Операционные системы

Лабораторная работа №13

Крупенникова Виктория Александровна

Содержание

# Цель работы

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

# Задание

1. Написать командный файл, реализующий упрощённый механизм семафоров.
2. Реализовать командуmanс помощью командного файла.
3. Используя встроенную переменную “$RANDOM”, напишите командный файл, гене-рирующий случайную последовательность букв латинского алфавита.

# Выполнение лабораторной работы

1. Написала 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). Для данной задачи я создала файл: sem.sh и написала соответствующий скрипт.

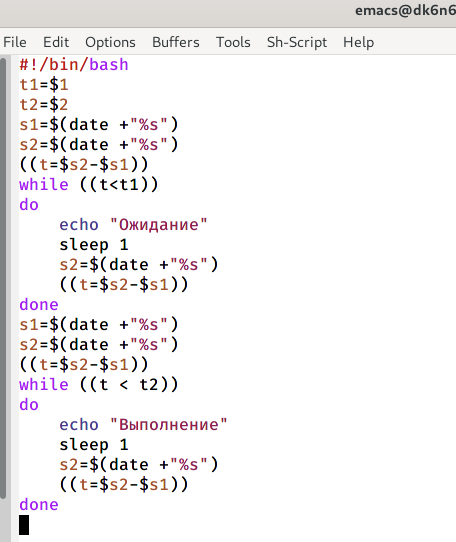


Figure 1: Скрипт

Далее я проверила работу написанного скрипта (./sem.sh 4 7), добавив право на исполнение файла (chmod +x .sh). Скрипт работает корректно.

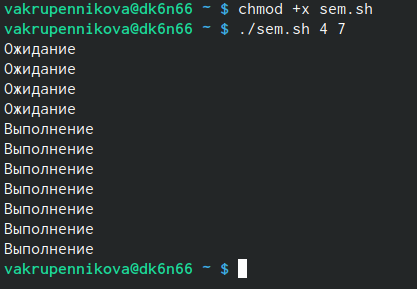


Figure 2: Проверка скрипта

После этого я изменила скрипт так,чтобы его можно было выполнять в нескольких терминалах и проверила его работу (например, команда «./sem.sh2 3 Ожидание > /dev/pts/1 &»)

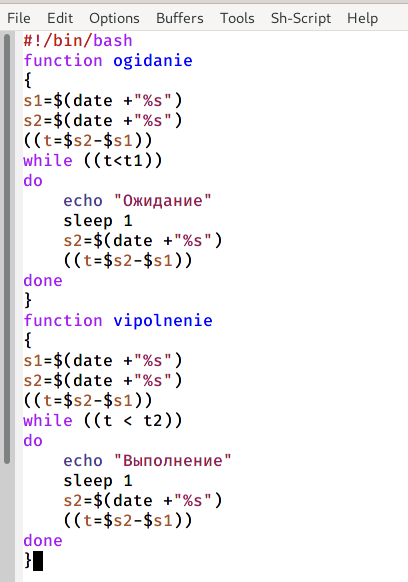


Figure 3: Изменение скрипта

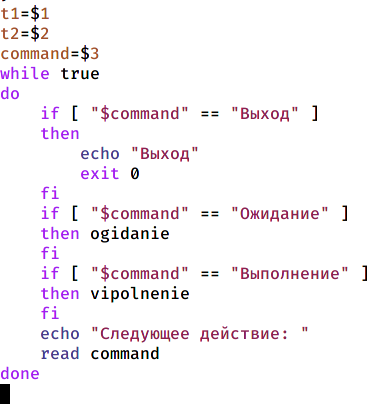


Figure 4: Изменение скрипта

Но ни одна команда не работала, так как мне “Отказано в доступе”. При этом скрипт работает корректно (команда «./sem.sh »).

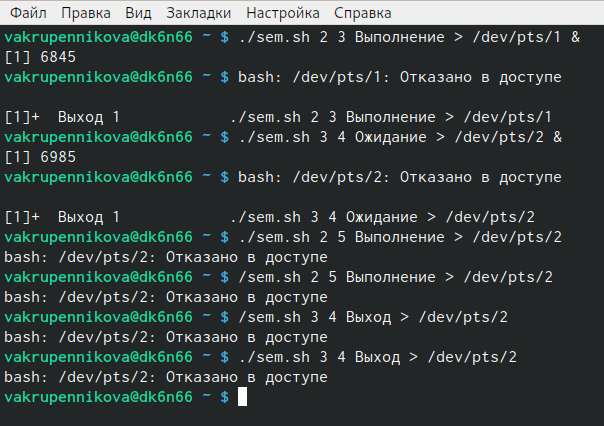


Figure 5: Проверка скрипта

1. Реализовала команду man с помощью командного файла. Изучиласодержимое 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.

