

# Лабораторная работа №14

Операционные системы

---

Тойчубекова Асель Нурлановна

10 мая 2024

Российский университет дружбы народов, Москва, Россия

# Информация

---

- Тойчубекова Асель Нурлановна
- Студент НПИбд-02-23
- факультет физико математических и естественных наук
- Российский университет дружбы народов
- 1032235033@rudn.ru
- <https://aseltoichubekova.github.io/ru/>

## Цель работы

---

Целью данной лабораторной рабораторной работы является изучение основ программирования в оболочке ОС Unix. Также научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

## Задание

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, в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имела возможность взаимодействия трёх и более процессов.

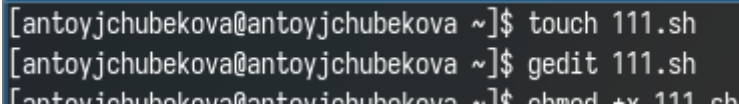
2. Реализовать команду `man` с помощью командного файла. Изучите содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. 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.

# Выполнение лабораторной работы

---



Для начала создаю файл в котором буду писать саму программу и открываю его в редакторе gedit

A terminal window with a dark background and a blue title bar. It shows three lines of text: the first line is the command 'touch 111.sh', the second line is 'gedit 111.sh', and the third line is partially visible as 'chmod +x 111.sh'. Each line is preceded by the prompt '[antoyjchubekova@antoyjchubekova ~]\$'.

```
[antoyjchubekova@antoyjchubekova ~]$ touch 111.sh  
[antoyjchubekova@antoyjchubekova ~]$ gedit 111.sh  
[antoyjchubekova@antoyjchubekova ~]$ chmod +x 111.sh
```

**Рис. 1:** Создание файла

## Выполнение лабораторной работы

Редактирую файл, записывая командный файл, который реализует упрощенный механизм семофонов. Для этого я сперва создаю переменную в которой будет храниться адрес файла блокирования, а также переменную, которому присваивается значение первого аргумента. Далее открыв файл блокирования с помощью команды `exes` и приваиваю ему свободный файловый дескриптор. Далее запускаю цикл, который будет работать пока файл блокирования существует и проверяю можно ли заблокировать файл с помощью команды `flock -n`, если да, то вывожу сообщение, что файл заблокирован, затем жду столько времени, сколько было указано в аргументе и разблокирую файл с помощью команды `flock -u`, выведя об этом сообщение. Если же файл изначально был заблокирован, то я вывожу сообщения, что файл заблокирован и жду пока не истечет время, указанное в аргументе.

## Выполнение лабораторной работы

```
1 #!/bin/bash
2 lock_file="./lock.file"
3 a=$1
4 exec {fn}>$lock_file
5 while test -f "$lock_file"
6 do
7   if flock -n ${fn}
8   then
9     echo Файл заблокирован \ (Ресурс используется другим процессом)\ \!
10    sleep $a
11    echo Файл разблокирован \ (Ресурс готов к использованию)\ \!
12    flock -u ${fn}
13  else
14    echo Файл заблокирован \ (Ресурс используется другим процессом)\ \!
15    sleep $a
16  fi
17 done
```

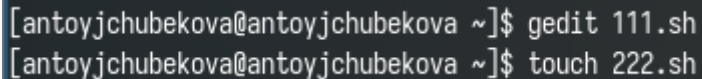
## Выполнение лабораторной работы

Даю права на выполнение и запускаю программу, мы видим, что все работает корректно.

```
[antoyjchubekova@antoyjchubekova ~]$ chmod +x 111.sh  
[antoyjchubekova@antoyjchubekova ~]$ bash 111.sh 5  
Файл заблокирован (Ресурс используется другим процессом)!  
Файл разблокирован (Ресурс готов к использованию)!  
Файл заблокирован (Ресурс используется другим процессом)!  
Файл разблокирован (Ресурс готов к использованию)!  
Файл заблокирован (Ресурс используется другим процессом)!  
Файл разблокирован (Ресурс готов к использованию)!
```

**Рис. 3:** Запуск программы

Создаю файл для написания программы второго задания.

A terminal window with a dark background and light-colored text. It shows two commands being executed in a shell. The first command is 'gedit 111.sh' and the second is 'touch 222.sh'. Both commands are preceded by the prompt '[antoyjchubekova@antoyjchubekova ~]\$'.

```
[antoyjchubekova@antoyjchubekova ~]$ gedit 111.sh  
[antoyjchubekova@antoyjchubekova ~]$ touch 222.sh
```

**Рис. 4:** Создание файла

# Выполнение лабораторной работы

Далее перехожу по ссылке `/usr/share/man/man1` и вижу архивы текстовых файлов, содержащих справки о командах.

```
ffprobe.1.gz
ffprobe-all.1.gz
fg.1.gz
fgconsole.1.gz
file.1.gz
file2brl.1.gz
filterdiff.1.gz
fincore.1.gz
find.1.gz
findhyph.1.gz
findrule.1.gz
firefox.1.gz
firewall-cmd.1.gz
firewalld.1.gz
firewall-offline-cmd.1.gz
fish.1.gz
fish_indent.1.gz
fish_key_reader.1.gz
fixcvdiff.1.gz
flex++.1.gz
flex.1.gz
flexiblas.1.gz
flipdiff.1.gz
flock.1.gz
fmt.1.gz
fmtutil.1.gz
fmtutil-sys.1.gz
fmtutil-user.1.gz
fold.1.gz
hunspell.1.gz
hyperxmp-add-bytecount.1.gz
iasecc-tool.1.gz
iconv.1.gz
id.1.gz
identify.1.gz
iecset.1.gz
ImageMagick.1.gz
implantisond5.1.gz
import.1.gz
imv.1.gz
imv-dir.1.gz
imv-msg.1.gz
imv-wayland.1.gz
imv-x11.1.gz
includes.1.gz
info.1.gz
info.lossl.gz
infocmp.1m.gz
infotocap.1m.gz
inif.1.gz
init.1.gz
initex.1.gz
inkscape.1.gz
install.1.gz
install-info.1.gz
install-tl.1.gz
interdiff.1.gz
intro.1.gz
```

Открываю созданный файл в редакторе и редактирую его, записывая командный файл, который реализует команду `man`, перейдя по адресу `/usr/share/man/man1` и используя архивы данных о командах выполняет эту команду. Для этого сперва завожу переменную, которой присваиваю значения аргумента, затем проверяю есть ли подходящий файл в `man1`, если есть с помощью команды `less` вывожу ее, а если нет вывожу соответствующее сообщение.

