

Лабораторная работа №13 по предмету  
Операционные системы

НПМбВ-02-19

Нечаева Виктория Алексеевна

# Содержание

Цель работы	5
Задание	6
Выполнение лабораторной работы	7
Задача 1 . . . . .	7
Задача 2 . . . . .	10
Задача 3 . . . . .	11
Выводы	14
Контрольные вопросы	15

## Список таблиц

## Список иллюстраций

1	Рисунок 1 . . . . .	8
2	Рисунок 2 . . . . .	9
3	Рисунок 3 . . . . .	9
4	Рисунок 4 . . . . .	10
5	Рисунок 5 . . . . .	11
6	Рисунок 6 . . . . .	12
7	Рисунок 7 . . . . .	13

## 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 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

## Задача 1

В качестве времени  $t_1$  задаем 10 секунд – это время на ожидание освобождения ресурса,  $t_2$  – время использования ресурса. Начальное значение семафора = 0.

В цикле `while` процесс ждет освобождения ресурса и данный цикл прерывается каждую секунд, пока время освобождения ресурса не истечет. Далее процесс может быть заблокирован, процесс может использовать ресурс и может быть разблокирован. Также процесс может ожидать в течение времени  $t_2$ .

Для вызова процесса в фоновом режиме использовать команду `./semaphore.sh > /dev/pts/№`, где № – номер терминала. Прервать процесс можно команды `kill PID`, где PID берется из фонового окна с запущенным скриптом. После прерывания программа завершает выполнение.

```
GNU nano 6.2 semaphore.sh
t1=10
t2=15
semaphore=0
lock="lockfile"
tty_num=$(tty | cut -c 9-)

while true
do
  while [ $semaphore -ne 0 ]
  do
    echo "Процесс $BASHPID ждет освобождения ресурса"
    sleep 1
  done

  echo "Процесс $BASHPID блокирует ресурс"
  touch $lock

  echo "Процесс $BASHPID использует ресурс в течение $t2 секунд"
  sleep $t2

  echo "Процесс $BASHPID освобождает ресурс"
  rm $lock

  semaphore=0

  echo "Процесс $BASHPID ожидает $t1 секунд"
  sleep $t1
  semaphore=1
done
```

**^G** Справка    **^O** Записать    **^W** Поиск    **^K** Вырезать    **^T** Выполнить    **^C** Позиция  
**^X** Выход    **^R** ЧитФайл    **^\_** Замена    **^U** Вставить    **^J** Вывод    **^\_/** К строке

