МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионально образования

“Московский авиационный институт

(национальный исследовательский университет)”

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ПРИКЛАДНОЙ МАТЕМАТИКИ**

**Кафедра вычислительной математики и программирования**

Курсовой проект по курсу:

“Основы информатики”

II семестр

Задание 6

“Обработка последовательной файловой структуры на языке Си”

Работу выполнил:

студент 1 курса

Группы М8О-107Б-2018

Гамов Павел Антонович

Проверила:

асп. каф. 806 Ридли Александра Николаевна.

Научный руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2019

Содержание

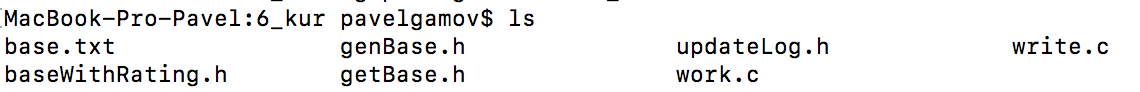
1. Введение и постановка задачи. ………………………………………………3
2. Способы решения и алгоритм. ……………………………………………….3
3. Общая структура документов. ……………………………………………….3
4. updateLog.h. Основная структура. …………………………………………...4
5. base.txt и рейтинг. ……………………………………………………………..4
6. Первая программа write.c …………………………………………………….5
   1. genbase.h …………………………………………………………………...6
   2. baseWithRating.h ……………………………………………………….….7
7. Вторая программа work.c ……………………………………………….……7
   1. getbase.h …………………………………………………………………...7
8. Тесты и пример работы. ……………………………………………………..11
9. Заключение. ………………………………………………………………….11
10. Список использованной литературы. ………………………………………12

**Поставка задачи:**

Использование файловой системы и функций обработки текстовых файлов на языке Си бывает необходимо при написании обработчиков баз данных, программ, принимающих и выдающих в результате большой объём данных, который может быть использован в другой программе. В данных случаях запись и обработка файлов являются решением многих проблем.

Задача звучит так: реализовать на языке Си программы по созданию и обработке файловых структур. Необходимо создать последовательную базу данных, имеющую характеристики компьютеров студентов общежития и вывести последовательно также в файл имена тех, чей компьютер необходимо обновить.

**Способы решения и мои действия, алгоритм:**

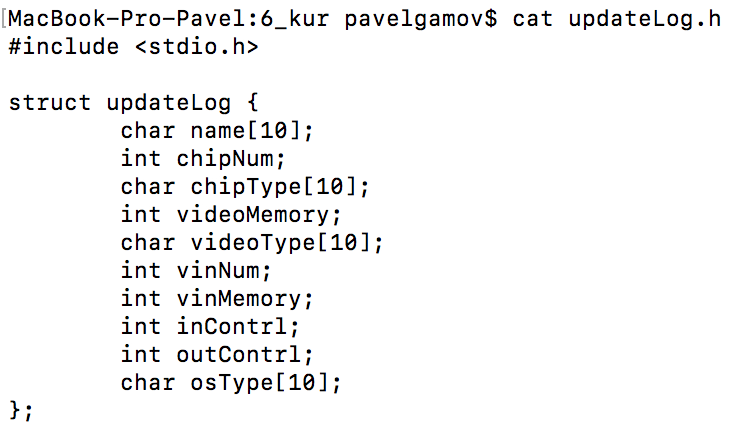


Я решил создать 2 отдельные программы, каждые, из которые выполняют свои функции. Одна из них, write.c, создаёт и записывает двоичную базу данных в файл, получая данные от пользователя.

Принцип работы был завязан на файле base.txt, который содержит таблицы сравнения характеристик компьютеров пользователей. Каждому параметру из каждой категории ставится в сравнение целочисленное значение рейтинга данного параметра, о чем позже.

Второй файл имеет название work.c, и он завязан на получении в качестве аргумента базы данных, которую создал первая программа. В результате работы вторая программа отдает базу данных, которая содержит имена людей, чьи компьютеры отстают по каким-либо параметрам. Форматы и структуры базы данных будут рассмотрены далее.

