**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ  
Факультет физико-математических и естественных наук  
Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 13  
Программирование в командном процессоре ОС UNIX. Расширенное программирование**  
дисциплина: Операционные системы

Студент: Тазаева Анастасия Анатольевна

Группа: НПИбд-02-20

МОСКВА 2021г.

**Содержание:**

1. Цель работы
2. Задания
3. Ход работы
4. Контрольные вопросы(ответы)
5. Выводы

**Цель работы:**

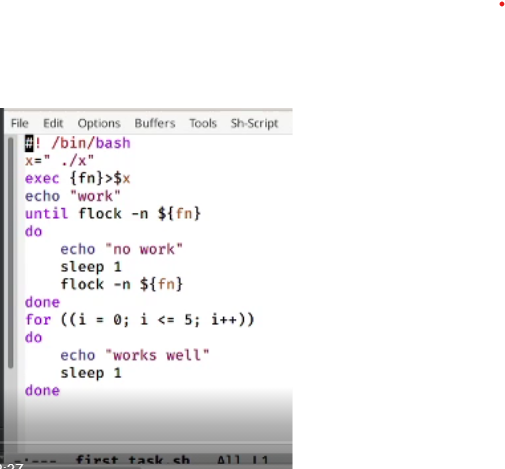
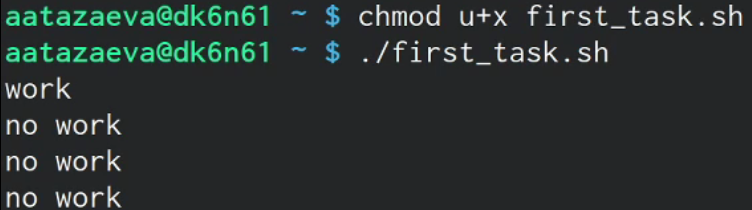
Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

**Задания:**

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.

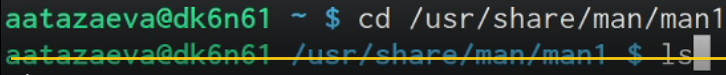
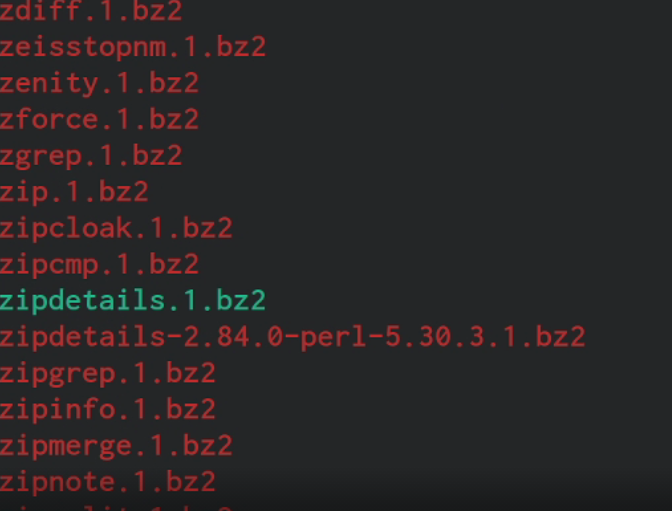
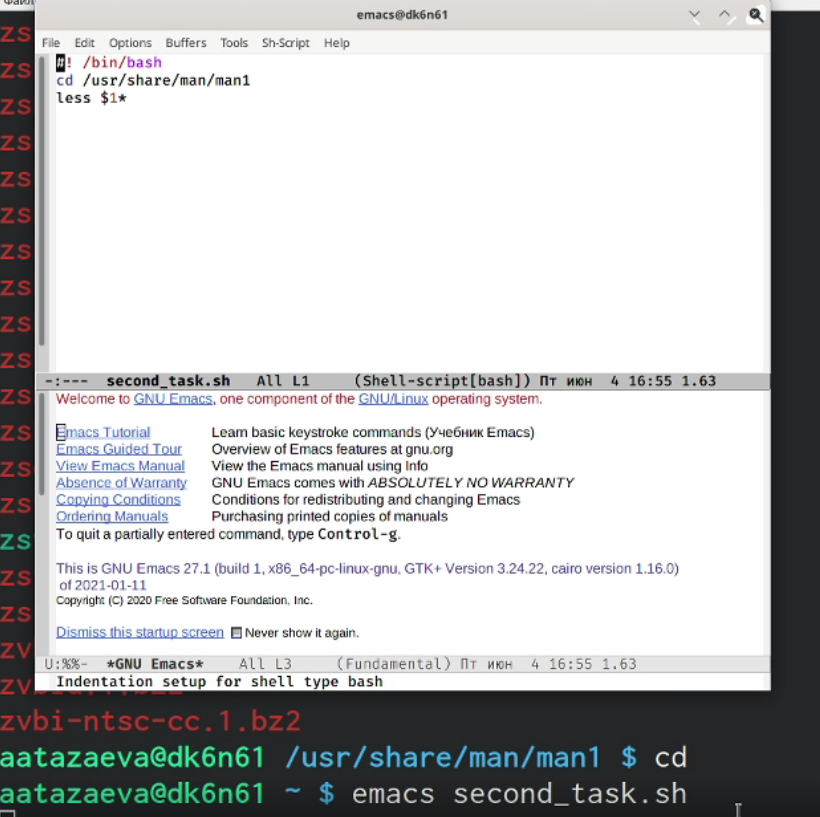
**Ход работы:**

