

Цель работы

Изучить основы программирования в оболочке ОС UNIX. 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 (`> /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.

Выполнение лабораторной работы

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.



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

```
# Проверяем, есть ли аргументы
```

```
if [ $# -ne 4 ]; then
```

```
    echo "Usage: $0 <wait_time> <use_time> <resource_file> <output_terminal>"
```

```
    exit 1
```

```
fi
```

```
# Аргументы
```

```
wait_time=$1
```

```
use_time=$2
```

```
resource_file=$3
```

```
output_terminal=$4
```

```
while true; do
```

```
    # Попытка захватить ресурс
```

```
    if ln "$resource_file" "$resource_file.lock" 2>/dev/null; then
```

```
        echo "$(date): Resource is being used by $$" > "$output_terminal"
```

```
        sleep "$use_time"
```

```
        rm "$resource_file.lock"
```

```
        echo "$(date): Resource released by $$" > "$output_terminal"
```

```
        break
```

```
    else
```

```
        echo "$(date): Resource is busy, waiting..." > "$output_terminal"
```

```
        sleep "$wait_time"
```

```
    fi
```

```
done
```

```
~
```

```
~
```

```
~
```

```
~
```

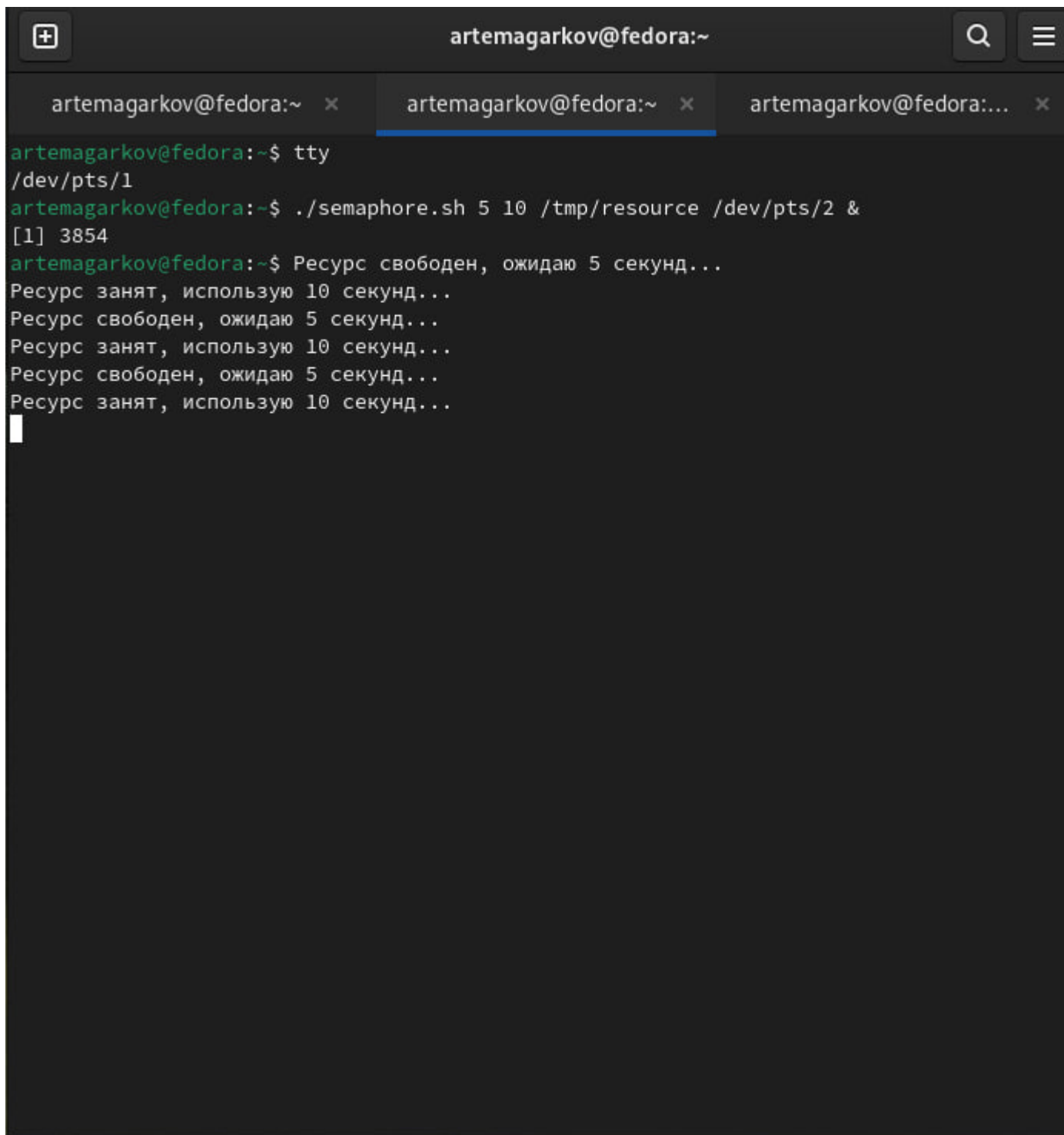
```
~
```

```
~
```

```
~
```

```
"semaphore.sh" 28L, 740B
```

```
1.1
```



```
artemagarkov@fedora:~$ tty
/dev/pts/1
artemagarkov@fedora:~$ ./semaphore.sh 5 10 /tmp/resource /dev/pts/2 &
[1] 3854
artemagarkov@fedora:~$ Ресурс свободен, ожидаю 5 секунд...
Ресурс занят, использую 10 секунд...
Ресурс свободен, ожидаю 5 секунд...
Ресурс занят, использую 10 секунд...
Ресурс свободен, ожидаю 5 секунд...
Ресурс занят, использую 10 секунд...
```

