**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

Курсовая РАБОТА

**по дисциплине «Программирование»**

Тема: РАБОТА С CSV ФАЙЛАМИ

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Ильяшук Д.И. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**ЗАДАНИЕ на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент Ильяшук Д.И. | | |
| Группа 6303 | | |
| Тема работы: Работа с csv файлами | | |
|  | | |
| Содержание пояснительной записки:   * Содержание * Введение * Описание функций, использующихся для работы программы * Описание работы программы * Примеры работы программы * Заключение * Список использованных источников * Приложение А. Исходный код программы | | |
|  | | |
| Дата выдачи задания: 28.04.2017 | | |
| Дата сдачи реферата: 02.06.2017 | | |
| Дата защиты реферата: 02.06.2017 | | |
| Студент |  | Ильяшук Д.И. |
| Преподаватель |  | Берленко Т.А. |

**Аннотация**

В данной работе была разработана программа на языке программирования C, которая позволяет работать с данными, хранящимися в cvs файлах, проверять их на корректность и сопоставлять информацию из двух таблиц, для проверки предоставляемой информации.

**содержание**

[**ВВЕДЕНИЕ 5**](#_Toc484176320)

[**ОПИСАНИЕ ФУНКЦИЙ И СТРУКТУР 6**](#_Toc484176321)

[**ОПИСАНИЕ РАБОТЫ ОСНОВНОЙ ЧАСТИ ПРОГРАММЫ 8**](#_Toc484176322)

[**ПРИМЕР РАБОТЫ ПРОГРАММЫ 12**](#_Toc484176323)

[**ЗАКЛЮЧЕНИЕ 14**](#_Toc484176324)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15**](#_Toc484176325)

[**ПРИЛОЖЕНИЕ А 16**](#_Toc484176326)

# ВВЕДЕНИЕ

* Требуется написать программу, на вход которой подаются две таблицы хрянящиеся 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 написали хуже, чем 60% от максимального балла 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>”.

# ОПИСАНИЕ ФУНКЦИЙ И СТРУКТУР

* 1. **Получение информации о студенте**

char\* getParameter(char\*\* strptr)

{

char\* startPtr = \*strptr;

int i = 0;

while (((\*strptr)[i] != ';') && ((\*strptr)[i] != '\0'))

i++;

(\*strptr)[i] = '\0';

(\*strptr) += (strlen(startPtr) + 1);

return startPtr;

}

Функция принимает в качестве аргумента указатель на строку. Данная функция выделяет один параметр из строки информации о студенте и возвращает его, при этом передвигая указатель на строку информации на оставшуюся часть строки.

* 1. **Проверка на повторения**

int checkForTheSame(Student\* student, int nubmerOfStudents)

{

int i, j;

for (i = 0; i < (nubmerOfStudents - 1); i++)

for (j = (i + 1); j < nubmerOfStudents; j++)

if (!strcmp(student[i].data[3], student[j].data[3]) && !strcmp(student[i].data[4], student[j].data[4]))

return 1;

return 0;

}

Функция принимает в качестве аргументов массив структур с информацией о студентах и число студентов. Затем производится проверка на повторение одного и того же студента в файле. Допускается одинаковые ФИО, но email и github аккаунт должны быть различными. Функция возвращает 0 если повторений нет, и 1 если верно обратное.

* 1. **Вывод ошибки**

void printError(int number, int file)

{

switch (number)

{

case 0:printf("Fail with Surname"); break;

case 1:printf("Fail with Name"); break;

case 2:printf("Fail with Middle name"); break;

case 3:printf("Fail with Git account"); break;

case 4:printf("Fail with Email"); break;

case 5:printf("Fail with Group"); break;

}

printf("(File %d)\n", file);

}

Функция принимает в качестве аргументов номер отсутствующего параметра и номер файла, в котором параметр отсутствует. Затем функция выводит на консоль ошибку и файл, в котором она произошла.

