Московский авиационный институт
(государственный технический университет)

Факультет «Прикладная математика и физика»
Кафедра «Вычислительная математика и информатика»

Курсовой проект по

Языкам и методам программирования по теме:

«Обработка последовательной файловой структуры на языке Си»

Выполнил: Щербаков А.А.
Студент группы М8О-106Б
Преподаватель: Дубинин А.В.
Оценка:
Дата:

2017 год

Введение

Человеческий мозг способен оперировать большим количеством информации. Чтобы повысить качество его работы необходимо её каким-то образом упорядочивать. Это позволяет не только работать с большим количеством информации более наглядно, но и повысить скорость работы с этой информацией. С развитием компьютерных технологий появились компьютеры, способные оперировать информацией подобно человеку, но гораздо быстрее. И, так же как и человек, компьютер работает быстрее и лучше с упорядоченной информацией. Таким образом, для качественной работы различных программ их входные данные необходимо представить в некотором компактном и удобном виде. Одним из таких представлений является представление в виде последовательной файловой структуры. Рассмотрим его реализацию на примере программы на языке Си. Для этого нужно разобраться с представлением файлов в ОС UNIX и их использование с помощью программ на Си.

Организация файловой системы в ОС UNIX и работа с файлами в Си

Организация файловой системы UNIX имеет древовидную структуру, вершина которой называется корнем, а сама структура называется файловым деревом. Каждая вершина в файловом дереве, за исключением листьев, является каталогом, листья же являются либо обычными файлами, либо файлами устройств. Существует понятие прав доступа к файлу. С помощью установки битов разрешения доступа можно управлять разрешениями на чтение, запись и выполнение для некоторых категорий пользователей. Так же пользователь может создавать файлы в каталоге, если он имеет к нему доступ. Язык программирования Си поддерживает множество функций стандартных библиотек для файлового ввода и вывода. Все файловые операции превращаются в операции с потоками байтов, которые могут быть потоками ввода или вывода. Файл открывается с помощью функции fopen, которая возвращает информацию потока ввода-вывода, прикрепленного к указанному файлу или другому устройству, с которого идет чтение или в который идет запись. В случае неудачи функция возвращает нулевой указатель. Ко всем потокам ввода-вывода можно получить доступ через файловые дескрипторы. Это неотрицательное целое число, которое ядро возвращает процессу, создавшему поток ввода-вывода. Это число является одним из полей структуры FILE, возвращаемой функцией fopen. Fopen способна работать в разных режимах, которые можно изменять в зависимости от нужды:

Режим	Описание	Начало потока
r(b)	Открывает для чтения	начало
w(b)	Открывает для записи (создает, если файл отсутствует), перезаписывает файл	начало
a(b)	Открывает для добавления (создает)	конец
r+(b)	Открывает для чтения и записи	начало
w+(b)	Открывает для чтения и записи, перезаписывает файл	начало
a+(b)	Открывает для чтения и записи (добавляет в случае существования файла)	конец

Значение b означает двоичным режим Си. В текстовом файле строки разделяются символом перевода строки, который при считывании данных связывается некоторой последовательностью символов конца строки, а в двоичном файле данные считываются без какого-либо связывания.

Для закрытия открытого потока используется функция fclose, которая при успехе возвращает 0 и EOF в обратном случае.

Для чтения и записи потока байтов используются функции fread и fwrite соответственно.

Работа с данными в файле

В качестве примера можно использовать файл со строками, в каждой из которых находятся информация о составе комплектующих ПЭВМ некоторых студентов, расположенная в 11 полях: фамилия владельца, число процессоров, тип процессоров, объем оперативной памяти, тип видеоконтроллера, объем видеопамяти, тип винчестеров, их число, общая емкость, количество контроллеров и внешних устройств, операционная система пользователя. Если занести эту информацию в переменную типа структура, то с ней можно беспрепятственно работать. Для расположения данных не в текстовом, а универсальном виде можно их считать, а затем переписать в файл в режиме бинарной записи. После этого данные станут возможны для чтения некоторых универсальных программ. В данном примере можно найти такие конфигурации, которые встречаются в файле более P раз.

Вывод

Несмотря на достоинства функциональности ввода-вывода в языке Си, в нем также имеются и недостатки. Например, язык Си не имеет прямой поддержки произвольного доступа к файлам данных, таким образом, чтобы считать записанную информацию в середине файла, необходимо создать поток, идущий в середине файла, а затем последовательно считывать байты из потока.

Потоковая модель файлового ввода-вывода была популяризована во многом благодаря операционной системе Unix, написанной на языке Си. Большая функциональность многих современных операционных систем унаследовала потоки от Unix, а многие языки унаследовали интерфейс файлового ввода-вывода языка Си с небольшими отличиями.

Подводя итог, можно сказать, что с помощью языка Си можно писать программы, которые будут обрабатывать большие объемы данных достаточно успешно. Некоторые недостатки перекрываются большей наглядностью и удобством.