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`.



artemagarkov@fedora:~ — /bin/bash ./semaphore.sh 5 10 /tmp/resource /de...



artemagarkov... x

artemagarkov... x

artemagarkov... x

artemagarkov... x

artemagarkov@fedora:~\$ tty

/dev/pts/2

artemagarkov@fedora:~\$ Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

^C

artemagarkov@fedora:~\$./semaphore.sh 5 10 /tmp/resource /dev/pts/2

Ресурс свободен, ожидаю./semaphore.sh 5 10 /tmp/resource /dev/pts/1

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

Ресурс занят, использую 10 секунд...

Ресурс свободен, ожидаю 5 секунд...

artemagarkov@fedora:~ — nano myman.sh

GNU nano 7.2myman.shModifi

```
#!/bin/bash

# Проверяем, передано ли название команды в аргументах командной строки
if [ $# -ne 1 ]; then
    echo "Usage: $0 <command>"
    exit 1
fi

# Проверяем, существует ли файл справки для указанной команды
if [ -e "/usr/share/man/man1/$1.1.gz" ]; then
    # Если файл существует, открываем его с помощью команды less
    less "/usr/share/man/man1/$1.1.gz"
else
    # Если файла нет, выводим сообщение об отсутствии справки
    echo "No manual entry for $1"
fi
```

