Лабораторная работа №14

Дисциплина: Операционные системы

Жибицкая Евгения Дмитриевна

Содержание

# 1 Цель работы

Изучение основ программирования в оболочке ОС UNIX. Приобретение практических навыков в написании более сложных командных файлов с использованием логических управляющих конструкций и циклов.

# 2 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.

# 3 Выполнение лабораторной работы

Для написания программы необходимо создать файл, открыть его в любом редакторе(желательно emacs, для удобной работы с подсветкой кода), наделить правом на исполнение(рис. 1).

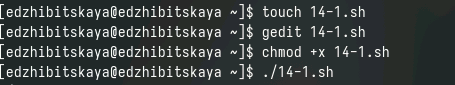


Рис. 1: Создание файла 14-1.sh

Далее реализуем код для первого задания - освобождения ресурса, выдавая об этом сообщение, также выдавать 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. Иметь возможность взаимодействия трёх и более процессов. (рис. 2).

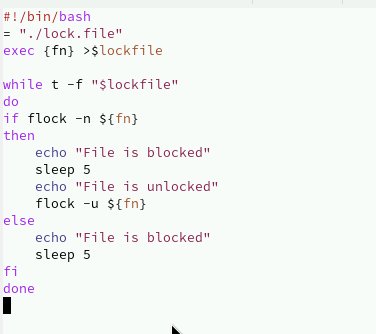


Рис. 2: Программа 1

#!/bin/bash  
= "./lock.file"  
exec {fn} >$lockfile  
  
while t -f "$lockfile"  
do  
if flock -n ${fn}  
then  
 echo "File is blocked"  
 sleep 5  
 echo "File is unlocked"  
 flock -u ${fn}  
else  
 echo "File is blocked"  
 sleep 5  
fi   
done

Запустим программу(./14-1.sh), посмотрим на результат (рис. 3).

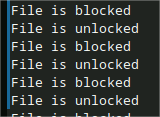


Рис. 3: Запуск файла 14-1.sh

Перейдем к реализации второй программы. Также создадим файл(рис. 4) и напишем сам код - команда man с использованием архивных файлов (рис. 5).

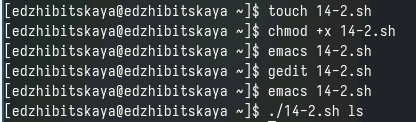


Рис. 4: Создание файла 14-2.sh

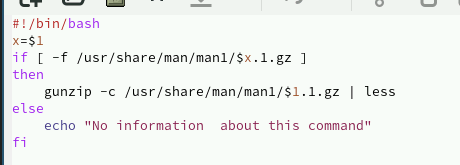


Рис. 5: Программа 2

#!/bin/bash  
x=$1  
if [ -f /usr/share/man/man1/$x.1.gz ]  
then  
 gunzip -c /usr/share/man/man1/$1.1.gz | less  
else  
 echo "No information about this command"  
fi

Посмотрим на вывод программы - man ls (рис. 6).

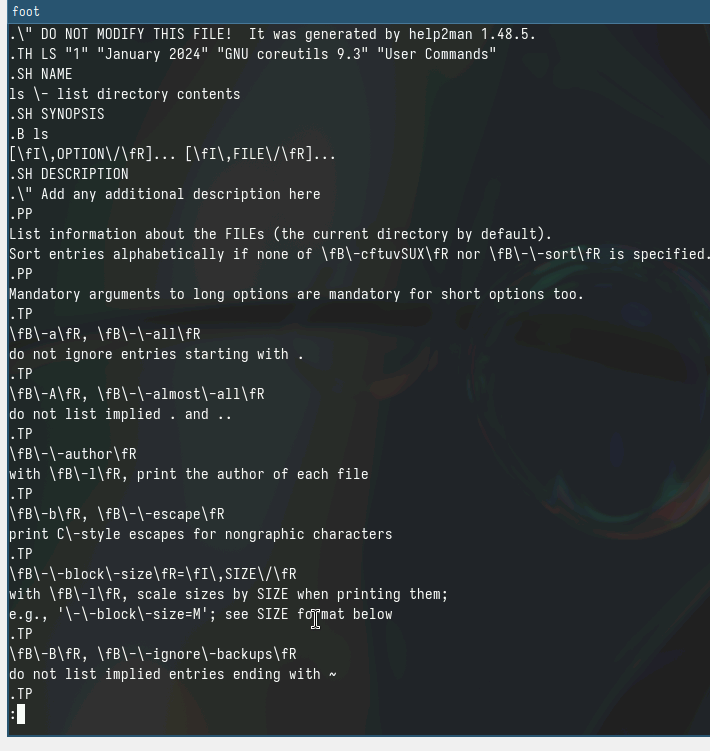


Рис. 6: Результат запуска файла 14-2.sh

Далее напишем скрипт для 3 программы. Также создадим файл и напишем код - генерация случайной последовательности букв латинского алфавита (рис. 7).