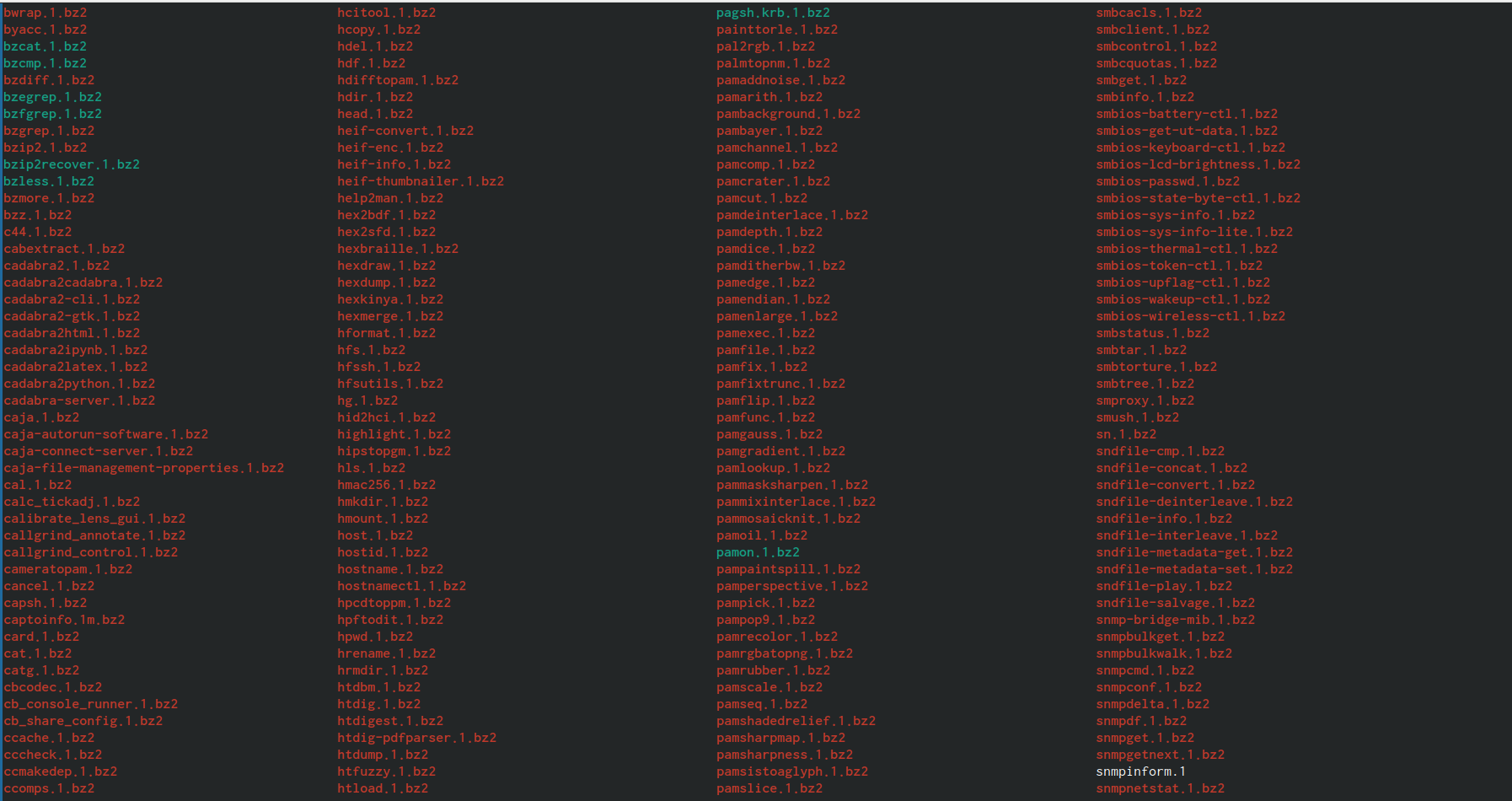


Figure 6: Реализация команды

Для данной задачи я создала файл : man.sh и написала соответствующий скрипт.

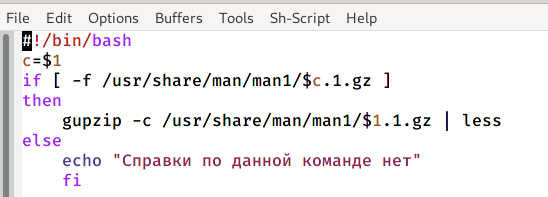


Figure 7: Написание скрипта

Далее я проверила работу написанного скрипта (команды «./man.sh ls», «./man.sh mkdir»), предварительно добавив право на исполнение файла (команда «chmod +x man.sh»). Cкрипт сработал и вывел, что по данным командам справок нет. Скрипт работает корректно.

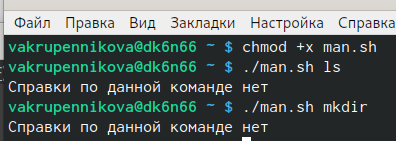


Figure 8: Проверка скрипта

1. Используя встроенную переменную $RANDOM, написала командный файл, генерирующий случайную последовательность букв латинского алфавита. Для данной задачи я создала файл: random.sh и написала соответствующий скрипт.

