Приложение А.

Общая постановка задачи.

Требуется разработать Linux-приложение, предназначенное для манипуляции каталогами или обычными файлами. При реализации обязательно использование изученных в лекционном курсе системных вызовов (ОС Linux) ввода-вывода. Использование высокоуровневых средств считается ошибочным.

Приложение должно быть устойчивым к некорректному пользовательскому вводу.

Варианты заданий.

Вариант 1. Разработать программу, которая позволяет удалять из каталога, указанного пользователем только те файлы, которые не соответствуют указанной пользователем маске[1].

Вариант 2. Разработать программу, которая выводит список файлов и для каждого файла размер, любой другой атрибут, и помечает файлы нулевого размера звездочкой (*). Каталог задается пользователем.

Вариант 3. Разработать программу, которая позволяет инвертировать содержимое и имя файла, указанного пользователем. Под инвертированием понимается запись строки в обратном порядке. Инвертируются только короткие имена файлов.

Вариант 4. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен содержать файлы только с заранее указанным пользователем расширением (например, «*.txt», «*.html»)[1].

Вариант 5. Разработать программу, которая позволяет переименовывать файлы следующим образом. Все строчные символы в имени файла заменяются на прописные и наоборот. Имя файла указывается пользователем. Заменяются только короткие имена.

Вариант 6. Разработать программу, которая позволяет перемещать файлы из текущего каталога во вновь создаваемые программой подкаталоги. Короткие имена этих подкаталогов должны быть эквивалентны расширениям перемещаемых файлов.

Вариант 7. Разработать программу, которая позволяет удалять каталоги, содержащие файлы. Имя каталога задается пользователем.

1 Для проверки соответствия имени файла заданной маске можно воспользоваться функцией *fnmatch()*

Вариант 8. Разработать программу, которая удаляет из каталога, указанного пользователем, все пустые файлы.

Вариант 9. Разработать программу, которая перемещает все файлы в указанный пользователем каталог все файлы из его подкаталогов.

Вариант 10. Некоторые файловые системы могут поддерживать версии файлов. Однако, в ОС семейства *UNIX* этого не предусмотрено. Необходимо разработать программу, которая позволяла бы добавлять номер версии к короткому имени файла, заданному пользователем, через символ подчеркивания. Например, для файла *f_1* должен быть создан файл *f_2*. Если файл с таким именем уже существует, то запрашивается подтверждение этой операции или ее отмена. Если номера версии в имени файла нет, то полагать ее равной нулю.

Учесть ограничения на длину имени файла и диапазон значений для чисел (номер версии — беззнаковое короткое целое число). Содержимое файлов может просто копироваться. Расширение в имени файла не должно изменяться.

Вариант 11. Некоторые файловые системы могут поддерживать версии файлов. Однако, в ОС семейства *UNIX* этого не предусмотрено. Необходимо разработать программу, которая позволяла бы добавлять номер версии к короткому имени файла через символ подчеркивания. Например, при появлении 29.02.2012г. очередной версии для файла *f_22.06.2003* может быть создан файл *f_29.02.2012*. Для разграничения фрагментов версии используется точка. Если файл с таким именем уже существует, то запрашивается подтверждение этой операции или ее отмена.

Учесть ограничения на длину имени файла и диапазон дат. Содержимое файлов может просто копироваться. Расширение в имени файла не должно изменяться.

Вариант 12. Некоторые файловые системы могут поддерживать версии файлов. Однако, в ОС семейства *UNIX* этого не предусмотрено. Необходимо разработать программу, которая позволяла бы добавлять номер версии к короткому имени файла через символ подчеркивания. Например, при появлении в 10ч.25мин.32 и 5 десятых секунды очередной версии для файла *f_02.03.04,1* может быть создан файл *f_10.25.32,5*.Для разграничения фрагментов версии используется точка и запятая (доли секунды).

Учесть ограничения на длину имени файла и формат времени. Содержимое файлов может просто копироваться. Расширение в имени файла не должно изменяться.

Вариант 13. Разработать программу, которая позволяет перемещать содержимое файлов, указанных пользователем, в один файл, имя которого также указывается пользователем.

Учесть ограничения на длину имени файла и набор разрешенных символов.

Вариант №14. Разработать программу, которая позволяет копировать содержимое файлов, указанных пользователем, в один файл, имя которого также указывается пользователем.

Учесть ограничения на длину имени файла и набор разрешенных символов.

Вариант 15. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен быть отсортирован по времени создания каждого его элемента – файла. Выводятся как имена файлов, так и время их создания.

Вариант 16. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен быть отсортирован по времени модификации каждого его элемента – файла. Выводятся как имена файлов, так и время их модификации.

Вариант 17. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен быть отсортирован по времени последнего доступа к каждому его элементу – файлу. Выводятся как имена файлов, так и время доступа к ним.

Вариант 18. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен быть отсортирован по размеру каждого его элемента – файла. Выводятся как имена файлов, так и их размер в байтах.

Вариант 19. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен быть отсортирован по имени (без расширения) каждого его элемента – файла. Выводятся как имена файлов, так и их расширения.

Вариант 20. Разработать программу, которая выводит список файлов из указанного пользователем каталога. Данный список должен быть отсортирован по расширению имени каждого его элемента – файла. Выводятся как имена файлов, так и их расширения.

