# Титульный лист

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

**Лабораторная работа 13**

По дисциплине "Операционные системы"

Выполнил:

Студент группы НПМбв-01-19

Студенческий билет №: 1032187017

Кушнирчук Дарья Вадимовна

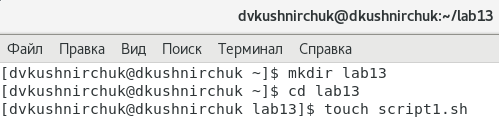
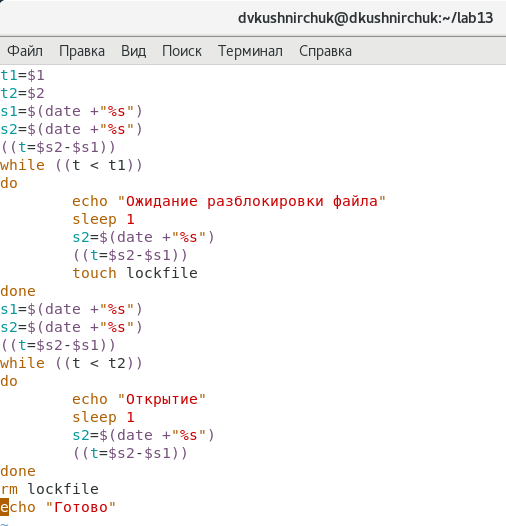
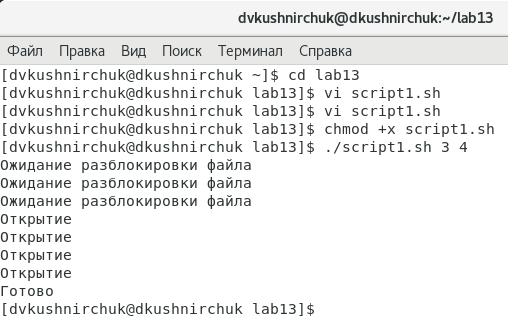
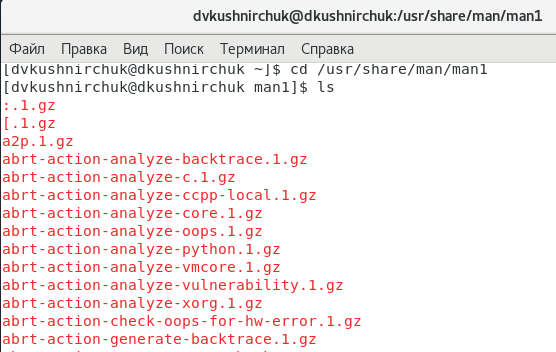
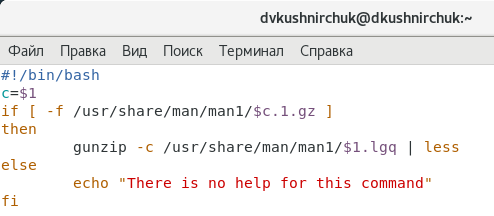
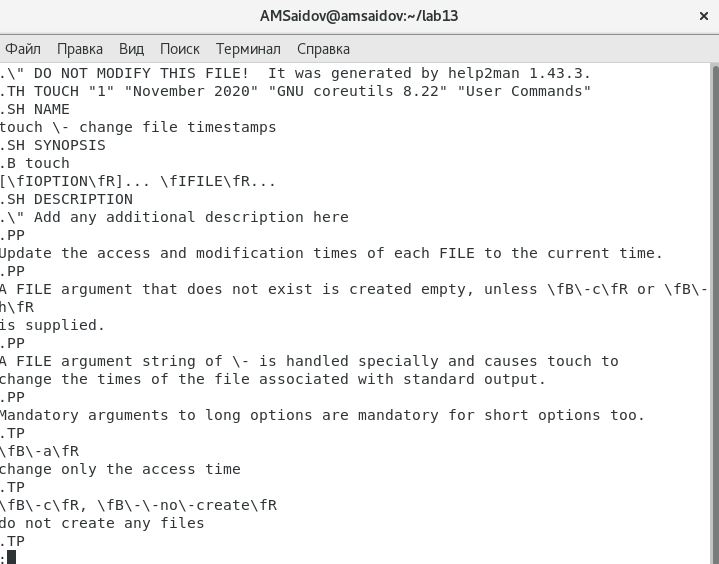
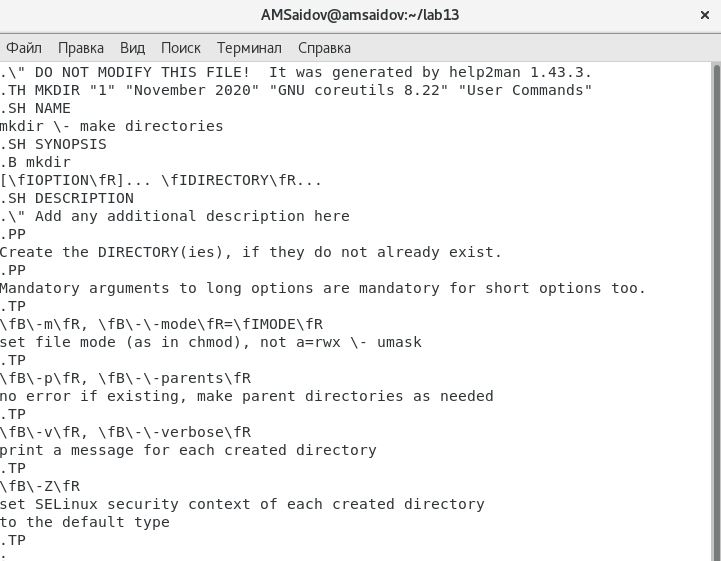
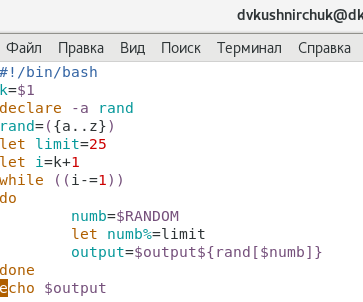
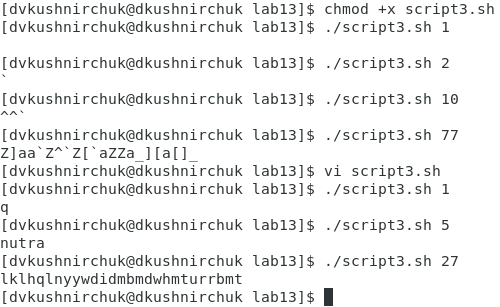
Руководитель: Валиева Татьяна Рефатовна

