Задания к работе №1 по Системному программированию.

Все задания реализуются на языке программирования C (стандарт C99 и выше).
Реализованные в заданиях приложения не должны завершаться аварийно; все возникающие исключительные ситуации должны быть перехвачены и обработаны.
Во всех заданиях запрещено пользоваться функциями, позволяющими завершить выполнение приложения из произвольной точки выполнения.
Во всех заданиях при реализации необходимо разделять контексты работы с данными (поиск, сортировка, добавление/удаление, модификация и т. п.) и отправка данных в поток вывода / выгрузка данных из потока ввода.
Во всех заданиях все вводимые (с консоли, файла, командной строки) пользователем данные должны (если не сказано обратное) быть подвергнуты валидации в соответствии с типом валидируемых данных.
Во всех заданиях необходимо контролировать ситуации с невозможностью [пере]выделения памяти; во всех заданиях необходимо корректно освобождать всю выделенную динамическую память.
Все ошибки, связанные с операциями открытия файла, должны быть обработаны; все открытые файлы должны быть закрыты.

1. Напишите примитивную оболочку для командной строки. При запуске вашей программы должно быть выведено приглашение к авторизации пользователя. У каждого пользователя есть login (длиной не более 6 символов, который состоит из символов латинского алфавита и цифр) и PIN-код (целое число в системе счисления с основанием 10, значение числа варьируется в диапазоне 0 до 100000). Для работы с оболочкой пользователь должен авторизоваться или зарегистрироваться (интерфейс взаимодействия с пользователем для его регистрации продумайте самостоятельно) в приложении. После удачной авторизации пользователя ему доступен следующий набор команд:
    - Time - запрос текущего времени в стандартном формате чч:мм:сс;
    - Date - запрос текущей даты в стандартном формате дд:мм:гггг;
    - Howmuch <time> flag - запрос прошедшего времени с указанной даты в параметре <time>, параметр flag определяет тип представления результата (-s в секундах, -m в минутах, -h в часах, -y в годах)
    - Logout - выйти в меню авторизации
    - Sanctions username <number> - команда позволяет ввести ограничения на работу с оболочкой для пользователя username, а именно данный пользователь не может в одном сеансе выполнить более <number> запросов. Для подтверждения ограничений после ввода команды необходимо ввести значение 12345.
2. Реализуйте консольное приложение для обработки файлов. Аргументы в Ваше приложение передаются как аргументы командной строки: первые параметры — пути ко входным файлам (может быть один или несколько файлов), следующий параметр – это флаг, который определяет действие, которое необходимо выполнить с файлом(файлами):
    - xorN - сложение по модулю 2 всех блоков файла, длиной 2^N бит, $N = 2, 3, 4, 5, 6$. (Например, xor3: сложение по модулю 2 всех байтов из файлов, xor5: сложение по модулю 2 групп четырехбайтовых значений из файлов). В случае необходимости недостающие байты заполняются значением 00_{16};
    - mask <hex> - подсчет четырёхбайтовых целых чисел из файла, которые соответствуют маске <hex>, которая задана в системе счисления с основанием 16.
    - copyN – создание N копий каждого из переданных файлов. Имена копий создавайте приписыванием номера к исходному имени файла, копии файлов сохраняйте в том же каталоге, где и исходный файл. Операции множественного копирования реализуйте в различных процессах. Данные во входных файлах могут быть произвольной структуры; входной файл и его копии должны быть идентичны после выполнения операции копирования.

- find <SomeString> - поиска строки в наборе текстовых файлов, список имен которых передается в файлах, которые являются начальными аргументами (аргументами командной строки). Задачу поиска строки в отдельном файле выполняйте в отдельном процессе. В случае успешного поиска необходимо вывести на консоль пути к файлам, в которых поисковая строка была обнаружена, в случае неудачного поиска (строка не найдена ни в одном из файлов) необходимо вывести на консоль сообщение об отсутствии строки в заданном файле.

3. Реализуйте приложение, демонстрирующее проблемы в задаче обедающих философов. Разработайте решение этой проблемы, которое основано на семафорах. При демонстрации работы вашего приложения покажите возникающие проблемы при отсутствии синхронизации философов.
4. Крестьянину нужно перевезти через реку волка, козу и капусту. Но лодка такова, что в ней может поместиться только крестьянин, а с ним или один волк, или одна коза, или одна капуста. Но если оставить на берегу волка с козой, то волк съест козу, а если оставить на берегу козу с капустой, то коза съест капусту. Как перевезти свой груз крестьянину? Реализуйте клиент-серверный комплекс приложений, клиент которого принимает на вход команды от пользователя и делегирует их выполнение серверному приложению посредством очереди сообщений. Продумайте возможность обработки сообщений от многих пользователей, для этого разработайте систему идентификации пользователей.

   В силу сложившихся обстоятельств крестьянин понимает достаточно ограниченный набор команд:

   1) take <object> - взять в лодку заданный объект. Вместо <object> может быть написано wolf, goat или cabbage. При этом команда может быть выполнена только если в лодке есть место;
   2) put; - выложить на берег то, что есть в лодке. При этом команда может быть выполнена только если в лодке помимо крестьянина есть ещё какой-либо объект;
   3) move; - переплыть реку на лодке. При этом команда может быть выполнена в любом случае вне зависимости от того, что именно находится в лодке.

   Клиентские приложения передают серверному приложению инструкции по одной из текстового файла (путь к этому файлу передаётся через аргументы строки), в котором лежит вся последовательность инструкций для крестьянина. Придумайте различные сценарии: которые решают данную задачу, а также с неверной последовательностью инструкций. Реализуйте программу так, чтобы были обработаны всевозможные ошибки времени выполнения.
5. (Таненбаум, Гл. 2, задание 61) Некий университет решил продемонстрировать свою политкорректность, применив известную доктрину верховного суда

США «Равенство порознь равенством не является» не только к цвету кожи, но и к полу. Результатом этого решения явились совместные ванные комнаты в общежитиях. Тем не менее, в поддержку исторически сложившийся традиции университет постановляет, что если в ванной комнате есть женщина, то другая женщина может туда зайти, а мужчина не может, и наоборот. На двери ванной есть индикатор, показывающий, в каком из трех состояний находится ванная комната:

1. В ванной никого нет;
2. В ванной только женщины;
3. В ванной только мужчины.

Реализуйте функции: woman_wants_to_enter, man_wants_to_enter, woman_leaves, man_leaves. При реализации функций допускается использование любых счетчиков и средств синхронизации. Примените реализованные функции в приложении, моделирующем работу ванной комнаты. В рамках приложения считается, что максимальное число человек в ванной комнате определяется параметром N, который передается через аргументы командной строки при запуске приложения.

6. Реализуйте серверное приложение, которое обрабатывает наборы строк, являющихся абсолютными путями к файлам. Результатом работы вашего приложения являются абсолютные пути к папкам (каталогам) и для каждого каталога список файлов, которые в них содержатся. Для демонстрации работы вашего приложения разработайте клиентское приложение, которое с помощью любых средств IPC обменивается данными с серверным приложением. Протокол взаимодействия определите самостоятельно.
7. (Таненбаум, Гл. 4, задание 46) Реализуйте собственную версию программы ls. Ваша версия должна принимать в качестве аргумента одно или несколько имен каталогов и для каждого каталога выдавать список всех файлов, содержащихся в этом каталоге, по одной строке на каждый файл. При выводе каждое поле должно форматироваться в соответствии с его типом. Для каждого файла укажите в списке только первый дисковый адрес файла.