Отчёт по лабораторной работе №13

дисциплина: Операционные системы

Быстров Глеб Андреевич

Содержание

1 Цель работы

В данной лабораторной работе мне будет необходимо изучить основы программирования в оболочке ОС UNIX/Linux. Будет необходимо научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

2 Теория

Bash — популярный командный интерпретатор, используемый в юниксоподобных системах, например, в GNU/Linux. Это программа, которую называют оболочка либо шелл (shell), а само название «bash» является сокращением от «Bourne Again Shell». Интерпретатор Bash принимает ваши команды, передавая их операционной системе. Чтобы осуществлялось взаимодействие с ОС, применяются терминалы (gnome-terminal, nxterm и прочие).

Библиографический список ссылка №1

Команда man позволяет получить доступ к общей базе справки по команде, функции или программе. Обычно для просмотра справки программе надо передать название команды или другого объекта в системе. Синтаксис у неё такой: $ man раздел название_страницы

Библиографический список ссылка №2

Команда less позволяет перематывать текст не только вперёд, но и назад, осуществлять поиск в обоих направлениях, переходить сразу в конец или в начало файла.

3 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 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.

4 Выполнение лабораторной работы

- ЗАДАНИЕ 1

1. С помощью команды emacs создал файл task1.sh. (рис. 4.1)

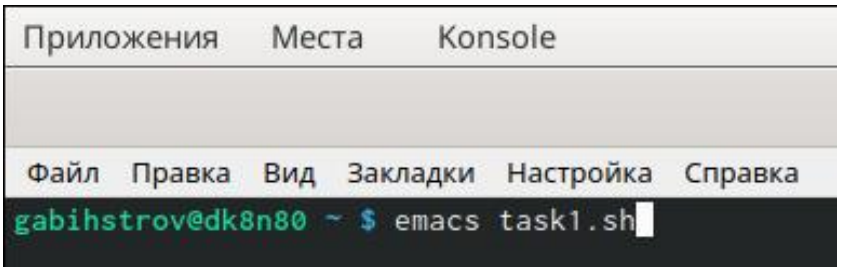
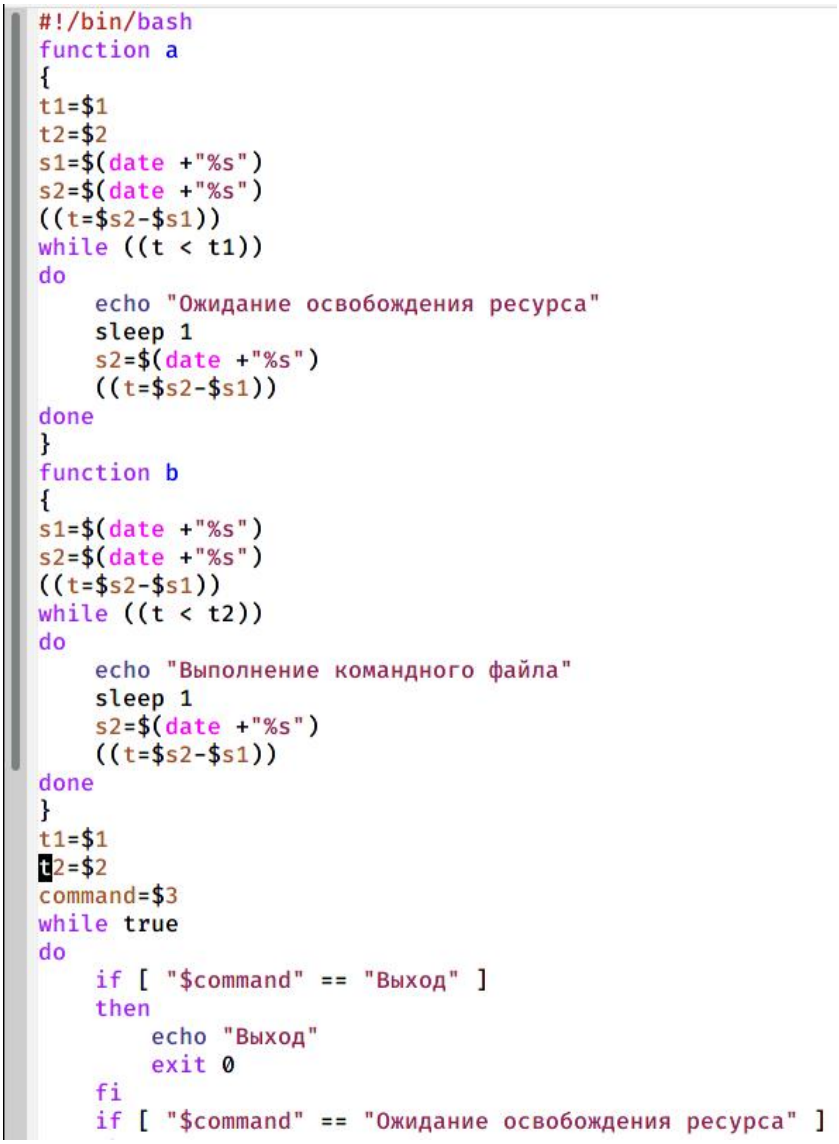