^G Help

^X Exit

^O Write Out

^R Read File

^W Where Is

^_ Replace

^K Cut

^U Paste

^T Execute

^J Justify

^C Location

^_ Go To Lin

artemagarkov@fedora:~ — /bin/bas

```
ESC[4mLESC[24m(1)                                User Commands                                ESC[4mLESC[24m(1)

ESC[1mNAMEESC[0m
ls - list directory contents

ESC[1mSYNOPSISESC[0m
ESC[1mls ESC[22m[ESC[4mOPTIONESC[24m]... [ESC[4mFILEESC[24m]...

ESC[1mDESCRIPTIONESC[0m
List information about the FILES (the current directory by default). Sort en-
tries alphabetically if none of ESC[1m-cftuvSUX ESC[22mnor ESC[1m--sort ESC[22mis specified.

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

ESC[1m-aESC[22m, ESC[1m--allESC[0m
do not ignore entries starting with .

ESC[1m-AESC[22m, ESC[1m--almost-allESC[0m
do not list implied . and ..

ESC[1m--authorESC[0m
with ESC[1m-lESC[22m, print the author of each file

ESC[1m-bESC[22m, ESC[1m--escapeESC[0m
print C-style escapes for nongraphic characters

ESC[1m--block-sizeESC[22m=ESC[4mSIZEESC[0m
with ESC[1m-lESC[22m, scale sizes by SIZE when printing them; e.g., '--block-size=M';
see SIZE format below

ESC[1m-BESC[22m, ESC[1m--ignore-backupsESC[0m
do not list implied entries ending with ~

ESC[1m-c ESC[22mwith ESC[1m-lESC[22m: sort by, and show, ctime (time of last change of file status in-
formation); with ESC[1m-lESC[22m: show ctime and sort by name; otherwise: sort by
ctime, newest first

ESC[1m-C ESC[22mlist entries by columns

ESC[1m--colorESC[22m[=ESC[4mWHENESC[24m]
color the output WHEN; more info below

ESC[1m-dESC[22m, ESC[1m--directoryESC[0m
list directories themselves, not their contents

ESC[1m-DESC[22m, ESC[1m--diredESC[0m
generate output designed for Emacs' dired mode

ESC[1m-f ESC[22mlist all entries in directory order

ESC[1m-FESC[22m, ESC[1m--classifyESC[22m[=ESC[4mWHENESC[24m]
append indicator (one of */=>@|) to entries WHEN

ESC[1m--file-typeESC[0m
likewise, except do not append '+'

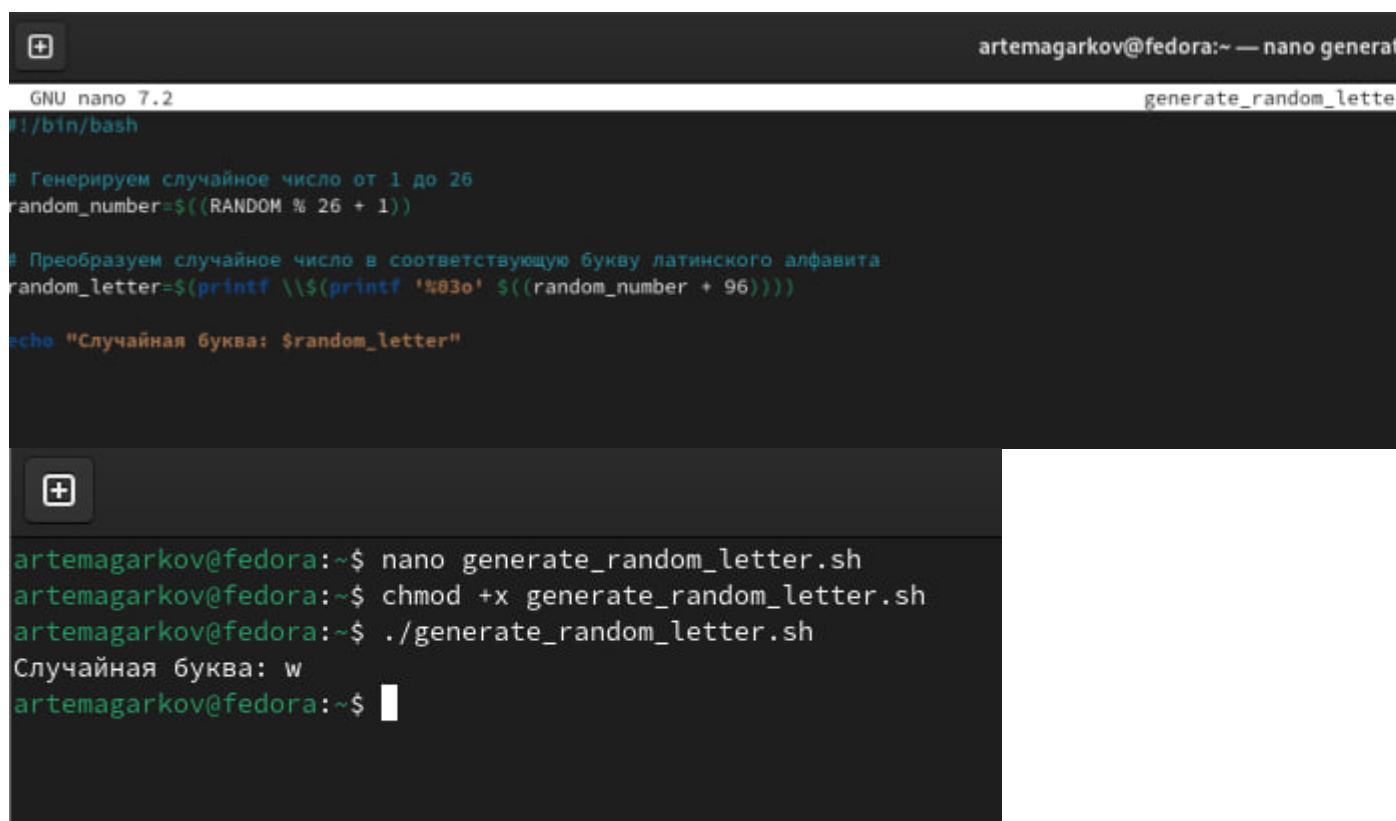
ESC[1m--formatESC[22m=ESC[4mWORDESC[0m
across ESC[1m-xESC[22m, commas ESC[1m-mESC[22m, horizontal ESC[1m-xESC[22m, long ESC[1m-lESC[22m, si
ESC[1m-lESC[22m, vertical ESC[1m-CESC[0m

ESC[1m--full-timeESC[0m
like ESC[1m-l --time-styleESC[22m=ESC[4mfull-isoESC[0m

ESC[1m-g ESC[22mlike ESC[1m-lESC[22m, but do not list owner

ESC[1m--group-directories-firstESC[0m
group directories before files; can be augmented with a ESC[1m--sort ESC[22moption, but
```

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 32767.4.



```
artemagarkov@fedora:~ — nano generate_random_letter.sh
GNU nano 7.2 generate_random_letter.sh
#!/bin/bash

# Генерируем случайное число от 1 до 26
random_number=$((RANDOM % 26 + 1))

# Преобразуем случайное число в соответствующую букву латинского алфавита
random_letter=$(printf \\$(printf '%03o' $((random_number + 96))))

echo "Случайная буква: $random_letter"

artemagarkov@fedora:~$ nano generate_random_letter.sh
artemagarkov@fedora:~$ chmod +x generate_random_letter.sh
artemagarkov@fedora:~$ ./generate_random_letter.sh
Случайная буква: w
artemagarkov@fedora:~$
```

Выводы

Я изучил основы программирования в оболочке ОС UNIX и научился писать более сложные командные файлы, используя логические управляющие конструкции и циклы.