**UpdateLog.h**



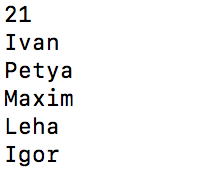
Касательно структуры, которую мы будем разворачивать в наших программах, она содержит 10 параметров, рассмотрим каждый.

Первый параметр name – имя студента, chipNum и chipType – количество и тип центрального процессора соответственно, videoMemory и VideoType – количество видеокарт и их типы, vinum и vinMemory – количество винчестеров и размер одного из них, inControl и outControl – количество подключенных внутренних и внешних периферийных устройств соответственно, osType – вид операционной системы.

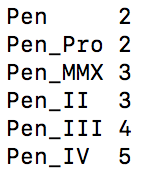
Все чаровые массивы в программах ограничиваем размером 10, потому что не часто встретишь имя длиной больше 10 символов. Хотя можно брать и больше, допустим 100.

**Base.txt**

Теперь, зная разворачиваемую структуру внутри наших программ, мы можем рассмотреть данный файл, который так необходим.



Такова его структура, число параметров и каждая новая строка содержит элемент для каждой ячейки структуры выше, начиная с имени, заканчивая типом операционной системы.



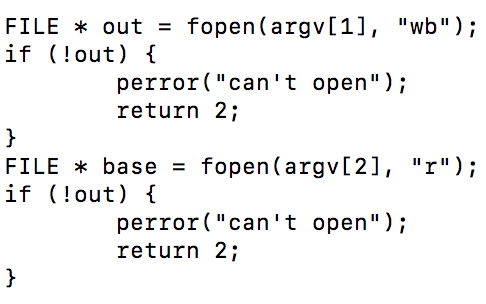
Также там могут присутствовать элементы другого плана, название которых содержит в паре с собой через табуляцию свой вес.

Имеем множество различных версий центрального процессора поколения Pentium. Каждому из них соответствует рейтинг, в зависимости от их крутости на нынешнее время.

Параметры структуры, имеющие вес, учувствуют в составлении общего рейтинга компьютера по отношению к другим машинам. Мы сравниваем Компьютеры по некоторым атрибутам, Наличие элементов имени обусловлено наличием функции авто заполнения базы данных, которую рассмотрим дальше.

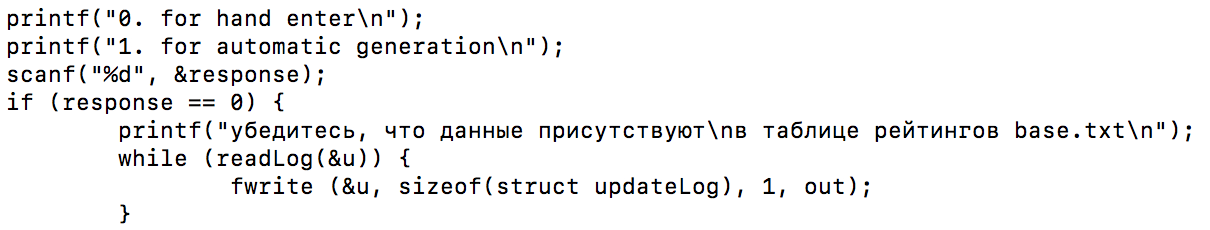
**Write.c**

Мы подошли к генерации базы данных, данная программа принимает на вход 2 параметра: название которым мы назовем выходной файл с двоичной базой данных и файл с таблицами рейтинга base.txt.



Первым делом открываем таблицу рейтингов и создаем бинарный файл для выходной базы данных.