1. С помощью команды emacs создала файл(рис.1), в котором и написала код, который реализует механизм семафоров(рис.2). Не забываем дать права на выполнение юзеру с помощью команды chmod u+x (файл)(рис.3)

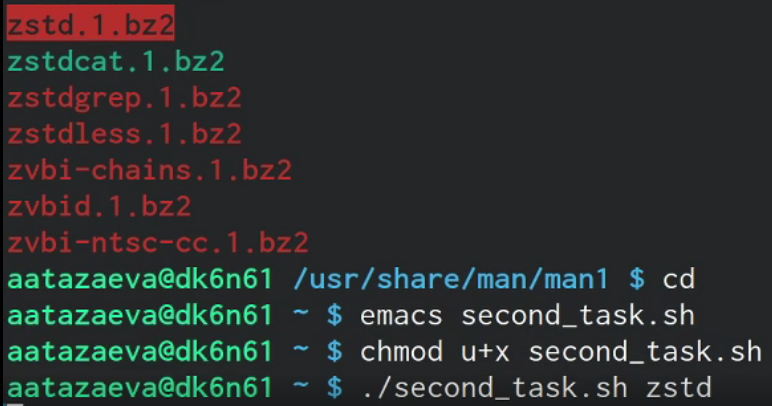
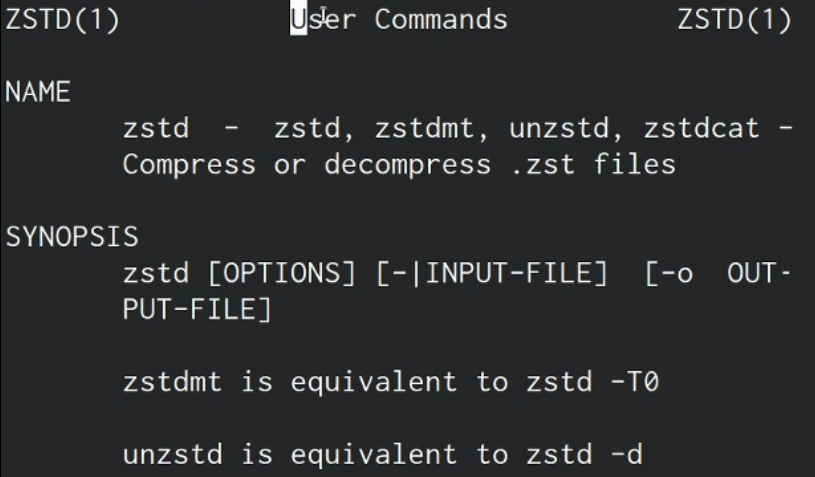
**  
(рис.1)  
  
(рис.2)  
  
(рис.3)**

Проверила работу(рис.3). Работает успешно.