* 1. **Информация о студентах и баллах за экзамен**

typedef struct Student

{

char data[6][LENGTH];

}Student;

typedef struct Mark

{

char data[2][LENGTH];

int mark;

int inUse;

}Mark;

Для хранения информации о студентах и их баллах за экзамен используются структуры двух типов: struct Mark и struct Student. Структура struct Student используется для хранения данных о студентах из первого файла, а структура struct Mark используется для хранения данных из второго файла, также в ней присутствует флаг, использующийся при сверке данных обоих файлов. Для удобства были созданы новые типы Student и Mark.

# ОПИСАНИЕ РАБОТЫ ОСНОВНОЙ ЧАСТИ ПРОГРАММЫ

* 1. **Считывание информации из файла**

FILE \*file1 = fopen("input\_file\_1.csv", "r");

if (!file1)

{

printf("Fail with first file(doesn`t exists)\n");

return 0;

}

fseek(file1, 0, SEEK\_END);

int sizeofFile1 = ftell(file1);

if (sizeofFile1 <= 2)

{

printf("Fail with first file(empty file)\n");

fclose(file1);

return 0;

}

char \*table1 = (char\*)calloc(sizeofFile1, sizeof(char));

rewind(file1);

fread(table1, sizeof(char), sizeofFile1, file1);

fclose(file1);

При считывании информации из файла происходит проверка на наличие файла и его пустоту. Если файл отсутствует или пустой, происходит вывод ошибки и завершение программы.

Аналогичным образом происходит считывание информация из второго файла.

* 1. **Обработка данных из первого файла**

int i, j;

int numberOfStudentsFile1 = 0;

int arrayLength = 10;

int index = 0;

char\* buffer;

Student \* student = (Student\*)malloc(arrayLength \* sizeof(Student));

char\* string = strtok(table1, "\n");

while (string)

{

numberOfStudentsFile1++;

if (numberOfStudentsFile1 == (arrayLength + 1))

{

arrayLength += 10;

student = (Student\*)realloc(student, sizeof(Student)\* arrayLength);

}

for (i = 0; i < 6; i++)

{

buffer = getParameter(&string);

strcpy(student[index].data[i], buffer);

if (!strcmp(buffer, ""))

{

printError(i, 1);

free(student);

return 0;

}

}

index++;

string = strtok(NULL, "\n");

}

free(table1);

Строка с данными разбивается на лексемы, содержащие информацию об одном студенте, затем эти лексемы разбиваются на отдельные параметры с помощью функции **getParameter,** а затем полученные параметры записываются в структуры. При отсутствии какого-либо параметра, на экран будет выведена ошибка и программа завершится.

* 1. **Проверка на повторения**

if (checkForTheSame(student, numberOfStudentsFile1))

{

printf("Fail with the same students(File 1)\n");

free(student);

return 0;

}

Затем происходит проверка на отсутствие повторений студентов в первом файле с помощью функции **checkForTheSame**. При наличии повторений программа выводит ошибку и завершается.

* 1. **Обработка данных из второго файла**

arrayLength = numberOfStudentsFile1;

index = 0;

Mark \* marks = (Mark\*)malloc(sizeof(Mark) \* arrayLength);

string = strtok(table2, "\n");

int numberOfStudentsFile2 = 0;

while (string)

{

numberOfStudentsFile2++;

if (numberOfStudentsFile2 == (arrayLength + 1))

{

arrayLength += 10;

marks = (Mark\*)realloc(marks, sizeof(Mark)\* arrayLength);

}

for (i = 0; i < 2; i++)

{

buffer = getParameter(&string);

strcpy(marks[index].data[i], buffer);

if (!strcmp(buffer, ""))

{

printError(i, 2);

free(marks);

free(student);

return 0;

}

}

buffer = getParameter(&string);

marks[index].mark = atoi(buffer);

marks[index].inUse = 0;

string = strtok(NULL, "\n");

index++;

}

free(table2);

Обработка данных из второго файла происходит аналогично первому, однако здесь считываются только имя и фамилия, а также балл за экзамен.

* 1. **Сопоставление числа студентов в файлах**

if (numberOfStudentsFile1 != numberOfStudentsFile2)

{

printf("Fail with number of students in files\n");

free(marks);

free(student);

return 0;

}

Если количество студентов в файлах не совпадает, программа выводит ошибку и завершается.