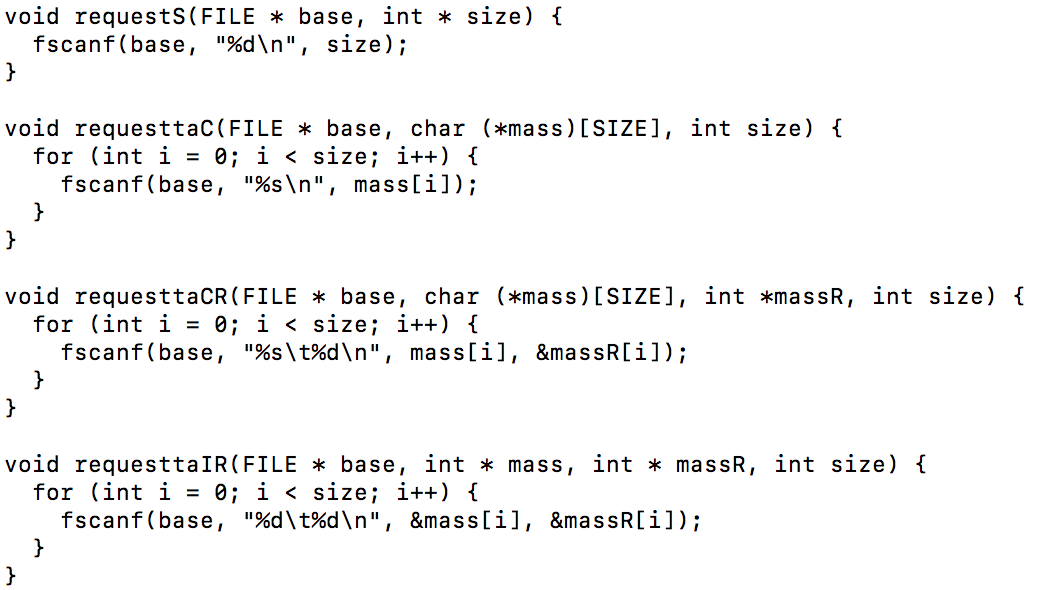




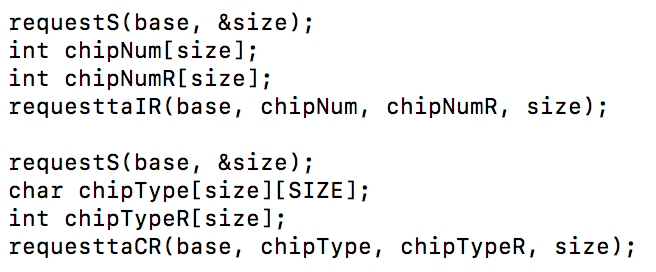
Стандартное меню, на 0 пользователь заполняет данные вручную. Но я сделал авто заполнение с функцией рандома, потому что это весело.

Файл write.c подключает 3 дополнительных файла типа .h. Один из них, это структура updateLog.h.

baseWithRating.h содержит функции для приема и обработки файла base.txt и для записи данных в массивы программы.



Каждые из них работают со своим видом данных, кто-то принимает char, char и int, int и int данные.

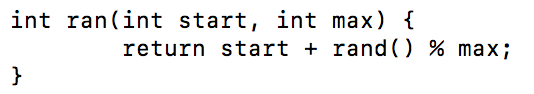


Так выглядит часть кода, которая заполняет в массивы базу данных с рейтингом из файла base.txt. Сначала мы получаем количество получаемых элементов, создаем массивы нужных размеров, заполняем их через функцию типа requestta.

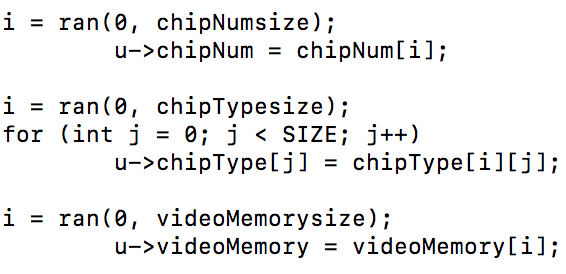
genBase.h содержит необходимые функции для автоматического создания базы данных.