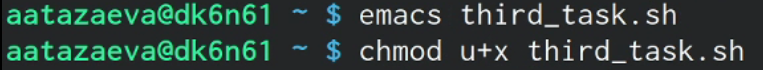
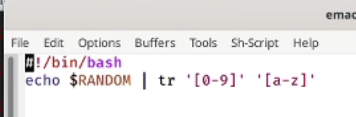
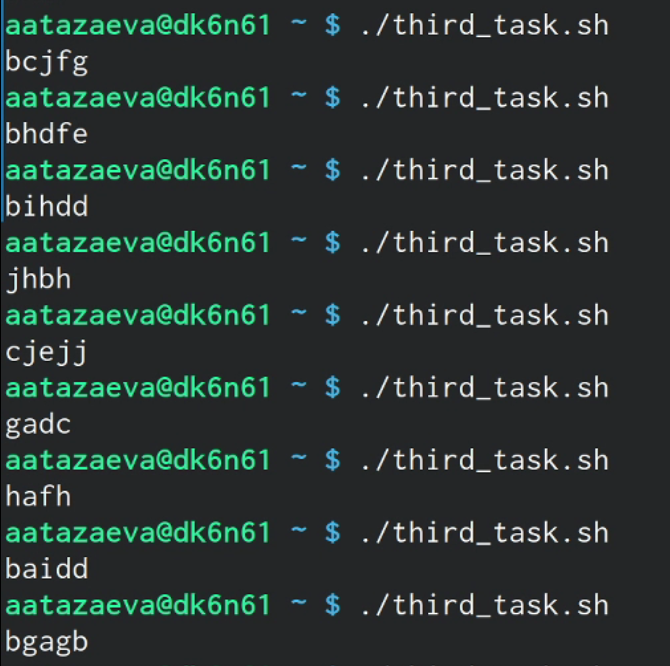
1. Далее реализовала команду man с помощью командного файла. Сначала нам необходимо было изучить содержимое каталога /usr/share/man/man1(рис.4-5). Заметила, что там все файлы имеют расширение 1.bz2. Cоздаем файл для второго задания (emacs second\_task.sh). Даем себе права на выполнение (chmod u+x srcond\_task.sh). Ну и сам командный файл выглядит так(рис.6-7)

**  
(рис.4)  
  
(рис.5)  
  
(рис.6)  
  
(рис.7)**

Проверила работу командного файла.(рис.8) Код справился с заданием и вывел нам информацию и zstd.(рис.9)

**  
(рис.8)  
  
(рис.9)**

1. Создаем командный файл, который будет выводить на экран рандомные буквы. Для этого создаем файл и даем ему права(рис.10):  
   emacs third\_task.sh  
   chmod u+x third\_task.sh

**  
(рис.10)  
  
(рис.11)  
  
(рис.12)**

Далее сам код и его проверка(рис.11 и 12).

**Контрольные вопросы:**

1. Найдите синтаксическую ошибку в следующей строке:

while [$1 != "exit"]

$1.

Так же между скобками должны быть пробелы. В противном случае скобки и рядом стоящие символы будут восприниматься как одно целое

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

cat file.txt | xargs | sed -e 's/. /.\n/g'

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

seq - выдает последовательность чисел.

Реализовать ее функционал можно командой

for n in {1..5}

do <КОМАНДА>

done

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

3

1. 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 (что не особо удобно на самом деле).

Если вы собираетесь писать скрипт, который легко будет запускать множество разработчиков, то я рекомендуется Bash. Если скрипты вам не нужны - Zsh (более простая работа с файлами, например)

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

Верен

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

Bash позволяет очень легко работать с файловой системой без лишних конструкций (в отличи от обычного языка программирования). Но относительно обычных языков программирования bash очень сжат. Тот же Си имеет гораздо более широкие возможности для разработчика.

ВЫВОД:

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