Вариант 21. Разработать программу, которая выводит список файлов из указанного пользователем каталога. В данном списке не должны показываться элементы, чей размер меньше указанного пользователем значения. Выводятся как имена файлов, так и их размер в байтах.

Вариант 22. Разработать программу, которая выводит список файлов из указанного пользователем каталога. В данном списке не должны показываться элементы, чье время создания раньше, чем указанное пользователем значение. Выводятся как имена файлов, так и время их создания.

Вариант 23. Разработать программу, которая выводит список файлов из указанного пользователем каталога. В данном списке не должны показываться элементы, чье время модификации раньше, чем указанное пользователем значение. Выводятся как имена файлов, так и время их модификации.

Вариант 24. Разработать программу, которая выводит список файлов из указанного пользователем каталога. В данном списке не должны показываться элементы, чье время доступа раньше, чем указанное пользователем значение. Выводятся как имена файлов, так и время доступа к ним.

Вариант 25. Разработать библиотеку функций, предназначенных для имитации работы с файлами с записями фиксированной длины (Как известно, в ОС семейства Windows и UNIX-подобных практически не поддерживаются структурированные файлы). Разработать программу, демонстрирующую работу с такими файлами.

В таких структурированных файлах данные представляются в виде записей, длина которых фиксирована в пределах файла (см. рисунок). Доступ к *N*-ой записи осуществляется либо путем последовательного чтения *N-1* предшествующих записей, либо прямо по адресу, вычисленному по ее порядковому номеру. Например, начальный адрес *K*-ой записи равен $L \times K$, если L – длина записи. Над файлами необходимо обеспечить операции создания, удаления, открытия, закрытия, чтения и модификации *K*-ой записи, а также проверку целостности структурированного файла.

Запись 1 длиной *L*	Запись 2 длиной *L*	…	Запись N длиной *L*

Вариант 26. Разработать библиотеку функций, , предназначенных для имитации работы с файлами с записями переменной длины (Как известно, в ОС семейства Windows и UNIX-подобных практически не поддерживаются структурированные файлы). Разработать программу, демонстрирующую работу с такими файлами.

В таких структурированных файлах данные представляются в виде записей, длина которых не фиксирована в пределах файла (см. рисунок). Доступ к *K*-ой записи осуществляется путем последовательного чтения *K-1* предшествующих записей. Над файлами необходимо обеспечить операции создания, удаления, открытия, закрытия, чтения и модификации *K*-ой записи, а также проверку целостности структурированного файла.

Запись 1 длиной L_1	Запись 2 длиной L_2	…	Запись N длиной L_N

Вариант 27. Разработать программу, которая принимает от пользователя адрес каталога и маску файлов, после чего выводит иерархическую структуру каталогов глубиной пять уровней, начиная с местоположения введенного каталога и копирует файлы, соответствующие маске, во вновь созданный каталог «*Temp*», запускает «проводник» для просмотра созданного каталога и ожидает завершения его просмотра, затем каталог «*Temp*» и скопированные файлы удаляются.

Вариант 28. Разработать программу для удаления каталога, имя которого задано пользователем. Каталог может содержать файлы или подкаталоги. Если файл или подкаталог пусты, то они удаляются, в противном случае – перемещаются в «корзину» (*каталог, расположенный в том же каталоге, что и исполняемый модуль программы*). Иерархия файлов и каталогов при перемещении в «корзину» сохраняется. Вывести информацию о проделанной работе:

- количество удаленных / перемещенных в «корзину» файлов;
- суммарный объем файлов, которые были перемещены или удалены.

Вариант №29. Разработать программу, которая позволяет переименовывать файлы следующим образом. К имени каждого непустого файла приписывается через символ «_» имя каталога, в котором файл находится. Название каталога задается пользователем.

Вариант 30. Разработать программу, которая производит обход каталога, указанного пользователем, в глубину[1] и добавляет в него и в каждый его подкаталог файл с коротким именем, указанным пользователем.

Вариант 31. Разработать программу, которая выводит список файлов и для каждого файла десять наиболее часто встречающихся в нем символов с указанием процентного содержания конкретного символа. Имя каталога задается пользователем.

1 Поиск в глубину- http://kvodo.ru/dfs.html

Вариант 32. Разработать библиотеку функций, предназначенных для имитации работы с индексированными файлами в ОС типа Windows или UNIX-подобной (в них в штатном режиме этой поддержки нет). Разработать программу, демонстрирующую работу с такими файлами.

В индексированном файле записи имеют одно или более ключевых (индексных) полей и могут адресоваться путем указания значений этих полей. Для быстрого поиска данных в таком файле предусмотрена специальная индексная таблица, в которой значениям ключевых полей ставится в соответствие адрес внешней памяти. Этот адрес может указывать непосредственно на искомую запись либо на некоторую область памяти, занимаемую несколькими записями, в число которых входит искомая запись. Над файлами необходимо обеспечить операции создания, удаления, открытия, закрытия, чтения и модификации *K*-ой записи, а также проверку целостности структурированного файла.

Индексная таблица может располагаться отдельно от индексированного файла.

Структура индексированного файла

Индексная таблица	Запись 1		Запись 2		…	Запись N	
	Ключ 1	Данные	Ключ 2	Данные	…	Ключ N	Данные

Структура индексной таблицы

Индекс	Ключ 1	Ключ 2	…	Ключ N
Адрес	0x23	0x225	...	0x989