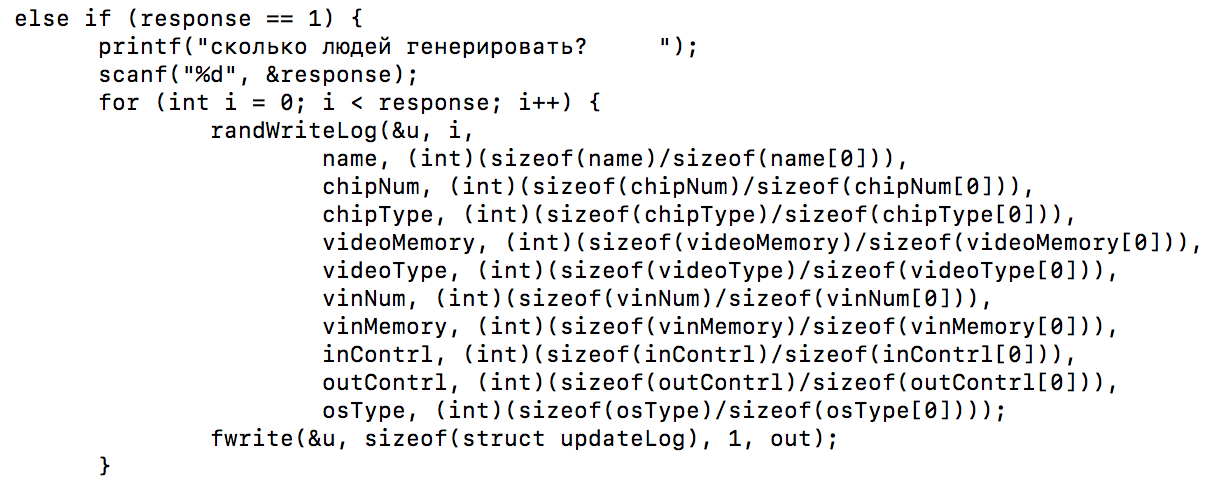
Подключаем time.h ради создания функции рандома на основании скорости работы программы.



Так выглядит функция рандома, ей мы будем брать индексы из массивов таблицы заполнения.



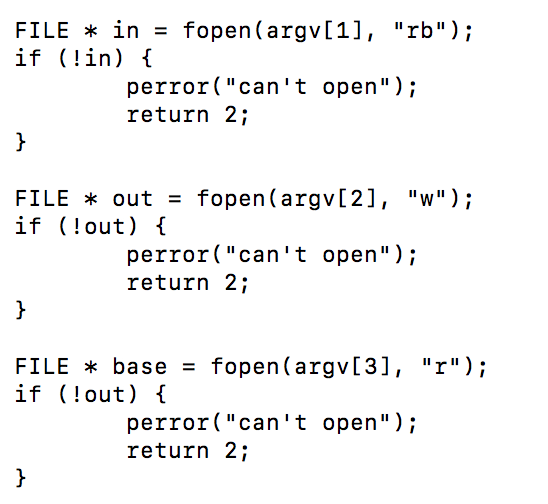
Часть кода функции randWriteLog(), которая заполняет переданную ей структуру.



Таким образом получая данные сколько людей нужно, мы автоматом заполним базу данных случайными значениями.

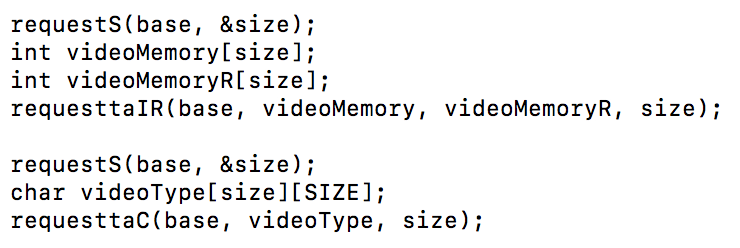
**Work.c**

Данная программа принимает на вход три параметра: название входной базы данных, название выходной базы данных и название базы данных рейтинга.