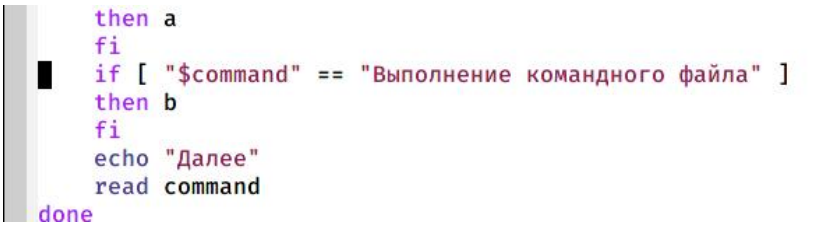
Figure 4.1: Создание файла

2. Написал 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). (рис. 4.2) (рис. 4.3)



```
#!/bin/bash
function a
{
t1=$1
t2=$2
s1=$(date +"%s")
s2=$(date +"%s")
((t=$s2-$s1))
while ((t < t1))
do
    echo "Ожидание освобождения ресурса"
    sleep 1
    s2=$(date +"%s")
    ((t=$s2-$s1))
done
}
function b
{
s1=$(date +"%s")
s2=$(date +"%s")
((t=$s2-$s1))
while ((t < t2))
do
    echo "Выполнение командного файла"
    sleep 1
    s2=$(date +"%s")
    ((t=$s2-$s1))
done
}
t1=$1
t2=$2
command=$3
while true
do
    if [ "$command" == "Выход" ]
    then
        echo "Выход"
        exit 0
    fi
    if [ "$command" == "Ожидание освобождения ресурса" ]
```

Figure 4.2: Скрипт командного файла



```
    then a
    fi
    if [ "$command" == "Выполнение командного файла" ]
    then b
    fi
    echo "Далее"
    read command
done
```

Figure 4.3: Скрипт командного файла

3. Вызвал командный файл. Код успешно справился с заданием. (рис. 4.4)

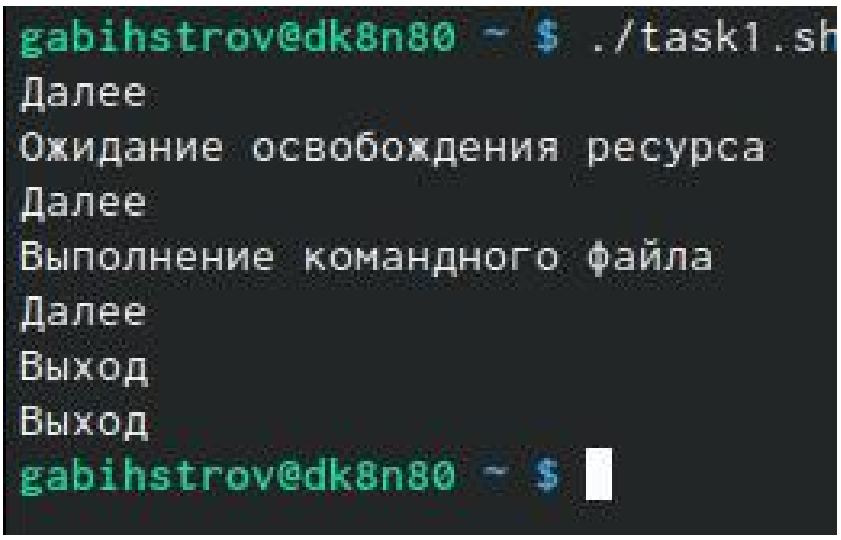


Figure 4.4: Демонстрация работы

- ЗАДАНИЕ 2

4. С помощью команды emacs создал файл task2.sh. (рис. 4.5)