## Выполнение лабораторной работы

```
1 #!/bin/bash
2 a=$1
3 if test -f /usr/share/man/man1/$a.1.gz
4 then
5 less /usr/share/man/man1/$a.1.gz
6 else
7 echo Описание команды не найдено\!
8 fi
```

Рис. 6: Редактирование файла



# Выполнение лабораторной работы

Даю права на выполнение и запускаю программу, мы видим, что все работает правильно и программа выдает нам сообщение о команде.

```
ESC[4mCPESC[24m(1)
ESC[4mCFESC[24m(1)

ESC[1mNAMEESC[0m
cp - copy files and directories

ESC[1mSYNOPSISESC[0m
ESC[1mcp ESC[22m[ESC[4mOPTIONESC[24m]... [ESC[4m-TESC[24m] ESC[4mSOURCEESC[24m ESC[4mDESTESC[0m
ESC[1mcp ESC[22m[ESC[4mOPTIONESC[24m]... ESC[4mSOURCEESC[24m... ESC[4mDIRECTORYESC[0m
ESC[1mcp ESC[22m[ESC[4mOPTIONESC[24m]... ESC[4m-tESC[24m ESC[4mDIRECTORYESC[24m ESC[4mSOURCEESC[24m...

ESC[1mDESCRIPTIONESC[0m
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

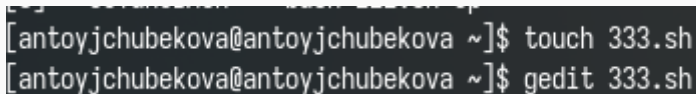
Mandatory arguments to long options are mandatory for short options too.

ESC[1m-aESC[22m, ESC[1m--archiveESC[0m
same as ESC[1m-dR --preserveESC[22m=ESC[4mallESC[0m

ESC[1m--attributes-onlyESC[0m
don't copy the file data, just the attributes

ESC[1m--backupESC[22m[=ESC[4mCONTROLESC[24m]
make a backup of each existing destination file
```

Создаю файл для написания программы для третьего задания и открываю его в редакторе.

A terminal window with a dark background and light gray text. It shows two commands being executed in a shell. The first command is 'touch 333.sh' and the second is 'gedit 333.sh'. Both commands are preceded by the prompt '[antoyjchubekova@antoyjchubekova ~]\$'.

```
[antoyjchubekova@antoyjchubekova ~]$ touch 333.sh  
[antoyjchubekova@antoyjchubekova ~]$ gedit 333.sh
```

**Рис. 8:** Создание файла

Редактирую файл, записывая командный файл, который используя встроенную переменную \$RANDOM, генерирует случайную последовательность букв латинского алфавита. Для этого я создаю переменную и присваиваю ей значения аргумента, далее запускаю цикл, который будет продолжаться пока i не станет равным аргументу, запускаю RANDOM и в соответствии с полученным числом с помощью команды case перебираю по цифрам все буквы, если совпадает вывожу на экран, в конце перехожу на новую строку.

# Выполнение лабораторной работы

```
1 #!/bin/bash
2 a=$1
3 for((i=0;i<a;i++))
4 do
5   rnum=$((RANDOM % 26+1))
6   case $rnum in
7     1) echo -n a;;
8     2) echo -n b;;
9     3) echo -n c;;
10    4) echo -n d;;
11    5) echo -n e;;
12    6) echo -n f;;
13    7) echo -n g;;
14    8) echo -n h;;
15    9) echo -n i;;
16    10) echo -n j;;
17    11) echo -n k;;
18    12) echo -n l;;
19    13) echo -n m;;
20    14) echo -n n;;
21    15) echo -n o;;
22    16) echo -n p;;
23    17) echo -n q;;
24    18) echo -n r;;
25    19) echo -n s;;
26    20) echo -n t;;
27    21) echo -n u;;
28    22) echo -n v;;
29    23) echo -n 'w';;
30    24) echo -n x;;
31    25) echo -n y;;
32    26) echo -n z;;
33  esac
34 done
35 echo
36
```

## Выполнение лабораторной работы

Даю права на выполнение и запускаю программу, мы видим, что программа правильно работает и выводит латинские буквы.

```
[antoyjchubekova@antoyjchubekova ~]$ chmod +x 333.sh  
[antoyjchubekova@antoyjchubekova ~]$ bash 333.sh 5  
piueh  
[antoyjchubekova@antoyjchubekova ~]$ bash 333.sh 10  
cbfptvxrcb  
[antoyjchubekova@antoyjchubekova ~]$ bash 333.sh 86  
tiasfjenugoxbedhujdafpfwvocwcyibaapineycuxenwuqeyyyrljlpplxlinyssfthlgadfvntxexkhqmaugr  
[antoyjchubekova@antoyjchubekova ~]$
```

Рис. 10: Запуск программы

## Выводы

---

В ходе лабораторной работы №14 я изучила основы программирования в оболочке ОС Unix. Также научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.