Открываем сразу три файла: in – входная база, out – результат. Как и в первой, присоединяем 3 файла. updateLog.h – наша структура.

baseWithRating.h – тот самый файл, который мы присоединяли в первой программе, нужен опять для получения таблиц рейтинга и заполнения их в массивы для дальнейшего использования.

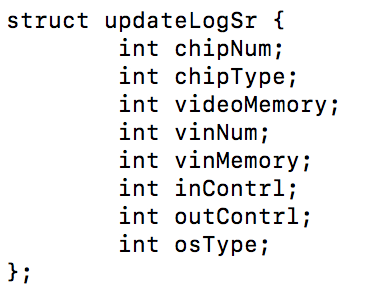


Представлена часть кода. Таким же способом, используя функции, заполняем массивы значений и их рейтингов.

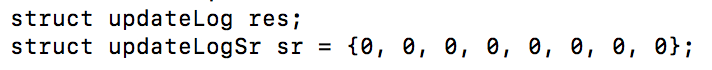
getBase.h – файл, который имеет функции для обработки и создания рейтинга среди студентов принимаемой базы данных.

Идея проста – полученную базу данных прогнать через таблицу рейтинга, попутно записывая веса характеристик каждого компьютера в массив, потом, разделить каждый пункт на количество студентов, таким образом мы получаем средний рейтинг.

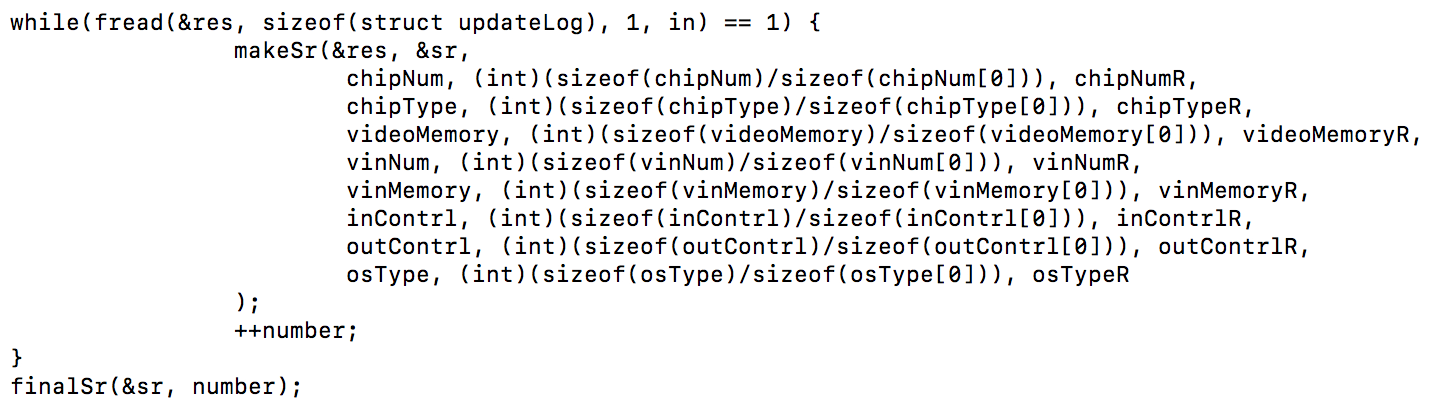
Далее простая сортировка, кто ниже рейтингом – уходят в список на обновление компьютера, кто выше, могут с радостью выдохнуть.



Объявляем структуру схожую нашей основной, но состоящей из целых значений, сюда мы будем прибавлять соответствующие рейтинги поступающих характеристик компьютеров.