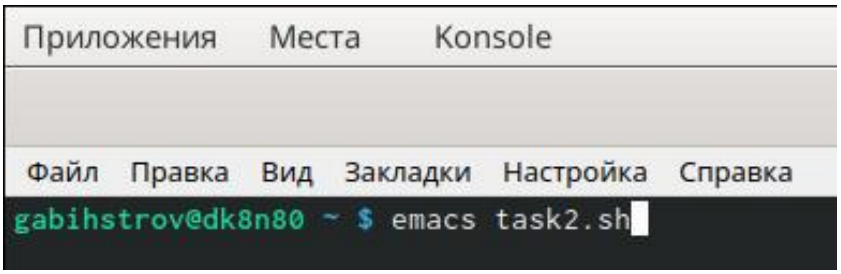


Figure 4.5: Создание файла

5. Реализовал команду man с помощью командного файла. Изучил 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. (рис. 4.6)

Figure 4.6: Скрипт командного файла

6. Вызвал командный файл. Запросил информацию о команде 7zr. (рис. 4.7)

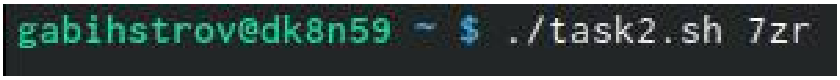


Figure 4.7: Вызов файла

7. Код успешно справился с заданием и вывел справку по введённой команде. (рис. 4.8)

Figure 4.8: Демонстрация работы

- ЗАДАНИЕ 3

7. С помощью команды emacs создал файл task3.sh. (рис. 4.9)

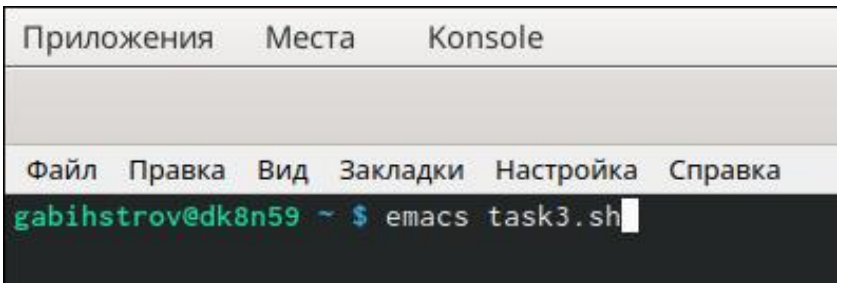


Figure 4.9: Создание файла

8. Используя встроенную переменную $RANDOM, написал командный файл, генерирующий случайную последовательность букв латинского алфавита. Учёл, что $RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767. (рис. 4.10)

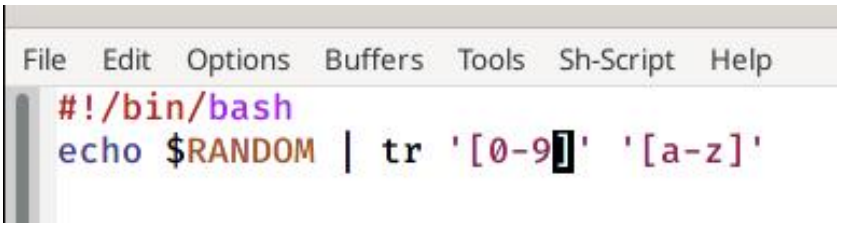


Figure 4.10: Скрипт командного файла

9. Вызвал командный файл. Код успешно справился с заданием. (рис. 4.11)