Рис. 1: Рисунок 1



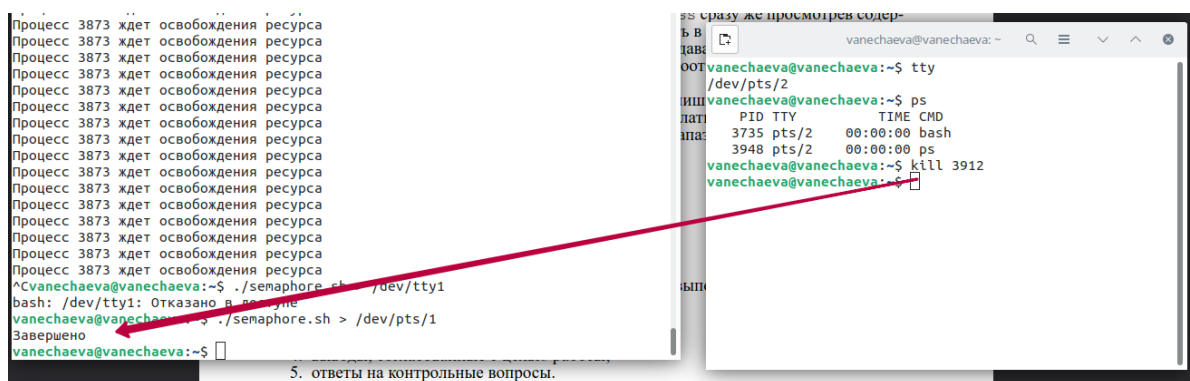


Рис. 2: Рисунок 2

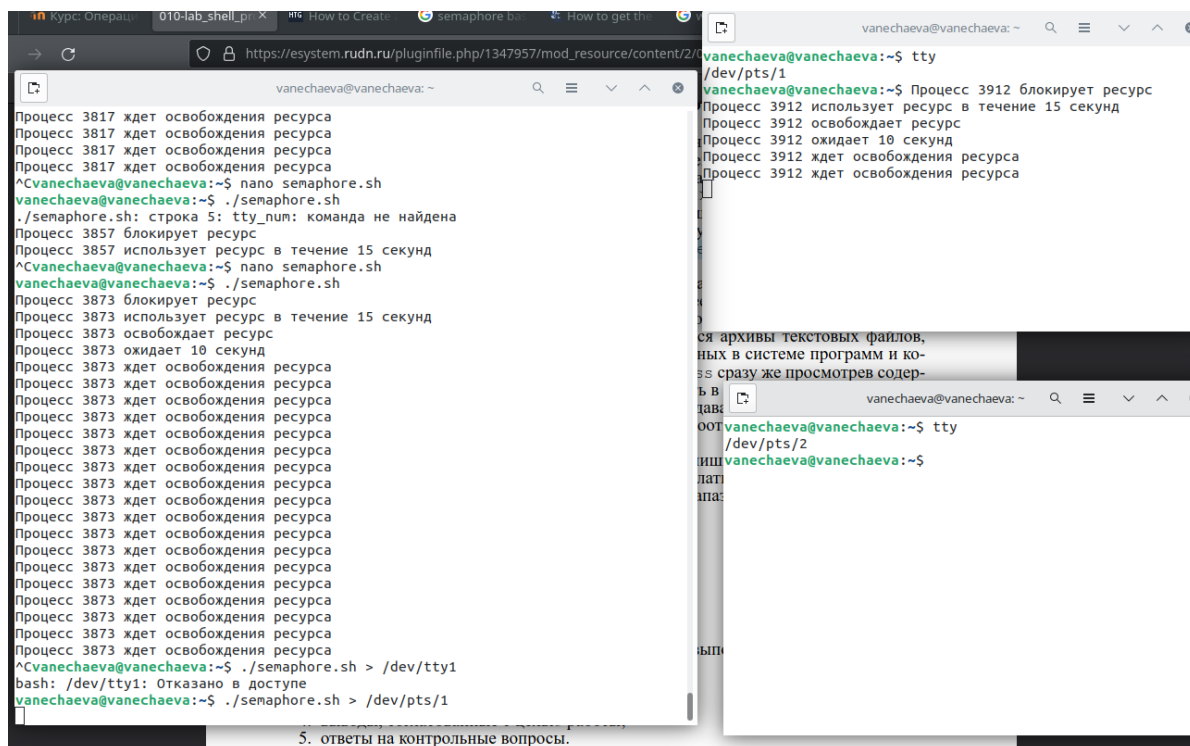
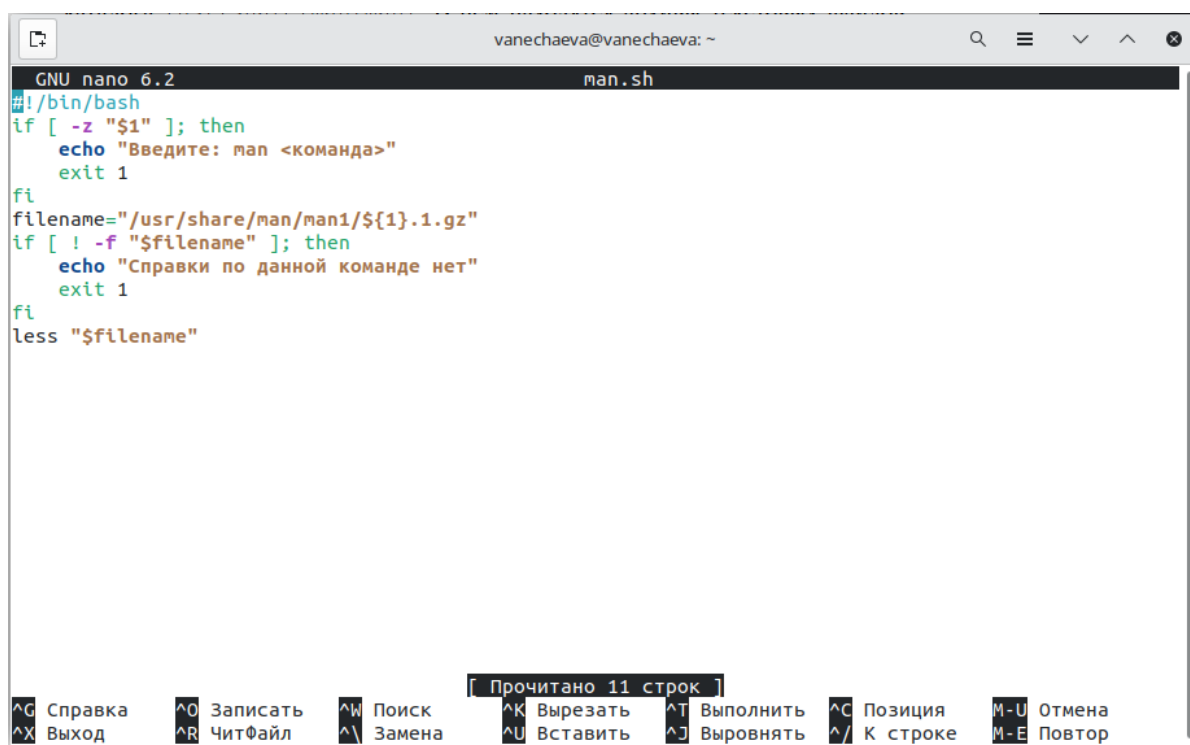


