Лабораторные работы по курсу «Операционные системы» 2024

Лабораторная работа №1. Архиватор

Задание:

1. Написать программу архиватор на языке C. На вход программе-архиватору передается папка с файлами, файлы имеют различное расширение, результатом работы программы является один файл, объединяющий все файлы из входной директории, а также содержащий заголовок с информацией об именах файлов и их размерах. Учесть возможность вложенной структуры папок.

2. Написать программу разархиватор, разделяющий архивированный файл обратно на файлы с учетом структуры папок.

Дополнительно.

3. Произвести сжатие файла алгоритмами:

1. Алгоритм Хаффмана
2. Алгоритм Lempel-Ziv-Welch (LZW)
3. Алгоритм Deflate (используемый в форматах ZIP, PNG)
4. Алгоритм Burrows-Wheeler Transform (BWT) + Move-to-Front (MTF) + Huffman
5. Алгоритм Run-Length Encoding (RLE)
6. Алгоритм Arithmetic Coding
7. Алгоритм LZ77/LZ78
8. Алгоритм Delta Encoding
9. Алгоритм Prediction by Partial Matching (PPM)
10. Алгоритм Context Mixing

Литература: Основы программирования в Linux. Автор: Мэтью Нейл, Стоунс Ричард. Глава 3. Работа с файлами.

Варианты:

1. Добавить для архивирования и разархивирования ввод пароля, который будет сохраняться в файле-архиве.
2. Добавить в каждый текстовый заархивированный файл информацию о том, когда был сделан архив и когда файлы были извлечены из архива.
3. Предусмотреть возможность разархивировать файлы в выбранную пользователем папку, а также архивировать файлы и папки из введенной пользователем директории.
4. Если среди архивированных файлов попадается архив, то внутренние файлы добавить во внешний архив. Т.е. сделать один общий архив без вложенности (задание без сжатия)
5. Выбрать возможность архивирования в папку «по умолчанию» (файл-архив автоматически помещается в данную папку) и разархивирования в папку «по умолчанию» (в данной папке создается папка с названием архива и в нее отправляются все разархивированные файлы)
6. Проверка копии архива. Если существует файл-архив и определенная папка снова архивируется, то происходит проверка содержимого старого архива на соответствие новым файлам. Ели содержимое совпадает архивирование не происходит.

7. Провести архивирование паки, содержащей архив, который тоже содержит папку с архивом. Обеспечить возможность извлечение из общего архива только фалов, содержащихся во вложенном архиве.
8. Провести архивирование паки, содержащей архив, который тоже содержит папку с архивом. При разархивировании извлечь файлы из всех архивов.
9. При добавлении в архив необходимо сохранять атрибуты файлов, связанные с правами доступа. При разархивировании восстановить права доступа к файлам.
10. Провести процедуру шифрации созданного архива, например, зеркальным отражением всех символов (поменять позиции символов) или сдвигом значений (шифр Цезаря). При разархивировании провести процедуру дешифрации.

Лабораторная работа №2. Управление процессом

1. Написать программу «терминал», которая анализирует входную строку и при обнаружении ключевых слов «ls», «cat», «nice» и «killall» запускает соответствующие процессы.
2. В программе реализовать возможность запуска процессов других программ, например, браузера.
3. Написать обработчики сигналов, например, при получении сигнала CTRL+C завершить запущенный программой процесс.
4. Заменить системный bash на собственный терминал.

Литература: Основы программирования в Linux. Автор: Мэтью Нейл, Стоунс Ричард. Глава 11. Процессы и сигналы.

Варианты:

1. Программа хранит все открытые процессы. Запустить 3 программы через терминал, сигнал CTRL+C должен закрывать программы в последовательности FIFO
2. Программа хранит все открытые процессы. Запустить 3 программы через терминал, сигнал CTRL+C должен закрывать программы в последовательности LIFO
3. Программа хранит все открытые процессы. Обеспечить возможность завершения процесса командой exit с указанием номера процесса.
4. Программа хранит все открытые процессы. Завершаются сразу все процессы.
5. Реализовать пакетный, одновременный запуск программ.
6. Реализовать возможность приостановки и продолжения выполнения процессов через терминал.
7. Добавить функционал для управления приоритетами процессов.
8. Разработать механизм автоматического перезапуска процесса в случае его аварийного завершения.
9. Реализовать логирование действий терминала и выполненных команд.
10. Создать механизм для уведомления пользователя о завершении выполнения процесса.

