Лабораторная работа №14

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Кижваткина Анна Юрьевна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

Создаем директорию lab14 и перемещаемся в неё. (рис. 2.1)

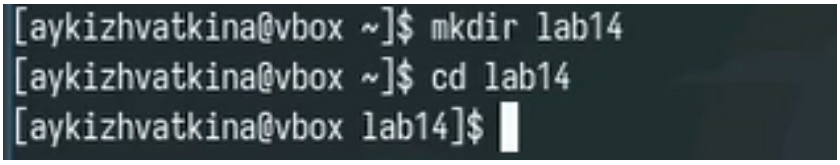

Рис. 2.1: Создание директории

Создаем файл 1.sh. (рис. 2.2)

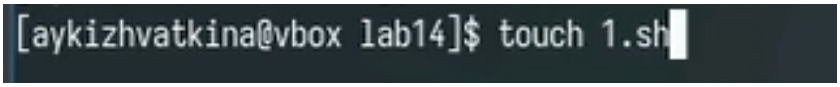

Рис. 2.2: Создание файла

Пишем 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. . (рис. 2.3)

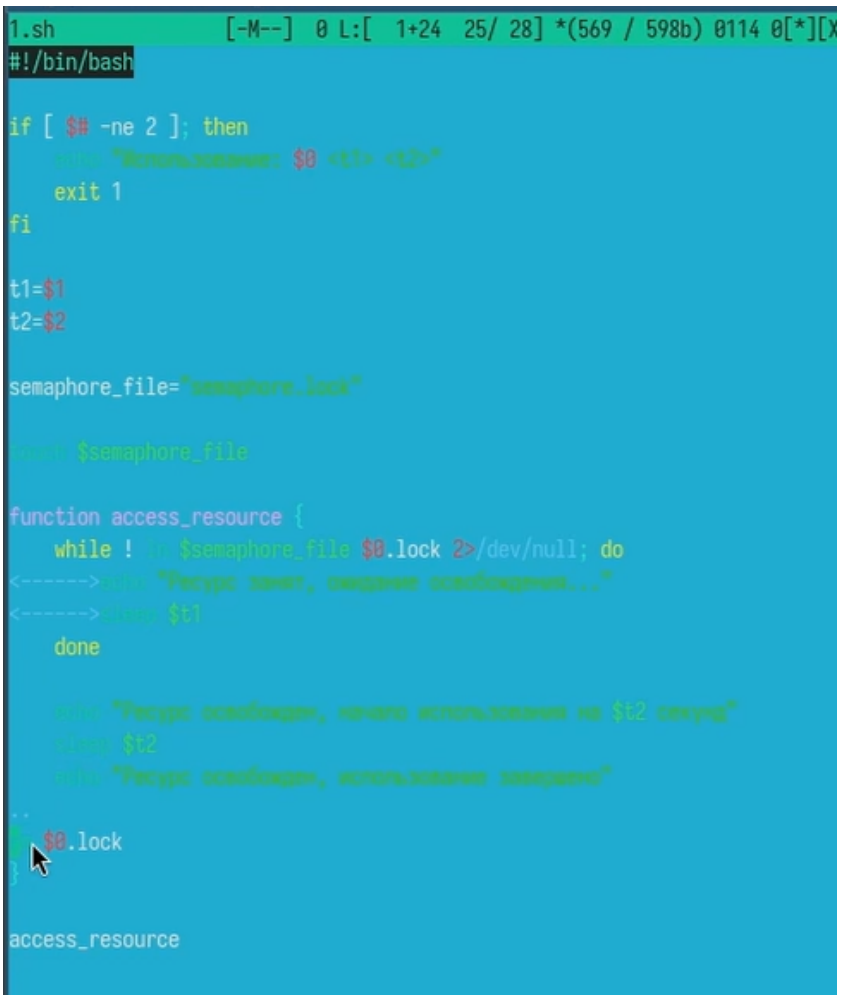

Рис. 2.3: Программа

Устанавливаем право на выполнение. (рис. 2.4)

Рис. 2.4: Предоставление доступа

Проверяем выполнение программы. (рис. 2.5)

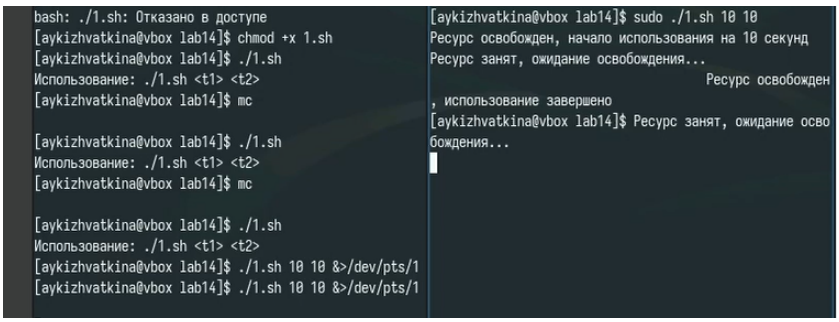

Рис. 2.5: Запуск программы

Создаем файл 2.sh и устанавливаем право на выполнение. (рис. 2.6)

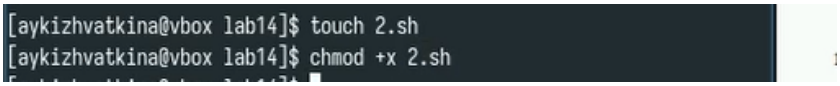

Рис. 2.6: Создание файла

Реализовываем 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. (рис. 2.7 и рис. 2.8)

