**Лабораторная работа 1**

**СИСТЕМА КОМАНД, ФАЙЛОВАЯ СТРУКТУРА, РАБОТА С ФАЙЛАМИ И УПРАВЛЕНИЕ ОС UNIX/LINUX С ПОМОЩЬЮ ИНТЕРПРЕТАТОРА BASH**

Цель работы: – изучение команд ОС для работы с файлами, каталогами, дисками, системной датой и временем; текстового редактора Kate и файлового менеджера Midnight Commander, исследовать основные объекты, команды, типы данных и операторы управления интерпретатора BASH; создать скрипт-файл.

Во всех заданиях должен быть контроль ошибок (если к какому-либо каталогу нет доступа, необходимо вывести соответствующее сообщение и продолжить выполнение).

Вывод сообщений об ошибках должен производиться в стандартный поток вывода сообщений об ошибках (stderr) в следующем виде:

имя\_модуля: текст\_сообщения.

Пример:

./1.exe : Error open file: 1.txt

Имя модуля, имя файла берутся из аргументов командной строки.

Для всех заданий формат вывода результатов:

На консоль выводится полный путь к файлу имя файла и его размер далее дополнительные сведения по индивидуальному заданию(права доступа и т.д.). Пример:

…

/home/stud210/1.txt 10298 …

/home/stud210/3.txt 1237 …

…

где после имени файла идет размер в байтах, далее см. по инд. заданию

В сообщении об ошибке выводится полный\_путь\_к\_файлу\_имя\_файла по формату:

/home/stud210/3.txt Error access denied…

0. Написать скрипт поиска файлов заданного размера в заданном каталоге (имя каталога задаётся пользователем в качестве третьего аргумента командной строки). Диапазон (мин.- мах.) размеров файлов задаётся пользователем в качестве первого и второго аргумента командной строки. На консоль выводится полный путь к файлу и его размер. Проверить работу программы для каталога /usr и диапазона (мин.- мах.) 1000 1010.

**Лабораторная работа 2**

**РАБОТА С КАТАЛОГАМИ, ОСНОВНЫЕ ПРИНЦИПЫ ПРОГРАММИРОВАНИЯ ПРОЦЕССОВ И ПОТОКОВ В ОС UNIX/LINUX**

Цель работы – изучение файловой системы ОС Linux и основных функций для работы с каталогами и файлами, исследовать методы создания процессов в ОС Linux, основные функции создания и управления процессами, обмен данными между процессами.

Во всех заданиях должен быть контроль ошибок (если к какому-либо каталогу нет доступа, необходимо вывести соответствующее сообщение и продолжить выполнение).

Вывод сообщений об ошибках должен производиться в стандартный поток вывода сообщений об ошибках (stderr) в следующем виде:

имя\_модуля: текст\_сообщения.

Пример:

./1.exe : Error open file: 1.txt

Имя модуля, имя файла берутся из аргументов командной строки.

0. Отсортировать в заданном каталоге (аргумент 1 командной строки) и во всех его подкаталогах файлы по следующим критериям (аргумент 2 командной строки, задаётся в виде целого числа):1 – по размеру файла, 2 – по имени файла. Записать отсортированные файлы в новый каталог (аргумент 3 командной строки). Процедуры копирования должны запускаться в отдельном процессе для каждого копируемого файла с использованием функций read() и write(). Каждый процесс выводит на экран свой pid, полный путь, имя копируемого файла и число скопированных байт (все в одной строке). Число запущенных процессов в любой момент времени не должно превышать N (вводится пользователем). Проверить работу программы для каталога /usr/include N=6..

**Лабораторная работа 3**

**СРЕДСТВА МЕЖПРОЦЕССНОГО ВЗАИМОДЕЙСТВИЯ (СИГНАЛЫ)**

Цель работы – изучить методы и средства взаимодействия процессов с использованием сигналов в ОС Linux.

Создать дерево процессов согласно варианта индивидуального задания.