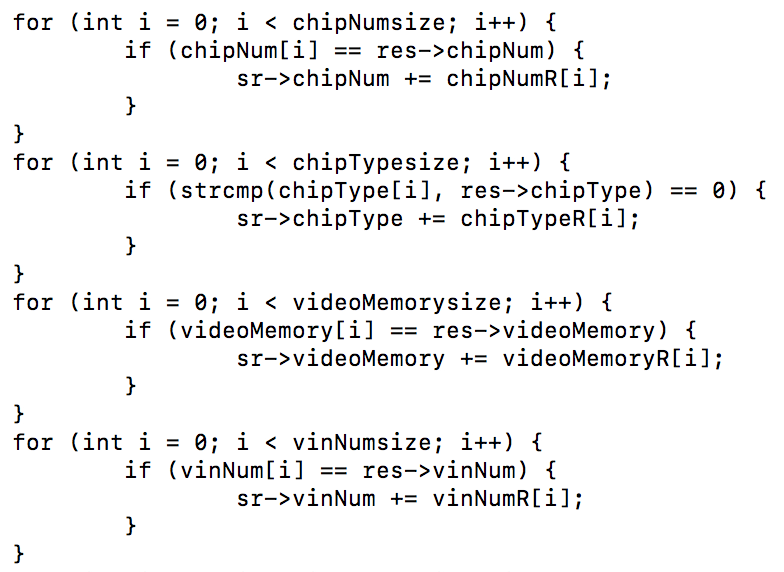


Объявляем структуру нашего студента и ему в пару такую же целочисленную.

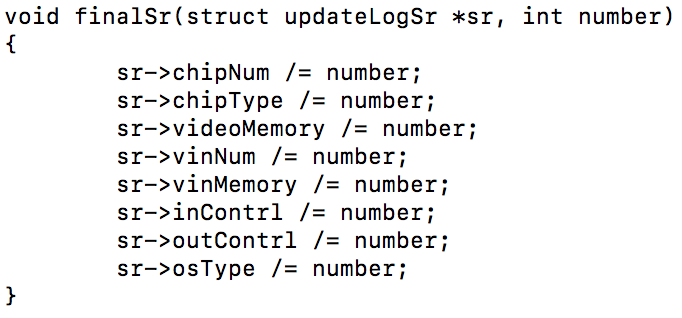


Пока наша база данных не закончится, мы принимаем построчно res, записываем, отдаем в функцию определения рейтинга.

Отдаем в функцию массивы параметров, их числовых рейтингов и размеры данных массивов.