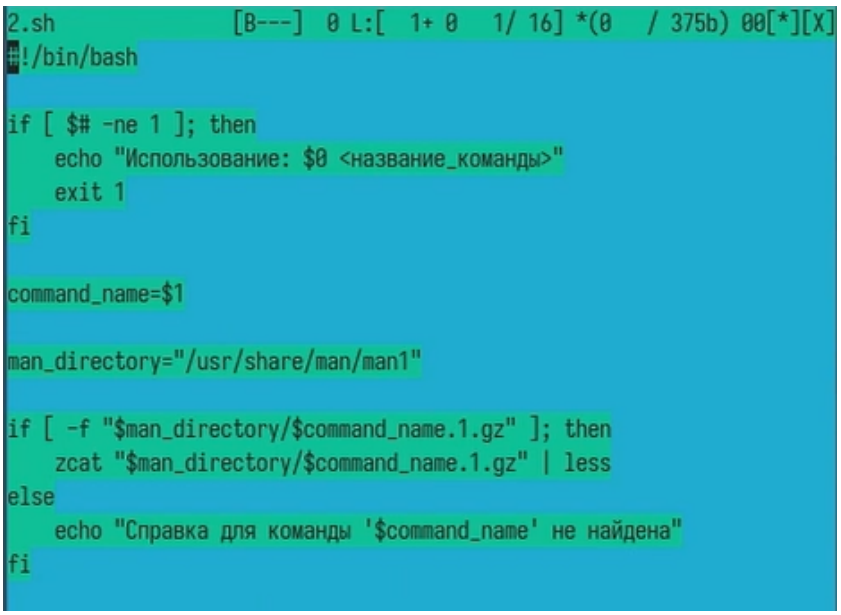

Рис. 2.7: Программа

Проверяем выполнение программы. (рис. 2.8)

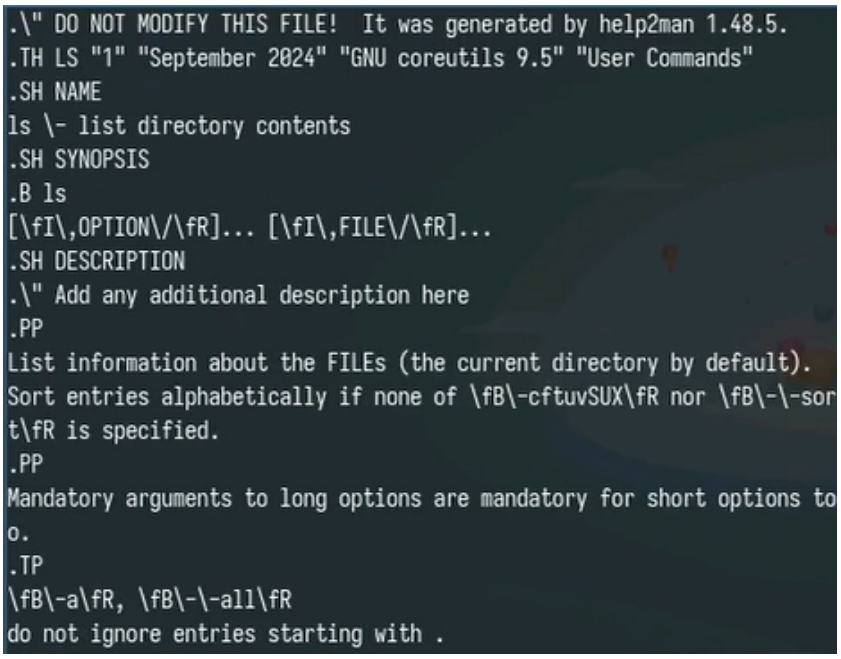

Рис. 2.8: Запуск программы

Создаем файл 3.sh и устанавливаем право на выполнение. (рис. 2.9)

```
[aykizhvatkina@vbox lab14]$ touch 3.sh
[aykizhvatkina@vbox lab14]$ chmod +x 3.sh
```

Рис. 2.9: Создание файла

Используя встроенную переменную $RANDOM, пишем командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что $RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767. (рис. 2.10)

```
3.sh              [BM--]  3 L:[  1+ 0   1/ 17] *(3   / 463b) 0098 0x062
#!/bin/bash

generate_random_letter() {
    # Случайное число от 0 до 25
    random_number=$((RANDOM % 26))

    letter=$(printf "\\$(printf '%03o' $((65 + random_number)))")

    echo -n "$letter"
}

random_sequence=""
for ((i=0; i<10; i++)); do
    random_sequence="$random_sequence$(generate_random_letter)"
done

echo "Случайная последовательность букв латинского алфавита: $random_sequence"
```

Рис. 2.10: Программа

Проверяем выполнение программы. (рис. **??**)

```
[aykizhvatkina@vbox lab14]$ ./3.sh
Случайная последовательность букв латинского алфавита: V)W)P)K)V)G)Z)W)N)T)
[aykizhvatkina@vbox lab14]$ ./3.sh
Случайная последовательность букв латинского алфавита: X)J)A)C)A)D)F)T)I)R)
```

{#fig:011width=70%

3 Выводы

Я изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.