

Отчёт по лабораторной работе №14

Операционные системы

Чистов Даниил Максимович

Содержание

1	Цель работы	4
2	Задания	5
3	Выполнение лабораторной работы	6
4	Программа 1	7
5	Программа 2	9
6	Программа 3	11
7	Выводы	13
	Список литературы	14

Список иллюстраций

4.1	Задание 1 - выполнено вид из терминала 1	7
4.2	Задание 1 - выполнено вид из терминала 2	7
4.3	Задание 1 - программа	8
5.1	Задание 2 - выполнено	9
5.2	Задание 2 - выполнено	10
5.3	Задание 2 - программа 1	10
6.1	Задание 3 - выполнено	11
6.2	Задание 3 - программа	12

1 Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

2 Задания

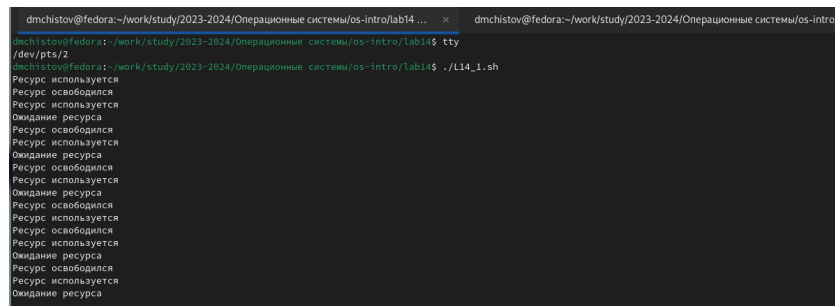
1. Программа 1
2. Программа 2
3. Программа 3

3 Выполнение лабораторной работы

4 Программа 1

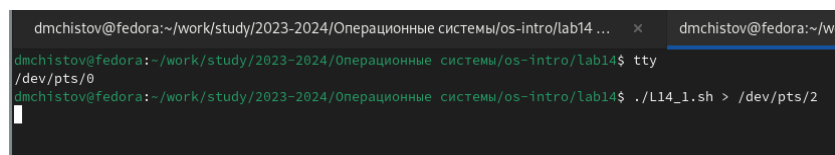
Задание: “Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию 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 имела возможность взаимодействия трёх и более процессов.

Приступаю к выполнению работы. Создаю файл с кодом и пишу код (сама программа будет показана позже), выполняю - готово (рис. 4.1), (рис. 4.2).



```
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14 ... x dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/...
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ tty
/dev/pts/2
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_1.sh
Ресурс используется
Ресурс освобожден
Ресурс используется
Ожидание ресурса
Ресурс освобожден
Ресурс используется
Ожидание ресурса
Ресурс освобожден
Ресурс используется
Ожидание ресурса
Ресурс освобожден
Ресурс используется
Ожидание ресурса
Ресурс освобожден
Ресурс используется
Ожидание ресурса
Ресурс освобожден
Ресурс используется
Ожидание ресурса
```

Рис. 4.1: Задание 1 - выполнено вид из терминала 1



```
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14 ... x dmchistov@fedora:~/wo...
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ tty
/dev/pts/0
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_1.sh > /dev/pts/2
```

Рис. 4.2: Задание 1 - выполнено вид из терминала 2

Код программы 1: создаём файл, и запускаем его. Бесконечно идём циклом while и проверяем утилитой flock, если программа используется, говорим об этом, ждём 5 секунд и освобождаем ресурс. Если не используется, то пишем, что ожидаем, и ждём 8 секунд. (рис. 4.3).

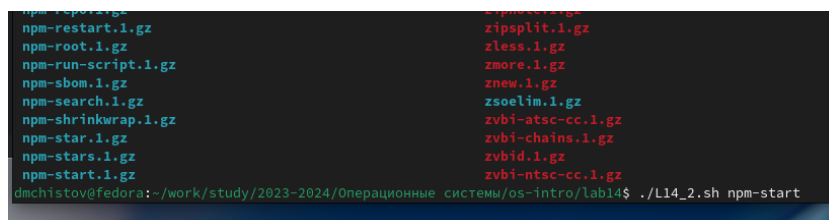
```
1  #!/bin/bash
2
3  file="./.lock.file"
4  exec {fn}<>$file
5
6  while true
7  do
8      if flock -n ${fn}; then
9          echo "Ресурс используется"
10         sleep 5
11         flock -u ${fn}
12         echo "Ресурс освобожден"
13         sleep 1
14     else
15         echo "Ожидание ресурса"
16         sleep 8
17     fi
18 done
19
```

Рис. 4.3: Задание 1 - программа

5 Программа 2

Задание: “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`.”