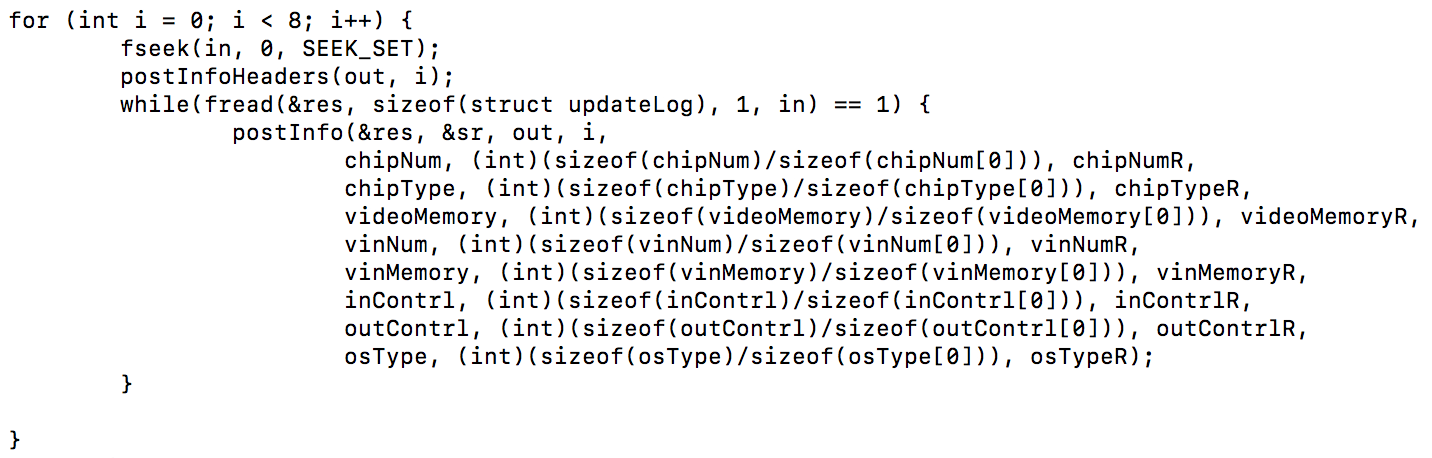


makeSr() – функция принимает структуру, для каждого ее параметра, который учувствует в сравнении и изменении общего рейтинга, она ищет такой же в base.txt, и находя, прибавляет ее рейтинг к структуре sr, которая является по сути массовом рейтингов.

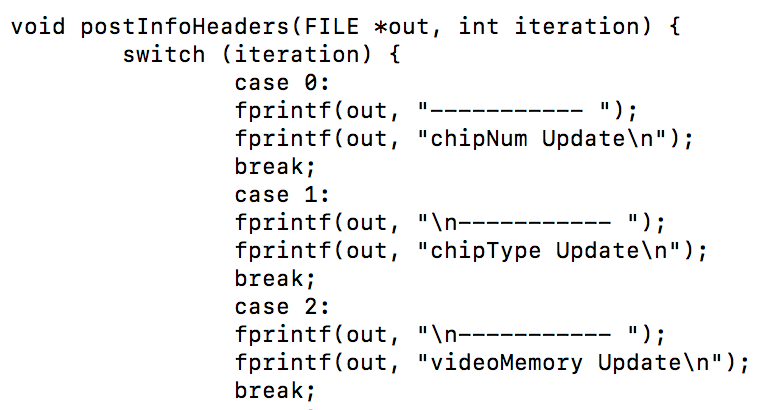


finalSr() – делит каждый параметр подаваемой структуры sr на количество студентов.

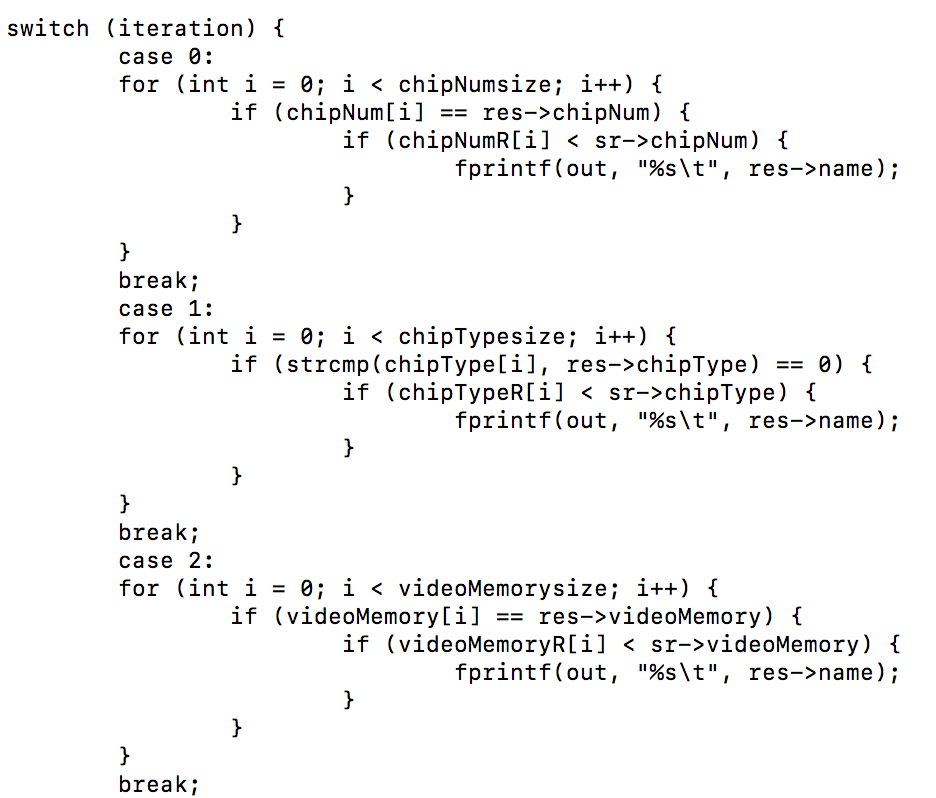
На этом этапе мы уже имеем структуру целых значений рейтингов для каждой характеристики компьютеров. Пришло время войти в цикл и отсечь слабых от сильных.