Москва 2023

# Цель работы

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

# Начало работы

1. Напишем 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.
   * Для этого создадим файл script1.sh
   * 
   * *Рисунок 1*
   * Откроем файл с помощью команды vi и запишем следующий код.
   * 
   * *Рисунок 2*
   * Предоставим права владельцу и проверим работу скрипта
   * 
   * *Рисунок 3*
2. Напишем 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 создал файл: semaphore.sh (Рисунки 1,2) и написал соответствующий скрипт.
   * Сперва изучим содержимое каталога */usr/share/man/man1*
   * 
   * *Рисунок 4*
   * Далее создадим файл script2.sh
   * 
   * *Рисунок 5*
   * При помощи команды vi откроем файл и запишем следующий код.
   * 
   * *Рисунок 6*
   * Предоставим права владельцу и проверим работу скрипта
   * 
   * *Рисунок 7*
   * 
   * *Рисунок 8*
   * 
   * *Рисунок 9*
3. Используя встроенную переменную $RANDOM, напишем командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтем, что $RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767.
   * Для этого создадим файл script3.sh
   * 
   * *Рисунок 10*
   * Откроем файл с помощью команды vi и запишем следующий код.
   * 
   * *Рисунок 11*
   * Предоставим права владельцу и проверим работу скрипта
   * 
   * *Рисунок 12*

# Вывод

Мы изучили основы программирования в оболочке ОС UNIX. Научились 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:
   * eq 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. or ((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