**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

# **ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 13**

*дисциплина:*  *Операционные системы*

Студент:Урзов Александр Урзов

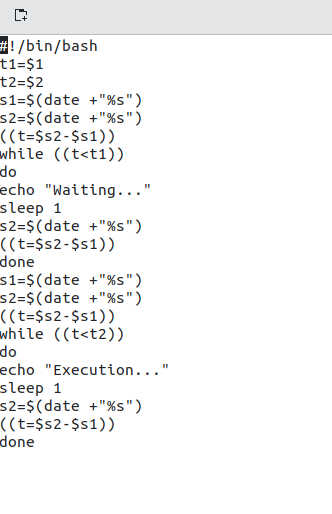
Группа: НПМбд-02-20

**Цель работы:** Изучить основы программирования в оболочке ОС UNIX . Научиться писать более сложные командные файлы с использованием логических управляющих инструкций и циклов.

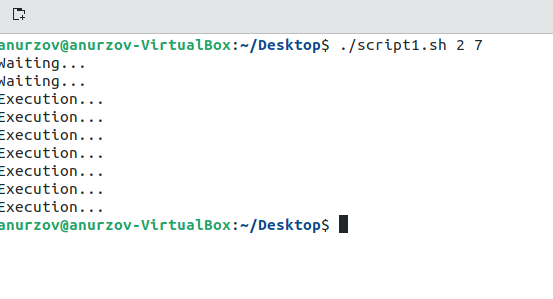
Ход работы:

1)Я написал командный файл реализующий упрощенный алгоритм семафоров. Командный файл должен в течении некоторго времени **t1** дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение времени **t2**,также выдавая информацию о том, что ресурс используется соответствующим командным файлом

Для данной задачи я создал файл : **script1.sh** и написал соответствующий скрипт. (см.Рис 1)

Рис.1

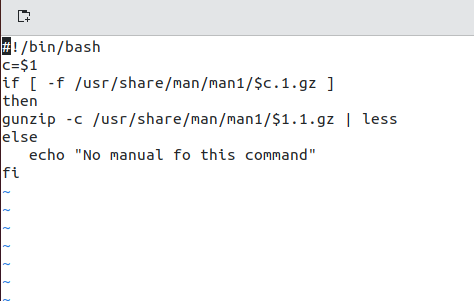
Далее я проверил работу написанного скрипта. Скрипт работает корректно. (см.Рис 2)

Рис.2

2)Я написал командный файл,являющийся аналогом команды **man** .Изучил содержимое каталога  **/usr/share/man/man1** .В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе команд и программ.Каждый файл можно открыть командой **less** и простмотреть содержимое справки.Командный файл получает на вход название команды и выдает справку об этой команде.Если команды нет в каталоге,то выводится ошибка.

Так выглядит скрипт:

(см.Рис 3)

Рис.3

Я проверил его работу на корректность. Работает корректно.

(см.Рис 4,5)

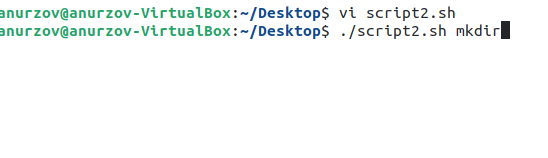
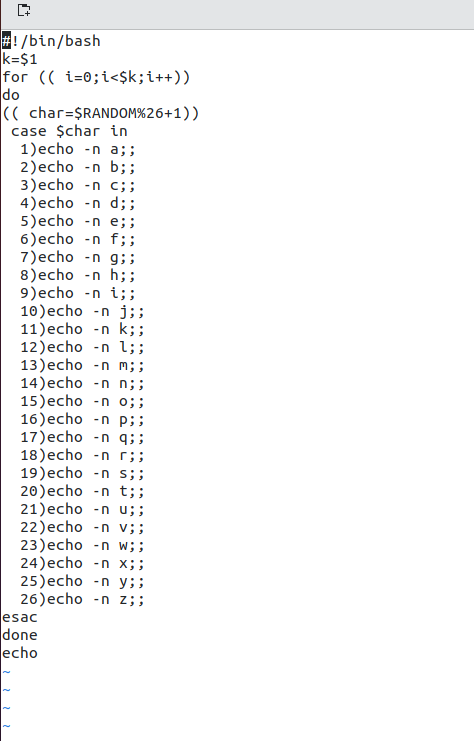
Рис.4

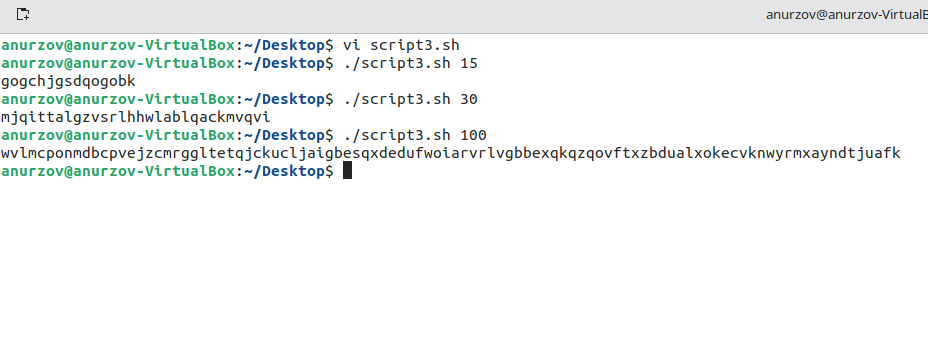
Рис.5

3)Я написал скрипт, который выводит случайную последовательность букв латинского алфавита. (см.Рис 6)

Рис.6

Далее я проверил скрипт на работоспособность. Работает корректно.

(см.Рис 7)

Рис.7

**Вывод: В ходе выполнения данной лабораторной работы я изучил основы программирования в оболочке ОС UNIX,а также научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.**

**Контрольные вопросы:**

1) while [$1 != "exit"] В данной строчке допущены следующие ошибки:

не хватает пробелов после первой скобки **[** и перед второй скобкой **]**

выражение $1 необходимо взять в **“”**, потому что эта переменная может содержать пробелы

Таким образом, правильный вариант должен выглядеть так:

**while [ “$1” != "exit" ]**

**2) Чтобы объединить несколько строк в одну, можно воспользоваться несколькими способами:**

**Первый:**

VAR1="Hello,"

VAR2=" World"

VAR3="$VAR1$VAR2"

echo "$VAR3"

Результат: Hello, World

**Второй:**

VAR1="Hello, "

VAR1+=" World"

echo "$VAR1"

Результат: Hello, World

**3) Команда seq в Linux используется для генерации чисел от ПЕРВОГО до ПОСЛЕДНЕГО шага INCREMENT.**

**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.

**4) Результатом данного выражения $((10/3)) будет 3, потому что это целочисленное деление без остатка.**

**5) Отличия командной оболочки zsh от bash:**

В zsh более быстрое автодополнение для cd с помощью Тab

В zsh существует калькулятор zcalc, способный выполнять вычисления внутри терминала

В zsh поддерживаются числа с плавающей запятой

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

**6) for ((a=1; a <= LIMIT; a++)) синтаксис данной конструкции верен, потому что, используя двойные круглые скобки, можно не писать $ перед переменными ().**

**7) 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 Linux

**Недостатки 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.