

Лабораторная работа №12

Курс “Операционные Системы”

Лихтенштейн А.А., НКАбд-03-22

29 апреля 2023

Российский университет дружбы народов, Москва, Россия

- Лихтенштейн Алина Алексеевна
- студент группы НКАбд-03-22
- кафедры Компьютерные и информационные науки
- Российский университет дружбы народов
- 1132229533@pfur.ru

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

1. Написать командный файл, реализующий упрощённый механизм семафоров.

Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, 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`.

- 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.

Выполнение лабораторной работы

Исходный код скрипта №1

semaphore.sh - GNU Emacs at aaliechtenstein

File Edit Options Buffers Tools Sh-Script Help



```
#!/bin/bash
```

```
resource="semaphore.txt"
```

```
t1=5
```

```
t2=10
```

```
# установка блокировки на ресурсе
```

```
exec 200>"$resource"
```

```
flock -x 200
```

```
# ожидаем освобождения ресурса
```

```
echo "Waiting for resource..."
```

```
sleep $t1
```

```
# использование ресурса
```

```
echo "Using resource..."
```

```
echo "Process $$ is using the resource" > "$resource"
```

```
sleep $t2
```

```
# освобождение ресурса
```

```
flock -u 200
```

```
echo "Resource released"
```


Данный скрипт реализует упрощенный механизм семафоров на основе файловой блокировки. Он дожидается освобождения ресурса, выдавая об этом сообщение, затем использует его, выдавая информацию о том, что ресурс используется соответствующим процессом. Ресурс защищен блокировкой на файле `semaphore.txt`. Скрипт можно запустить в фоновом режиме и перенаправить его вывод в другой терминал.

Результат выполнения скрипта №1

```
aaliechtenstein@aaliechtenstein:~/scripts/script1 — tmux
Запуск скрипта в терминале /dev/pts/27
Запуск скрипта в терминале /dev/pts/28
Запуск скрипта в терминале /dev/pts/3
Запуск скрипта в терминале /dev/pts/4
Запуск скрипта в терминале /dev/pts/5
Запуск скрипта в терминале /dev/pts/6
Запуск скрипта в терминале /dev/pts/7
Запуск скрипта в терминале /dev/pts/8
Запуск скрипта в терминале /dev/pts/9
[aaliechtenstein@aaliechtenstein script1]$
[aaliechtenstein@aaliechtenstein script1]$ tty
/dev/pts/27
[aaliechtenstein@aaliechtenstein script1]$
[aaliechtenstein@aaliechtenstein script1]$ tty
/dev/pts/28
[aaliechtenstein@aaliechtenstein script1]$
[aaliechtenstein@aaliechtenstein script1]$ tty
/dev/pts/26
[aaliechtenstein@aaliechtenstein script1]$
```

Исходный код скрипта №2

Этот скрипт предоставляет пользователю возможность получить справку для команды, указанной в качестве аргумента. Он использует стандартную команду `man` с указанием пути к каталогу с `man`-страницами через переменную окружения `MANPATH`.

Скрипт проверяет, указано ли имя команды в качестве аргумента. Если аргумент не указан, выводит справочное сообщение и завершает работу. Запоминает указанное имя команды и путь к каталогу `man`-страниц. Устанавливает значение переменной окружения `MANPATH` равным пути к каталогу `man`-страниц. Вызывает стандартную команду `man` с именем команды в качестве аргумента.

Результат выполнения скрипта №2

```
aaliechtenstein@aaliechtenstein:~/scripts/script2- /bin/bash ./my_man.sh less
LESS(1)                                General Commands Manual                                LESS(1)

NAME
    less - opposite of more

SYNOPSIS
    less -?
    less --help
    less -V
    less --version
    less [-[+]aAbCcDeEfFgGiIjKlMmNnQqRrSsUuVvWwX~-]
        [-b space] [-h lines] [-j line] [-k keyfile]
        [-{oO} logfile] [-p pattern] [-P prompt] [-t tag]
        [-T tagfile] [-x tab,...] [-y lines] [-[z] lines]
        [-# shift] [+[[+]cmd] [--] [filename]...
    (See the OPTIONS section for alternate option syntax with long option
    names.)

DESCRIPTION
    less is a program similar to more(1), but which allows backward move-
    ment in the file as well as forward movement. Also, less does not have
    to read the entire input file before starting, so with large input
    files it starts up faster than text editors like vi(1). less uses
    termcap (or terminfo on some systems), so it can run on a variety of
    terminals. There is even limited support for hardcopy terminals. (On
    a hardcopy terminal, lines which should be printed at the top of the
    Manual page less(1) line 1 (press h for help or q to quit)
```

Исходный код скрипта №3

script.sh - GNU Emacs at aaliechtenstein

File Edit Options Buffers Tools Sh-Script Help



```
#!/bin/bash
```

```
# Задайте количество букв в последовательности
sequence_length=10
```

```
# Создайте пустую строку
random_string=""
```

```
# Цикл для генерации последовательности
```

```
for ((i=0; i<$sequence_length; i++)); do
```

```
  # Генерируйте случайное число от 0 до 25 с использованием $RANDOM
  random_number=$((($RANDOM % 26))
```

```
  # Преобразуйте случайное число в соответствующую букву латинского алфавита
  random_letter=$(printf "\\${printf '%03o' $((random_number + 65))}")
```

```
  # Добавьте случайную букву к строке
  random_string+="$random_letter"
```

```
done
```

```
# Выведите случайную последовательность
```

```
echo "$random_string"
```

Скрипт генерирует случайную последовательность латинских букв заданной длины. Он использует переменную `$RANDOM` для получения псевдослучайных чисел, преобразует их в соответствующие буквы и объединяет их в строку. Затем выводит эту строку на экран.

Результат выполнения скрипта №3

```
aaliechtenstein@aaliechtenstein:~/scripts/script3
[aaliechtenstein@aaliechtenstein script3]$ chmod +x script.sh
[aaliechtenstein@aaliechtenstein script3]$ ./script.sh
GHAEFWUSXW
[aaliechtenstein@aaliechtenstein script3]$ ./script.sh
AWWHDPPIQD
[aaliechtenstein@aaliechtenstein script3]$ ./script.sh
DAJXTQFFQH
[aaliechtenstein@aaliechtenstein script3]$ ./script.sh
THKSKQVLUB
[aaliechtenstein@aaliechtenstein script3]$ ./script.sh
ZEUTSUTWLS
[aaliechtenstein@aaliechtenstein script3]$ ./script.sh
EGOCPHQXZA
[aaliechtenstein@aaliechtenstein script3]$
```


Результаты

Были получены практические навыки написания более сложные командных файлов с использованием логических управляющих конструкций и циклов.