* 1. **Сопоставление студентов из двух файлов**

int correct;

for (i = 0; i < numberOfStudentsFile1; i++)

{

correct = 0;

for (j = 0; j < numberOfStudentsFile1; j++)

{

if (!strcmp(student[i].data[0], marks[j].data[0]) && !strcmp(student[i].data[1], marks[j].data[1]))

{

if (!marks[j].inUse)

{

marks[j].inUse = 1;

correct = 1;

}

}

}

if (!correct)

{

printf("Fail with students(different students in files)\n");

free(marks);

free(student);

return 0;

}

}

Происходит сопоставление каждому студенту из первого файла, его же из второго файла. Если обнаруживается несовпадение, то программа выводит ошибку и завершается.

* 1. **Подсчет числа студентов, написавших хуже 60% от максимального балла**

int max = marks[0].mark;

for (i = 1; i < numberOfStudentsFile1; i++)

{

if (marks[i].mark > max)

max = marks[i].mark;

}

double sixtyPer = max\*(0.6);

int count = 0;

for (i = 0; i < numberOfStudentsFile1; i++)

if (marks[i].mark < sixtyPer)

count++;

printf("%d\n", count);

free(student);

free(marks);

Происходит поиск максимального балла, а затем подсчёт числа студентов, написавших хуже, чем 60% от максимального балла. Затем освобождается выделенная память и программа завершается.

# 

# ПРИМЕР РАБОТЫ ПРОГРАММЫ

Для демонстрации всех возможностей программы используется следующие файлы:

* Input\_file\_1.csv
* Input\_file\_2.csv

И вывода:

* Ошибка (при некорректных данных)
* Искомый результат (при корректных данных)

Корректные данные:

|  |  |  |  |
| --- | --- | --- | --- |
| № | input\_file\_1.csv | input\_file\_2.csv | вЫВОД |
| 1 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 | Бушков;Сергей;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | 4 |
| 2 | Абдиваитова;Далия;Махмудовна;viva41@mail.ru;Najar;6303  Абдуллаев;Анар;Таироглы;vlad.sh49@inbox.ru;Nilatus;6304  Авдонин;Андрей;Владимирович;kornev-av@yandex.ru;Ishndar;6305  Агапов;Илья;Александрович;rozakep99@mail.ru;Adoralar;6307  Адаменко;Олеся;Николаевна;irina251277@mail.ru;Monin;6308  Адыг;Угулза;Григорьевна;chib-chi@yandex.ru;Arann;6309  Азарова;Диана;Вадимовна;toma555@list.ru;Doshura;6310  Азима;Артем;Валерьевич;fedpoch@gmail.com;Kajizragore;6311  Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316  Вакутина;Александра;Владиславовна;azs9292@yandex.ru;Goltile;6317  Валеева;Гузель;Рашитовна;mc-deman@yandex.ru;Dabandis;6318  Валиахметова;Эльвина;Ильдусовна;sanyaron@gmail.com;Karr;6319  Валиев;Тимур;Маратович;plastilin1981@bk.ru;Arantrius;6320  Гуринова;Екатерина;Евгеньевна;spicin71@mail.ru;Manardin;6321  Гурман;Арина;Юрьевна;anashko@list.ru;Nalmenn;6322  Дударь;Александр;Михайлович;nzaharia77@gmail.com;Tokree;6324  Дульзон;Дарья;Алексеевна;m.zahir@mail.ru;Morluth;6325  Дурова;Наталья;Сергеевна;tsoy53@gmail.com;Dahelm;6326  Дьяченко;Мария;Леонидовна;spacegt@mail.ru;Opidor;6327 | Абдиваитова;Далия;100  Абдуллаев;Анар;60  Авдонин;Андрей;65  Агапов;Илья;26  Адаменко;Олеся;15  Адыг;Угулза;4  Азарова;Диана;8  Азима;Артем;15  Бушков;Сергей;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;12  Вакутина;Александра;15  Валеева;Гузель;48  Валиахметова;Эльвина;45  Валиев;Тимур;26  Гуринова;Екатерина;15  Гурман;Арина;4  Гурьева;Наталья;8  Дульзон;Дарья;51  Дурова;Наталья;46  Дьяченко;Мария;15 | 20 |