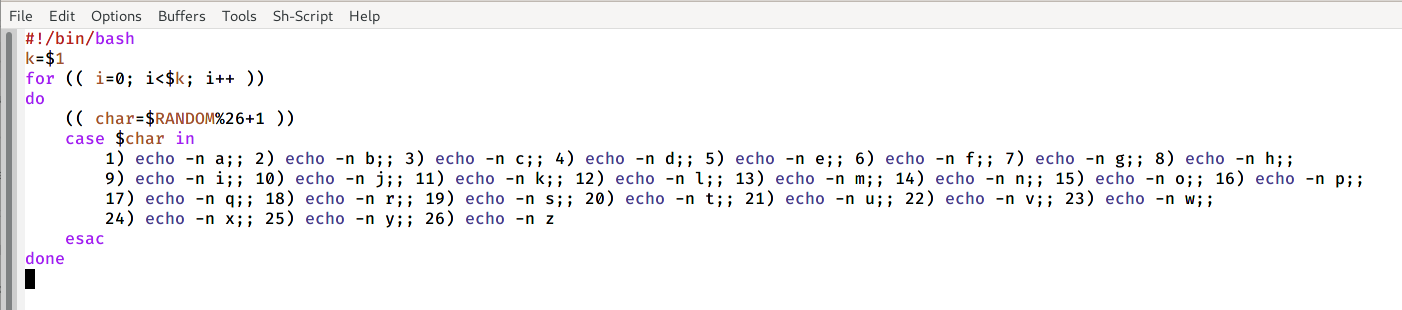


Figure 9: Написание скрипта

Далее я проверила работу написанного скрипта (команды «./random.sh 7» и «./random.sh 15»), предварительно добавив право на исполнение файла (команда «chmod +x random.sh»). Скрипт работает корректно.

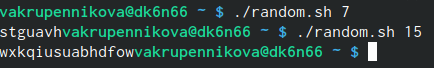


Figure 10: Проверка скрипта

# Контрольные вопросы:

1. while [$1 != “exit”]

В данной строчке допущены следующие ошибки:

* не хватает пробелов после первой скобки
* выражение $1 необходимо взять в “”, потому что эта переменная может содержатьпробелыТаким образом, правильный вариант должен выглядеть так: while [“$1”!= “exit”]

1. Чтобы объединить несколько строк в одну, можно воспользоваться несколькими способами:

* Первый:VAR1=“Hello,”VAR2=" World“VAR3=”VAR2“echo”$VAR3"Результат: Hello, World
* Второй:VAR1=“Hello,”VAR1+=" World“echo”$VAR1"Результат: Hello, World

1. Команда seq в Linux используется для генерации чисел.

Параметры:

* seq LAST: если задан только один аргумент, он создает числа от 1 до LAST с шагом шага, равным 1. Если LAST меньше 1, значение is не выдает.
* seq FIRST LAST: когда заданы два аргумента, он генерирует числа от FIRST до LAST с шагом 1, равным 1. Если LAST меньше FIRST, он не выдает никаких выходных данных.
* seq FIRST INCREMENT LAST: когда заданы три аргумента, он генерирует числа от FIRST до LAST на шаге INCREMENT . Если LAST меньше, чем FIRST, он не производит вывод.
* seq -f «FORMAT» FIRST INCREMENT LAST: эта команда используется для генерации последовательности в форматированном виде. FIRST и INCREMENT являются необязательными.
* seq -s «STRING» ПЕРВЫЙ ВКЛЮЧЕНО: Эта команда используется для STRING для разделения чисел. По умолчанию это значение равно /n. FIRST и INCREMENT являются необязательными.
* seq -w FIRST INCREMENT LAST:эта команда используется для выравнивания ширины путем заполнения начальными нулями. FIRST и INCREMENT являются необязательными.

1. Результатом данного выражения $((10/3))будет 3, потому что это целочисленное деление без остатка.
2. Отличия командной оболочки zshот bash:

* В zsh более быстрое автодополнение для cdс помощью Тab
* В zsh существует калькулятор zcalc, способный выполнять вычисления внутри терминала
* В zsh поддерживаются структуры данных «хэш»
* В zsh поддерживается раскрытие полного пути на основе неполных данных
* В zsh поддерживается замена части пути
* В zsh есть возможность отображать разделенный экран, такой же как разделенный экран vim

1. for((a=1; a<= LIMIT; a++)) синтаксис данной конструкции верен, потому что, используя двойные круглые скобки, можно не писать $перед переменными ().
2. Преимущества скриптового языка bash:

* Один из самых распространенных и ставится по умолчанию в большинстве дистрибутивах Linux, MacOS
* Удобное перенаправление ввода/вывода
* Большое количество команд для работы с файловыми системами Linux, можно писать собственные скрипты, упрощающие работу в LinuxНедостатки скриптового языка bash: дополнительные библиотеки других языков позволяют выполнить больше действий
* Bash не является языков общего назначения
* Утилиты, при выполнении скрипта, запускают свои процессы, которые, в свою очередь, отражаются на быстроте выполнения этого скрипта
* Скрипты, написанные на bash, нельзя запустить на других операционных системах без дополнительных действий

# Выводы

В ходе выполнения данной 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.