Лабораторная работа №3. Межпроцессное взаимодействие (Inter-process communication (IPC))

1. Написать следующие программы:

Программа 1. Программа читает заданный файл и выводит его содержимое в поток вывода с применение IPC в соответствии с вариантом.

Программа 2. Программа запускает два экземпляра программы 1 и разными исходными файлами осуществляет с выходными данными программ побитовую операцию XOR и сохраняет результат в файл.

2. Запустить вторую программу и подать на вход текстовый файл и файл, содержащий случайную последовательность.
3. Запустить вторую программу с файлом, полученным во втором пункте и файлом, содержащим случайную последовательность.
4. Сравнить текстовый файл до и после шифрования.

Литература: Основы программирования в Linux. Автор: Мэтью Нейл, Стоунс Ричард. Глава 13. Межпроцессное взаимодействие.

Варианты:

1. Написать программу, которая создает каналы для обмена данными между процессами.
2. Написать программу, которая создает разделяемую память для обмена данными между процессами.
3. Реализовать программу, которая использует механизм очередей сообщений для передачи данных между процессами.
4. Написать программу, использующую INET сокеты для обмена данными между процессами на разных узлах сети.
5. Создать программу, которая использует механизмы сокетов домена UNIX для обмена данными между процессами на одном компьютере.
6. Написать программу, которая использует механизмы сигналов SIGUSR1 и SIGUSR2 для передачи пользовательских данных между процессами.
7. Программа с использованием механизмов сигналов SIGPIPE и SIGCHLD для передачи пользовательских данных между процессами.
8. Реализовать программу, которая использует механизмы очередей сообщений POSIX для передачи данных между процессами.
9. Создать программу, которая использует разделяемую память POSIX для обмена данными между процессами.
10. Написать программу, использующую механизмы RPC (Remote Procedure Call) на основе XML-RPC для взаимодействия между процессами.

Лабораторная работа №4. Многопоточность. Pthread.

Многопоточный фильтр Собела. Программа получает фотографию на вход, делится на pthread
и накладывает фильтр Собела. Каждый поток обрабатывает определенное количество строк исходного изображения. Программа фиксирует время своего выполнения.

Запустить программу с 1,2,4,8, 16 и 32 потоками, зафиксировать время выполнения для каждой реализации. Сделать выводы.

Литература: Основы программирования в Linux. Автор: Мэтью Нейл, Стоунс Ричард. Глава 12. Потоки POSIX.

Варианты для всех студентов одинаковы, но желательно выполнять лабораторную работу на разных машинах.

Пример программы, реализующей фильтр Собела:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

// Функция для применения фильтра Собела к изображению
void applySobelFilter(int image[][3], int result[][3]) {
    int gx, gy;

    // Ядро фильтра Собела по оси X
    int kernelX[3][3] = {
        {-1, 0, 1},
        {-2, 0, 2},
        {-1, 0, 1}
    };

    // Ядро фильтра Собела по оси Y
    int kernelY[3][3] = {
        {1, 2, 1},
        {0, 0, 0},
        {-1, -2, -1}
    };

    for (int i = 1; i < 2; i++) {
        for (int j = 1; j < 2; j++) {
            gx = 0;
            gy = 0;

            // Применяем ядра фильтра Собела к пикселям изображения
            for (int k = -1; k <= 1; k++) {
                for (int l = -1; l <= 1; l++) {
                    gx += image[i + k][j + l] * kernelX[k + 1][l + 1];
                    gy += image[i + k][j + l] * kernelY[k + 1][l + 1];
                }
            }

            // Вычисляем градиент изображения
            result[i][j] = sqrt(gx * gx + gy * gy);
        }
    }
}

int main() {
    int image[3][3] = {
        {10, 20, 30},
        {40, 50, 60},
        {70, 80, 90}
```

```
    };

    int result[3][3];

    applySobelFilter(image, result);

    // Выводим результат применения фильтра Собела к изображению
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Лабораторная работа № 5 Разделение ресурсов. Организация критической секции.

В работе необходимо написать 2 программы для реализации критической секции в соответствии с вариантом задания. Критическую секцию первой программы реализовать на семафорах, а вторую с применением мониторов. Реализовать вывод на экран информации о попытках зайти в критическую секцию, успешном заходе и выходе из критической секции.