Некорректные данные:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Input\_file\_1.csv | Input\_file\_2.csv | вЫВОД |
| 1 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 | Отсутствует | Fail with second file(doesn`t exists) |
| 2 | Отсутствует | Бушков;Сергей;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | Fail with first file(doesn`t exists) |
| 3 | Бушков;Сергей;Витальевич;;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 | Бушков;Сергей;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | Fail with Email (File 1) |
| 4 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 | Бушков;;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | Fail with name (File 2) |
| 5 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 |  | Fail with second file(empty file) |
| 6 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 | Бушков;Сергей;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | Fail with the same students(File 1) |
| 7 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316  Жук;Виктория;Павловна;nvlevina09@gmail.com;Fenrishicage;6337 | Бушков;Сергей;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | Fail with number of students in files |
| 8 | Бушков;Сергей;Витальевич;shela@shela71.ru;Fenrigor;6312  Бушова;Элеонора;Сергеевна;krisa72@list.ru;Brightstone;6313  Быкова;Ирина;Сергеевна;fanatka2003@yandex.ru;Zolotaxe;6314  Ваапов;Бейтулла;Рустемоглы;moroz.lik@yandex.ru;Shanius;6315  Вакорина;Анна;Игоревна;vibra4ka@mail.ru;Akinolmaran;6316 | Бушков;**Семён**;51  Бушова;Элеонора;46  Быкова;Ирина;15  Ваапов;Бейтулла;15  Вакорина;Анна;100 | Fail with students(different students in files) |

# ЗАКЛЮЧЕНИЕ

Была реализована программа на языке программирования С, обрабатывающая информацию из csv файлов. Программа считает число студентов, получивших балл ниже, чем 60% от максимального балла. Также обрабатываются многие ошибки, возможные при предоставлении некорректной информации либо ее отсутствии.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Язык программирования СИ / Керниган Б., Ритчи Д. СПб.: Издательство «Невский Диалект», 2001. 352 с.
  2. Основы программирования на языках C и C++ [Электронный ресурс]. URL: http://cppstudio.com/

# ПРИЛОЖЕНИЕ А

# ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale.h>

#define LENGTH 30

/\*1 - surname

2 - name

3 - middle name

4 - git account

5 - email

6 - group\*/

typedef struct Student

{

char data[6][LENGTH];

}Student;

typedef struct Mark

{

char data[2][LENGTH];

int mark;

int inUse;

}Mark;

char\* getParameter(char\*\* strptr);

void printError(int number, int file);

int checkForTheSame(Student\* student, int nubmerOfStudents);

int main()

{

setlocale(LC\_ALL, "RUS");

//Read files and checking them for emptiness

FILE \*file1 = fopen("input\_file\_1.csv", "r");

if (!file1)

{

printf("Fail with first file(doesn`t exists)\n");

return 0;

}

fseek(file1, 0, SEEK\_END);

int sizeofFile1 = ftell(file1);

if (sizeofFile1 <= 2)

{

printf("Fail with first file(empty file)\n");

fclose(file1);

return 0;

}

char \*table1 = (char\*)calloc(sizeofFile1, sizeof(char));

rewind(file1);

fread(table1, sizeof(char), sizeofFile1, file1);

fclose(file1);

FILE \*file2 = fopen("input\_file\_2.csv", "r");

if (!file2)

{

printf("Fail with second file(doesn`t exists)\n");

return 0;

}

fseek(file2, 0, SEEK\_END);

int sizeofFile2 = ftell(file2);

if (sizeofFile2 <= 2)

{

printf("Fail with second file(empty file)\n");

fclose(file2);

return 0;

}

char \*table2 = (char\*)calloc(sizeofFile2, sizeof(char));

rewind(file2);

fread(table2, sizeof(char), sizeofFile2, file2);

fclose(file2);

//Get information from input\_file\_1

int i, j;

int numberOfStudentsFile1 = 0;

int arrayLength = 10;

int index = 0;

char\* buffer;

