidityOfString(string, 3), 3) == -1)

{

return -2;

}

if (SaveStud(string, StudFromSec, i2)==-1)

return -4;

i2++;

string[0] = '\0';

}

}

fclose(file1);

fclose(file2);

free(string);

if (i1 == i2)

return i1;

return -1;

}

void BeforeEnd(FILE \*file1, FILE \*file2, Students \*StudFromSec)

{

fclose(file1);

fclose(file2);

free(StudFromSec);

}

int main()

{

setlocale(LC\_ALL, "");

FILE \*input1, \*input2;

input1 = fopen("input\_file\_1.csv", "r");

input2 = fopen("input\_file\_2.csv", "r");

Students \*StudFromSec = (Students \*)calloc(MAX\_NUM, sizeof(Students));

int Error = ValidOfFile(input1, input2, StudFromSec);

switch(Error)

{

case -1:

{

printf("Failed with <Different Numbers of Students>\n");

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}

case -4:

{

printf("Failed with <One name for a few students. File №2>\n");

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}

case -2:

case -3:

{

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}

default: break;

}

int NumOfStud = Error;

input2 = fopen("input\_file\_2.csv", "r");

int MaxScore = 0, count = 0;

for(int i=0; i<NumOfStud; i++)

{

input1 = fopen("input\_file\_1.csv", "r");

int ScoreOfStud = Search(input1, StudFromSec[i]); switch (ScoreOfStud) {

case -1:

{

printf("Failed with <Student isn't found>\n");

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}

case -2:

{

printf("Failed with <One name for a few students. File №1>\n");

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}

case -3:

{

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}

}

if (ScoreOfStud > MaxScore)

{

MaxScore = ScoreOfStud;

count = 1;

}

else if (ScoreOfStud == MaxScore)

count++;

fclose(input1);

}

printf("Max=%d, Count=%d\n", MaxScore, count);

BeforeEnd(input1, input2, StudFromSec);

system("pause");

return 0;

}