fseek() – сбрасывает положение прочтенного файла бинарной базы данных.



Часть кода функции postInfoHeaders(). Данный код просто выводит хедер раздела по которому идет отбор. Позднее рассмотрим структуру вывода.



postInfo() – данная функция сравнивает характеристики студента в зависимости от итерации, так, для итерации 1 postInfoHeaders() будет писать сравнение по chipType, а после, данная функция будет выводить имена студентов, которые провалились по этой характеристике.

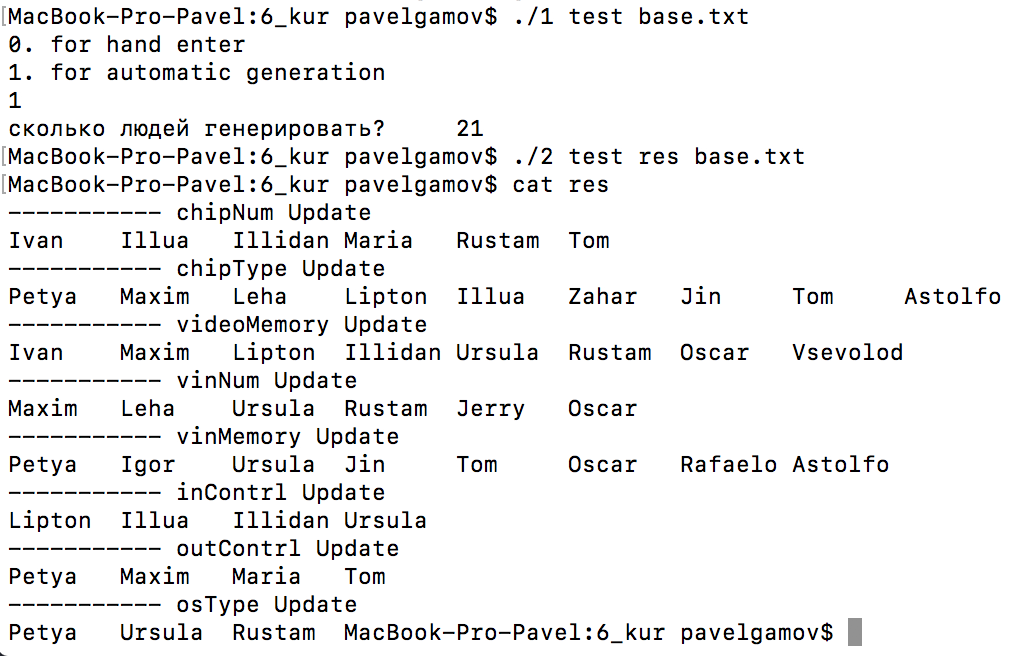
Полученная структура будет иметь такой вид:

ChipType Update

Ivan Petya Ded

И так далее по каждому из характеристик. Теперь каждый студент знает, какую часть компьютера ему необходимо обновить.

**Пример работы:**



**Заключение**

Текстовые базы данных позволяют избежать перебивания информации, позволяют иметь надежное и структурированное хранение нужных данных и легкий доступ к нужным из них.

**Список использованной литературы**

Работа с файлами на Си - https://prog-cpp.ru/c-files/

Программирование на Си урок 37: Работа с файлами в языке Си - https://www.youtube.com/watch?v=6I3wA2yQVis

Базы данных на Си - http://forum.codenet.ru/q26021/Создание+базы+данных+на+Си