Приступаю к выполнению - создаю, даю права, пишу код, исправляю ошибки, запускаю - работает (рис. 5.1), (рис. 5.2).



```
npm-repair.1.gz      zipmerge.1.gz
npm-restart.1.gz     zipsplit.1.gz
npm-root.1.gz        zless.1.gz
npm-run-script.1.gz  zmore.1.gz
npm-sbom.1.gz        znew.1.gz
npm-search.1.gz      zsoelim.1.gz
npm-shrinkwrap.1.gz  zvbi-atsc-cc.1.gz
npm-star.1.gz        zvbi-chains.1.gz
npm-stars.1.gz       zvbi.1.gz
npm-start.1.gz       zvbi-ntsc-cc.1.gz
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_2.sh npm-start
```

Рис. 5.1: Задание 2 - выполнено

```
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14 ... x dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14 ... x
ESC [4mNPM-STARTESC [24m(1)
4mNPM-STARTESC [24m(1)
ESC [1mNPMESC [0m
ESC [1mnpminstallESC [22m- Start a package
ESC [1msynopsisESC [0m
npm start [-- <args>]
ESC [1mdescriptionESC [0m
This runs a predefined command specified in the package's "scripts" property.
If the "scripts" object does not define a "start" property, npm will run "node server.js".
Note that this is different from the default node behavior of running the file specified in a package's "main" attribute when evoked with "node".
node .
ESC [0m
As of npm 2.0.0, you can use custom arguments when executing scripts. Refer
to npm help run-script for more details.
ESC [1mExampleESC [0m
{
  "scripts": {
    "start": "node foo.js"
  }
}
npm start
> npm&x.x start
> node foo.js
(foo.js output would be here)
ESC [1mConfigurationESC [0m
/usr/share/man/man1/npm-start.1.gz
```

Рис. 5.2: Задание 2 - выполнено

Код программы 2: Создаю переменную (название команды, которую мы хотим изучить), утилитой test проверяем, есть ли документация по этой команде в заданной директории. Если есть, то читаем её утилитой less, если нету, пишем что такой документации нет (рис. 5.3).

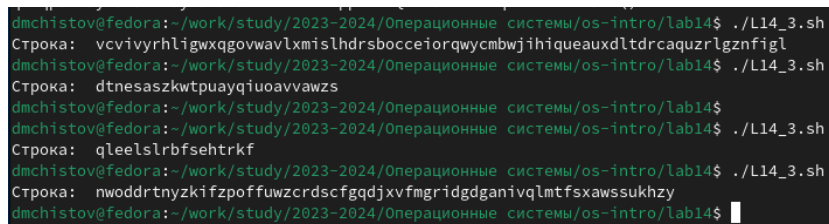
```
1  #!/bin/bash
2
3  com=$1
4
5  if test -f "/usr/share/man/man1/$com.1.gz"
6  then less /usr/share/man/man1/$com.1.gz
7  else
8      echo "Такой справки нету :("
9
10 fi
11
```

Рис. 5.3: Задание 2 - программа 1

6 Программа 3

Задание: “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.”

Создаю файл, даю права, начинаю писать код, всё работает (рис. 6.1).



```
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_3.sh
Строка: vcvivyrlhigwxqgovwavlxmislhdsbocceiorqwymbjihiqueauxdltdrcaquzrlgznfigl
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_3.sh
Строка: dtneasazkwtpuayqiuoavvawzs
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_3.sh
Строка: qleelslrbfsehtrkf
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$ ./L14_3.sh
Строка: nwoddrtnyzkifzpoffuwzcrdscfgqdxvfmgridgaganivqlmtfsxawssukhzy
dmchistov@fedora:~/work/study/2023-2024/Операционные системы/os-intro/lab14$
```

Рис. 6.1: Задание 3 - выполнено

Код программы 3: Создаю переменную - строку со всеми буквами латинского алфавита. Затем задаю ещё одну переменную “Длина строки” - утилитой \$RANDOM выбираю случайное число. Это и будет длина. Задаю переменную результата - пустую строку. Циклом for иду столько число раз, сколько в перменной “длина” В каждой итерации - беру индекс - утилитой \$RANDOM это будет случайное число до 26. Затем временной переменной letter достаю из строки из алфавита букву, которая находится по ранее найденному индексу. Приписываю эту букву к перменной результат. Вывожу (рис. 6.2).

```
1  #!/bin/bash
2
3  alphabet="abcdefghijklmnopqrstuvwxyz"
4
5  length=$(( $RANDOM % 100 + 1 ))
6
7  result=""
8
9  for ((i=0; i<length; i++))
10 do
11     index=$(( $RANDOM % 26 ))
12     letter="${alphabet:index:1}"
13     result+="$letter"
14 done
15
16 echo "Строка: " $result
17
18
```

Рис. 6.2: Задание 3 - программа

7 Выводы

В результате выполнения данной работы я изучил основы программирования в оболочке ОС UNIX/Linux. Научился писать небольшие командные файлы.

Список литературы

Лабораторная работы №14

Работа с строками и shell scripting

Команда flock