Лабораторная работа 12

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Арина Олеговна Аристова

Содержание

[Цель работы 1](#_Toc104602324)

[Задание 1](#_Toc104602325)

[Теоретическое введение 2](#_Toc104602326)

[Выполнение лабораторной работы 2](#_Toc104602327)

[Вывод 5](#_Toc104602328)

[Контрольные вопросы 5](#_Toc104602329)

# 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, в котором также запущен этот файл, но не 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.

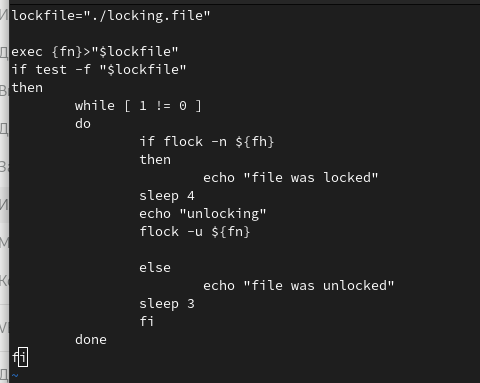
# Теоретическое введение

Преимущества и недостатки Bash:

Многие языки программирования намного удобнее и понятнее для пользователя. Например, Python более быстр, так как компилируется байтами. Однако главное преимущество Bash – его повсеместное распространение. Более того, Bash позволяет очень легко работать с файловой системой без лишних конструкций (в отличие от других языков программирования). Но относительно таких bash очень сжат. То есть, например, C имеет гораздо более широкие возможности для разработчика.

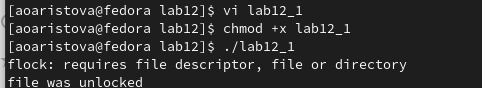
# Выполнение лабораторной работы

1. Я написала командный файл, реализующий упрощенный механизм семафоров.



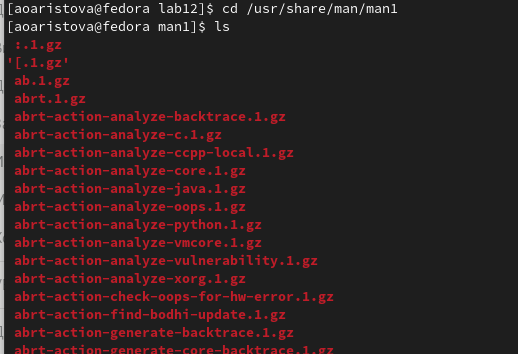
Скрипт к заданию 1.

Затем я добавила право на исполнение файла и выполнила его.



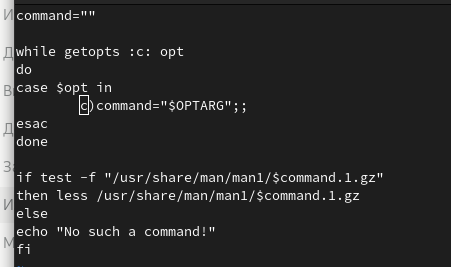
Результат выполнения скрипта 1.

1. Я просмотрела содержимое каталога /usr/share/man/man1.



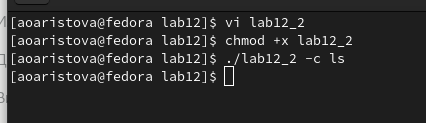
Просмотр каталога /usr/share/man/man1.

Я написала командный файл, позволяющий реализовать команду man с помощью команды less, которая выдает содержимое справки по команде.



Скрипт к заданию 2.

Затем я добавила право на исполнение файла и выполнила его.



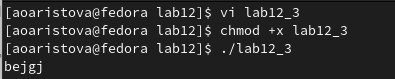
Результат выполнения скрипта 2.

1. Я написала командный файл, который генерировал случайную последовательность букв латинского алфавита, для этого я использовала встроенную переменную $RANDOM.

Скрипт к заданию 3.

Скрипт к заданию 3.

Затем я добавила право на исполнение файла и выполнила его.



Результат выполнения скрипта 3.

# Вывод

В ходе выполнения лабораторной работы я изучила основы программирования в оболочке ОС UNIX, а также научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# Контрольные вопросы

1. Найдите синтаксическую ошибку в следующей строке: 1 while [$1 != “exit”]

Между скобками должны быть пробелы, иначе символы в скобках и сами скобки буду восприняты как один элемент.

1. Как объединить (конкатенация) несколько строк в одну?

cat file.txt | xargs

1. Найдите информацию об утилите seq. Какими иными способами можно реализовать её функционал при программировании на bash?

Команда seq выводит последовательность целых или действительных чисел, подходящую для передачи в другие программы. Реализовать ее функционал можно командой for n in {1..5} do done

1. Какой результат даст вычисление выражения $((10/3))? Вычисление этого выражения даст результат 3
2. Укажите кратко основные отличия командной оболочки zsh от bash.

Zsh очень сильно упрощает работу. Но существуют различия. Например, в zsh после for обязательно вставлять пробел, нумерация массивов в zsh начинается с 1. Если вы собираетесь писать скрипт, который будет запускать множество разработчиков, то рекомендуется Bash. Если скрипты вам не нужны - Zsh.

1. Проверьте, верен ли синтаксис данной конструкции 1 for ((a=1; a <= LIMIT; a++))

Да, этот синтаксис верен.

1. Сравните язык bash с какими-либо языками программирования. Какие преимущества у bash по сравнению с ними? Какие недостатки?

Многие языки программирования намного удобнее и понятнее для пользователя. Например, Python более быстр, так как компилируется байтами. Однако главное преимущество Bash – его повсеместное распространение. Более того, Bash позволяет очень легко работать с файловой системой без лишних конструкций (в отличие от других языков программирования). Но относительно таких bash очень сжат. То есть, например, C имеет гораздо более широкие возможности для разработчика.