Процессы непрерывно обмениваются сигналами согласно табл. 2 . Запись в таблице 1 вида: 1->(2,3,4,5) означает, что исходный процесс 0 создаёт дочерний процесс 1, который, в свою очередь, создаёт дочерние процессы 2,3,4,5. Запись в таблице 2 вида: 1->(2,3,4) SIGUSR1 означает, что процесс 1 посылает дочерним процессам 2,3,4 одновременно (т.е. за один вызов kill() ) сигнал SIGUSR1.Каждый процесс при получении или посылке сигнала выводит на консоль информацию в следующем виде:

N pid ppid послал/получил USR1/USR2 текущее время (мксек)

где N-номер сына по табл. 1

После передачи 101–го по счету сигнала SIGUSR родительский процесс посылает сыновьям сигнал SIGTERM и ожидает завершения всех сыновей, после чего завершается. Сыновья, получив сигнал SIGTERM, завершают работу с выводом на консоль сообщения вида:

pid ppid завершил работу после X-го сигнала SIGUSR1 и Y-го сигнала SIGUSR2

где X,Y – количество посланных за все время работы данным сыном сигналов SIGUSR1 и SIGUSR2

Для создания правильной последовательности сигналов в соответствие с таблицей задания необходимо для каждого процесса написать свой обработчик сигналов в котором он (процесс) принимает сигнал от предыдущего (в таблице) процесса и посылает следующему (в таблице) процессу!

Во всех заданиях должен быть контроль ошибок (если к какому-либо каталогу нет доступа, необходимо вывести соответствующее сообщение и продолжить выполнение).

Вывод сообщений об ошибках должен производиться в стандартный поток вывода сообщений об ошибках (stderr) в следующем виде:

имя\_модуля: текст\_сообщения.

Пример: ./1.exe : Error open file: 1.txt

Имя модуля, имя файла берутся из аргументов командной строки.

Варианты индивидуальных заданий в табл.1, табл.2.

**Дерево процессов**

Таблица 1

|  |  |
| --- | --- |
| **№** | **Дерево процессов** |
| 0 | 1->2 2->(3,4) 4->5 3->6 6->7 7->8 |

**Последовательность обмена сигналами**

Таблица 2

|  |  |
| --- | --- |
| **№** | **Последовательность обмена сигналами** |
| 0 | 1->2 SIGUSR1 2->(3,4) SIGUSR2 4->5 SIGUSR1  3->6 SIGUSR1 6->7 SIGUSR1 7->8 SIGUSR2 8->1 SIGUSR2 |

**Лабораторная работа 4**

**СРЕДСТВА МЕЖПРОЦЕССНОГО ВЗАИМОДЕЙСТВИЯ (КАНАЛЫ, РАЗДЕЛЯЕМАЯ ПАМЯТЬ, СЕМАФОРЫ)**

Цель работы – изучить методы и средства взаимодействия процессов с использованием каналов, разделяемой памяти и семафоров в ОС Linux.

Во всех заданиях должен быть контроль ошибок (если к какому-либо каталогу нет доступа, необходимо вывести соответствующее сообщение и продолжить выполнение).

Вывод сообщений об ошибках должен производиться в стандартный поток вывода сообщений об ошибках (stderr) в следующем виде:

имя\_модуля: текст\_сообщения.

Пример:

./1.exe : Error open file: 1.txt

Имя модуля, имя файла берутся из аргументов командной строки.

0. Cоздать два дочерних процесса. Родительский процесс создаёт семафор (сем1) и 2 неименованных канала (кан1 и кан2). Оба дочерних процесса непрерывно записывают в каналы по 110 строк вида: номер\_строки pid\_процесса текущее\_время (мксек). Родительский процесс читает из каждого канала по 75 строк и выводит их на экран. Всего дочерние процессы должны записать по 1010 строк. Семафор (сем1) используется процессами для разрешения кому из процессов получить доступ к каналу. Дочерние процессы начинают работу после получения сигнала SIGUSR2 от родительского процесса. По завершению работы они посылают сигнал SIGUSR1 родительскому процессу.