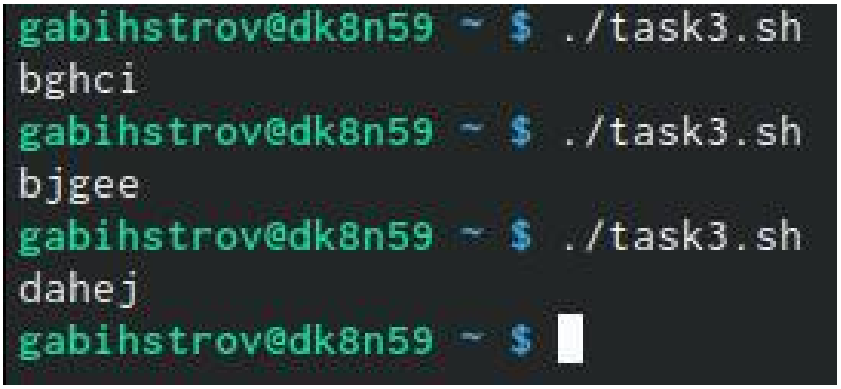


Figure 4.11: Демонстрация работы

5 Контрольные вопросы

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

$1 должен быть в ””, так как переменная может иметь пробелы.

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

Библиографический список ссылка №3

- 1 способ

Самый простой способ объединить две или более строковые переменные – записать их одну за другой:
VAR1=“Hello,” VAR2=” World” VAR3=“$VAR1VAR2” echo “$VAR3”
Hello, World

- 2 способ

Вы также можете объединить одну или несколько переменных с литеральными строками:
VAR1=“Hello,” VAR2=“$VAR1World"echo"VAR2”
Hello, World

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

Библиографический список ссылка №4

Команда seq в Linux используется для генерации чисел от ПЕРВОГО до ПОСЛЕДНЕГО шага INCREMENT. Это очень полезная команда, в которой нам пришлось генерировать список чисел в цикле while, for, before.

Синтаксис: seq [OPTION]… LAST or seq [OPTION]… FIRST LAST or seq [OPTION]… FIRST INCREMENT LAST

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.
- seq –help : отображает справочную информацию.
- seq –version : отображает информацию о версии.

4. Какой результат даст вычисление выражения $((10/3))?

Целочисленное деление без остатка выдаст число 3.

5. Укажите кратко основные отличия командной оболочки zsh от bash.

Библиографический список ссылка №5

Первое, на что мы взглянем (и это один из наиболее значительных аспектов, на мой взгляд) — это популярность оболочки. Хотя у Z Shell имеется ряд пользователей в среде разработчиков, обычно безопаснее писать свои скрипты для Bash, поскольку гораздо больше людей способны запустить эти скрипты.

Важность всего этого заключается в адаптации скриптов для общедоступных репозиториев, а также в возможности написать грамотную документацию. Благодаря своему большому сообществу для Bash есть несколько крупных ресурсов, которые смогут помочь вам разобраться, как её использовать.

Так что, если вы собираетесь писать скрипт, который легко будет запускать множество разработчиков, то я рекомендую вам Bash. Хотя это не должно помешать вам использовать Z Shell там, где она более применима для ваших целей. Найти верное решение задачи гораздо важнее, чем взять то, что популярно, так что имейте это в виду.

Хотя Bash куда более распространён, это не означает, что у Z Shell нет полезных возможностей. Её часто хвалят за интерактивную работу, поскольку она лучше настраивается, чем Bash. Например, командная строка более гибкая. Можно отобразить команду слева, а другую справа, как в разделённом экране vim. Автодополнение также быстрее и более изменяемое, чем в Bash.

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

Синтаксис данной конструкции верен так как мы можем использовать двойные скобки и не писать $.

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

Преимущества:

- Встроенные сложные типы данных и операции над ними;
- Удобная работа со строками при помощи регулярных выражений;
- Возможность использовать совместно различные программы;
- Относительная простота изучения и использования.

Недостатки:

- Время исполнения скрипта;
- Проблемы безопасности;
- Скрипты на web страницах могут использовать уязвимости браузера и ОС;
- Результат выполнения скрипта зависит от версии интерпретатора.

6 Выводы

В данной лабораторной работе мне успешно удалось изучить основы программирования в оболочке ОС UNIX/Linux. Получилось научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

7 Библиографический список

1. Что такое man (https://losst.ru/chto-takoe-man)
2. Команда less в Linux (https://losst.ru/komanda-less-v-linux)
3. Конкатенации строк в Bash (https://andreyex.ru/operacionnaya-sistema-linux/konkatenatsii-strok-v-bash/)
4. Команда seq в Linux (http://espressocode.top/seq-command-in-linux-with-examples/)
5. Zsh vs Bash: особенности и различия (https://gitjournal.tech/zsh-vs-bash/#_Z_Shell_Bash)