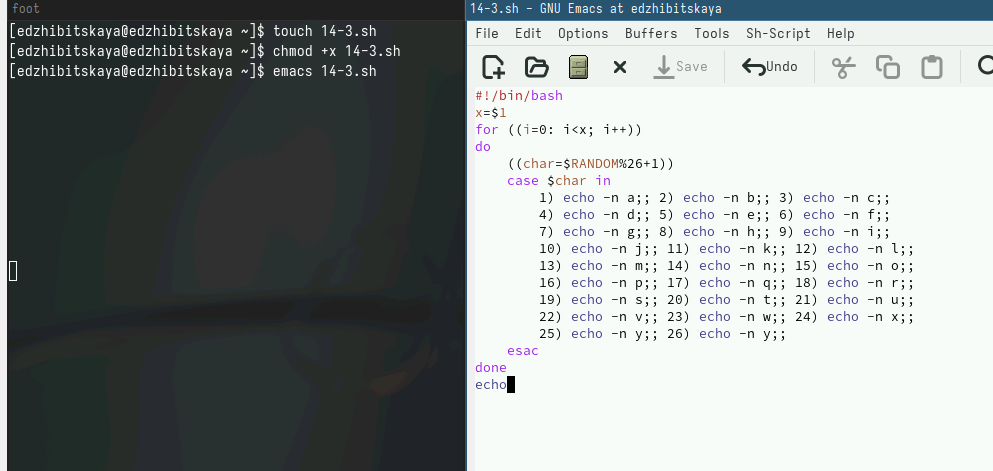


Рис. 7: Файл 14-3.sh

#!/bin/bash  
x=$1  
for ((i=0; i<x; i++))  
do  
 ((char=$RANDOM%26+1))  
 case $char in  
 1) echo -n a;; 2) echo -n b;; 3) echo -n c;;  
 4) echo -n d;; 5) echo -n e;; 6) echo -n f;;  
 7) echo -n g;; 8) echo -n h;; 9) echo -n i;;  
 10) echo -n j;; 11) echo -n k;; 12) echo -n l;;  
 13) echo -n m;; 14) echo -n n;; 15) echo -n o;;  
 16) echo -n p;; 17) echo -n q;; 18) echo -n r;;  
 19) echo -n s;; 20) echo -n t;; 21) echo -n u;;  
 22) echo -n v;; 23) echo -n w;; 24) echo -n x;;  
 25) echo -n y;; 26) echo -n y;;  
 esac  
done  
echo

Запустим файл, проверим корректность вывода (рис. 8).

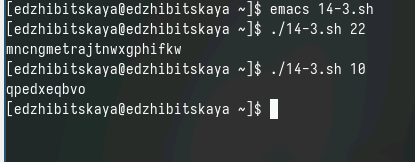


Рис. 8: Запуск файла 14-3.sh

# 4 Контрольные вопросы

1. Необходимо использовать двойные квадратные скобки для условия в цикле while.

while [[ $1 != “exit” ]]

1. Для объединения нескольких строк в одну можно использовать оператор конкатенации в bash - символ +
2. Seq возвращает последовательность чисел, можно реализовать используя цикл for или питоновский range() при программировании на bash.
3. Выражение $((10/3)) даст результат 3, так как в bash целочисленное деление возвращает только целую часть от деления.
4. Основные отличия zsh от bash включают в себя более продвинутый и гибкий автодополнитель, расширенные возможности настройки интерфейса и мощные встроенные функции. Знание bash хорошо переносится на zsh, но zsh имеет больше возможностей.
5. Синтаксис конструкции for ((a=1; a <= LIMIT; a++)) верен, это цикл for с Заданным началом, пределом и шагом.
6. Язык bash отличается от других языков программирования, таких как Python или C++, тем что он предназначен для автоматизации задач в командной строке операционной системы. Его преимущества включают встроенные утилиты для обработки текста, удобные средства работы с файлами и переменными окружения. Недостатки включают ограниченные возможности работы с более сложными структурами данных и ограниченные возможности взаимодействия с графическим интерфейсом.

# 5 Выводы

В ходе работы было изучено программирование в оболочке ОС UNIX. Приобретены навыкы в написании болеесложных командных файлов с использованием логических управляющих конструкций и циклов, а также реализовано 3 программы в данной оболочке.