Отчёт о выполнении лабораторной работы №13

Российский Университет Дружбы Народов
Факультет Физико-Математических и Естественных Наук

Дисциплина: *Операционные системы*

Работу выполняла: *Арежина Адриана*

№ ст. билета: *1032201674*

Группа: *НКНбд-01-20*

Москва. 2021г.

Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Задание

1. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени *t1* дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени *t2<>t1*, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (*> /dev/tty#*, где *#* — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не в фоновом, а в привилегированном режиме. Доработать программу так, чтобы имелась возможность взаимодействия трёх и более процессов.
2. Реализовать команду *man* с помощью командного файла. Изучите содержимое каталога */usr/share/man/man1*. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой *less* сразу же просмотрев содержимое справки. Командный файл должен получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде или сообщение об отсутствии справки, если соответствующего файла нет в каталоге *man1*.
3. Используя встроенную переменную *$RANDOM*, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что *$RANDOM* выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

Выполнение работы

1. Написала командный файл, реализующий упрощённый механизм семафоров. Командный файл в течение некоторого времени *t1* дожидается освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использует его в течение некоторого времени *t2<>t1*, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустила командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (*> /dev/tty#*, где *#* — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не в фоновом, а в привилегированном режиме. Доработала программу так, чтобы имелась возможность взаимодействия трёх и более процессов. (см. рисунки файл 1, файл 1.1, результат 1)

2. Реализовала команду *man* с помощью командного файла. Изучила содержимое каталога */usr/share/man/man1*. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой *less* сразу же просмотрев содержимое справки. Командный файл получает в виде аргумента командной строки название команды и в виде результата выдаёт справку об этой команде или сообщение об отсутствии справки, если соответствующего файла нет в каталоге *man1*. (см. рисунки файл 2, запуск, результат 2, less, результат less, ls, результат ls)

3. Используя встроенную переменную *$RANDOM*, написала командный файл, генерирующий случайную последовательность букв латинского алфавита. Учла, что *$RANDOM* выдаёт псевдослучайные числа в диапазоне от 0 до 32767. (см. рисунки ниже файл 3, [результат 3]https://github.com/Adriana-Arezhina/Lab/blob/main/Lab13/pict/3.2.JPG))

Контрольные вопросы

1. Нужно взять в кавычки *«$1»*.
2. Написать переменные одну за другой. Например: *A = "BC"*

Либо с помощью оператора *+=*

B += C

3. Эта утилита выводит последовательность целых чисел с заданным шагом. Также можно реализовать с помощью утилиты *jot*.
4. 3
5. В *zsh* можно настроить отдельные сочетания клавиш так, как вам нравится. Использование истории команд в *zsh* ничем особенным не отличается от *bash*. *Zsh* очень удобен для повседневной работы и делает добрую половину рутины за вас. Но стоит обратить внимание на различия между этими двумя оболочками. Например, в *zsh* после *for* обязательно вставлять пробел, нумерация массивов в *zsh* начинается с 1, чего совершенно невозможно понять. Так, если вы используете *shell* для повседневной работы, исключающей написание скриптов, используйте *zsh*. Если вам часто приходится писать свои скрипты, только *bash*! Впрочем, можно комбинировать. Как установить *zsh* в качестве оболочки по умолчанию для отдельного пользователя: о.
6. Синтаксис верен.
7. Преимущества:

- По сравнению с *cmd* у *bash* больше возможностей. - По сравнению с нескриптовыми языками программирования у него более низкий порог вхождения. - Его не нужно отдельно устанавливать, он встроен в операционную систему.

Недостатки: - В интернете меньше дополнительной информации про него, чем про языки программирования. - Сложнее отлаживать программу.

Вывод

Я изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.