Student \* student = (Student\*)malloc(arrayLength \* sizeof(Student));

char\* string = strtok(table1, "\n");

while (string)

{

numberOfStudentsFile1++;

if (numberOfStudentsFile1 == (arrayLength + 1))

{

arrayLength += 10;

student = (Student\*)realloc(student, sizeof(Student)\* arrayLength);

}

for (i = 0; i < 6; i++)

{

buffer = getParameter(&string);

strcpy(student[index].data[i], buffer);

if (!strcmp(buffer, ""))

{

printError(i, 1);

free(student);

return 0;

}

}

index++;

string = strtok(NULL, "\n");

}

free(table1);

//Check for the repetitions

if (checkForTheSame(student, numberOfStudentsFile1))

{

printf("Fail with the same students(File 1)\n");

free(student);

return 0;

}

//Get information from input\_file\_2

arrayLength = numberOfStudentsFile1;

index = 0;

Mark \* marks = (Mark\*)malloc(sizeof(Mark) \* arrayLength);

string = strtok(table2, "\n");

int numberOfStudentsFile2 = 0;

while (string)

{

numberOfStudentsFile2++;

if (numberOfStudentsFile2 == (arrayLength + 1))

{

arrayLength += 10;

marks = (Mark\*)realloc(marks, sizeof(Mark)\* arrayLength);

}

for (i = 0; i < 2; i++)

{

buffer = getParameter(&string);

strcpy(marks[index].data[i], buffer);

if (!strcmp(buffer, ""))

{

printError(i, 2);

free(marks);

free(student);

return 0;

}

}

buffer = getParameter(&string);

marks[index].mark = atoi(buffer);

marks[index].inUse = 0;

string = strtok(NULL, "\n");

index++;

}

free(table2);

//If number of students isn`t eual in tables, print error

if (numberOfStudentsFile1 != numberOfStudentsFile2)

{

printf("Fail with number of students in files\n");

free(marks);

free(student);

return 0;

}

//Compare students in tables

int correct;

for (i = 0; i < numberOfStudentsFile1; i++)

{

correct = 0;

for (j = 0; j < numberOfStudentsFile1; j++)

{

if (!strcmp(student[i].data[0], marks[j].data[0]) && !strcmp(student[i].data[1], marks[j].data[1]))

{

if (!marks[j].inUse)

{

marks[j].inUse = 1;

correct = 1;

}

}

}

if (!correct)

{

printf("Fail with students(different students in files)\n");

free(marks);

free(student);

return 0;

}

}

//Find maximal mark and count how many students got less than 60% of maximal mark

int max = marks[0].mark;

for (i = 1; i < numberOfStudentsFile1; i++)

{

if (marks[i].mark > max)

max = marks[i].mark;

}

double sixtyPer = max\*(0.6);

int count = 0;

for (i = 0; i < numberOfStudentsFile1; i++)

if (marks[i].mark < sixtyPer)

count++;

printf("%d\n", count);

free(student);

free(marks);

return 0;

}

char\* getParameter(char\*\* strptr)

{

char\* startPtr = \*strptr;

int i = 0;

while (((\*strptr)[i] != ';') && ((\*strptr)[i] != '\0'))

i++;

(\*strptr)[i] = '\0';

(\*strptr) += (strlen(startPtr) + 1);

return startPtr;

}

//Students can have the same names, surname, middle names and groups, but can`t have the same mails and git accounts

int checkForTheSame(Student\* student, int nubmerOfStudents)

{

int i, j;

for (i = 0; i < (nubmerOfStudents - 1); i++)

for (j = (i + 1); j < nubmerOfStudents; j++)

if (!strcmp(student[i].data[3], student[j].data[3]) && !strcmp(student[i].data[4], student[j].data[4]))

return 1;

return 0;

}

void printError(int number, int file)

{

switch (number)

{

case 0:printf("Fail with Surname"); break;

case 1:printf("Fail with Name"); break;

case 2:printf("Fail with Middle name"); break;

case 3:printf("Fail with Git account"); break;

case 4:printf("Fail with Email"); break;

case 5:printf("Fail with Group"); break;

}

printf("(File %d)\n", file);

}