Варианты в словесном описании программы, для которой необходимо реализовать много поточное приложение. Каждый поток реализует одну сущность из предложенного варианта. Все задания можно свести к классическим задачам синхронизации. Варианты:

1. Жил-был один человек, и звали его Полуэкт. И было у него две бабушки, одна мама и не менее одной девушки. Человек этот очень любил работать допоздна, а его бабушки, мама и девушки очень волновались, не случилось ли с ним чего-нибудь. И вот каждый вечер все эти мамы, бабушки и девушки начинают друг другу звонить, чтобы узнать, не случилось ли чего-нибудь с Полуэктом. Человек и каждая из мам, бабушек и девушек имеет один телефон и может говорить или ждать звонка (ждать они не любят, а вот говорить очень даже любят). Соединение Полуэкт->(бабушка | мама | девушка) может быть установлено только по звонку Полуэкта. Бабушки прекращают звонить, если они уже переговорили со всеми остальными ожидающими и получили подтверждение хотя бы от одного из них. Полуэкт прекращает звонить после первого успешного соединения. Задача: разработать систему, позволяющую всем бабушкам и т.

п. получить подтверждение того, что Полуэкт ещё на работе и что-нибудь с ним ещё не случилось. Информацию может сообщить как сам Полуэкт, так и кто-нибудь из бабушек.

2. Проблема составителей букетов. Три составителя букетов представлены процессами S1, S2 и S3. Три поставщика представлены процессами V1, V2 и V3. Каждому составителю букетов необходимы розы, фиалки и пионы. Когда эти ресурсы у него есть, он некоторое время тратит на составление букета и переходит в состояние готовности составить букет снова. У S1 есть пионы в неограниченном количестве, у S2 есть фиалки в неограниченном количестве, у S3 есть розы в неограниченном количестве. V1 поставляет розы и фиалки, V2 поставляет пионы и розы, V3 поставляет пионы и фиалки. Для V1, V2 и V3 обеспечивается взаимное исключение. Следующий поставщик не может функционировать, пока ресурсы предыдущего поставщика не будут потреблены составителем букетов.

3. В общежитии института имеется совместная ванная комната. Если в ванной комнате есть женщина, то другая женщина может туда войти, а мужчина не может, и наоборот. На ванной есть индикатор, показывающий, в каком из трех состояний находится ванная: 1) никого нет; 2) в ванной женщины; 3) в ванной мужчины. Напишите систему, контролирующую доступ в ванную.

4. Студент, специализирующийся на антропологии, решил использовать свои знания в области системного программирования для обучения африканских бабуинов. Бабуины движутся через глубокий каньон по канату в двух направлениях: западном и восточном. Одновременно в одном направлении может двигаться произвольное количество бабуинов, но если бабуины движутся по канату навстречу, то возникает тупик. Разработайте два варианта контролирующей системы: **предпочтение отдается западным бабуинам**.

5. Студент, специализирующийся на антропологии, решил использовать свои знания в области системного программирования для обучения африканских бабуинов. Бабуины движутся через глубокий каньон по канату в двух направлениях: западном и восточном. Одновременно в одном направлении может двигаться произвольное количество бабуинов, но если бабуины движутся по канату навстречу, то возникает тупик. Разработайте два варианта контролирующей системы: **все бабуины равноправны**.

6. В доме культуры организована продажа билетов на концерты через Internet. Разработайте систему, которая позволит обрабатывать запросы на

бронирование мест и запросы о наличии свободных мест. Одновременно может работать до 10 процессов, получающих справки о свободных местах. Процесс, бронирующий места, может работать только, когда никто другой не бронирует места и не получает справку.

7. Разработайте систему, управляющую игровым автоматом типа «однорукий бандит». Один контролирующий процесс отвечает за прием и выдачу денег, а также за запуск трех игровых процессов. Каждый игровой процесс «крутит» соответствующее колесо.

8. Разработайте систему, управляющую пассажирским лифтом, с одной кнопкой вызова. Реализуйте схему работы лифта при перевозке пассажиров: если есть вызовы, то лифт останавливается при движении вниз.

9. Разработайте систему, управляющую пассажирским лифтом, с кнопками вызова «вверх» и «вниз». Реализуйте схему работы лифта: **лифт реагирует на вызовы в порядке их поступления**.

10. Разработайте систему, управляющую пассажирским лифтом, с кнопками вызова «вверх» и «вниз». Реализуйте схему работы лифта: **лифт реагирует на все вызовы в направлении своего движения. Если в текущем направлении вызовов нет, то лифт меняет направление движения**.