Рис. 3: Рисунок 3

## Задача 2

Сначала проверяется условие, что аргумент присутствует в строке вызова скрипта. Далее значение аргумента помещается в путь переменной filename и выводится в терминале в случае, если такая команда описана. Если нет – сообщение об этом выводится.



```
GNU nano 6.2                                man.sh
#!/bin/bash
if [ -z "$1" ]; then
    echo "Введите: man <команда>"
    exit 1
fi
filename="/usr/share/man/man1/${1}.1.gz"
if [ ! -f "$filename" ]; then
    echo "Справки по данной команде нет"
    exit 1
fi
less "$filename"
```

Прочитано 11 строк

<b>^G</b> Справка	<b>^O</b> Записать	<b>^W</b> Поиск	<b>^K</b> Вырезать	<b>^T</b> Выполнить	<b>^C</b> Позиция	<b>M-U</b> Отмена
<b>^X</b> Выход	<b>^R</b> ЧитФайл	<b>^L</b> Замена	<b>^U</b> Вставить	<b>^J</b> Выровнять	<b>^/_</b> К строке	<b>M-E</b> Повтор

Рис. 4: Рисунок 4

```
vanechaeva@vanechaeva: ~
.\\" DO NOT MODIFY THIS FILE! It was generated by help2man 1.47.3.
.TH PWD "1" "February 2022" "GNU coreutils 8.32" "User Commands"
.SH NAME
pwd \- print name of current/working directory
.SH SYNOPSIS
.B pwd
[\\FI\\,OPTION\\|\\fR]...
.SH DESCRIPTION
.\" Add any additional description here
.PP
Print the full filename of the current working directory.
.TP
\\fB\\-L\\fR, \\fB\\-\\-logical\\fR
use PWD from environment, even if it contains symlinks
.TP
\\fB\\-P\\fR, \\fB\\-\\-physical\\fR
avoid all symlinks
.TP
\\fB\\-\\-help\\fR
display this help and exit
.TP
\\fB\\-\\-version\\fR
output version information and exit
.PP
If no option is specified, \\fB\\-P\\fR is assumed.
.PP
NOTE: your shell may have its own version of pwd, which usually supersedes
the version described here. Please refer to your shell's documentation
for details about the options it supports.
.SH AUTHOR
Written by Jim Meyering.
.SH "REPORTING BUGS"
GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
.br
Report any translation bugs to <https://translationproject.org/team/>
.SH COPYRIGHT
Copyright \\(co 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
.br
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
.SH "SEE ALSO"
getcwd(3)
.PP
.br
:
```

Рис. 5: Рисунок 5

## Задача 3

В функции задана генерация одной случайной буквы из 26-буквенного латинского алфавита. RANDOM генерирует число от 1 до 26, letter переводит число в букву при помощи вырезания из последовательности `echo{a..z}` одной из букв согласно сгенерированному числу.

В `count` задается случайная длина слова от 1 до 30 символов, в цикле далее согласно длине `count` генерируется каждая буква и в финале слово печатается в терминале.

```
GNU nano 6.2                                rand.sh
#!/bin/bash
function random_letter() {
  rand=$((RANDOM%26))
  letter=$(echo {a..z} | cut -d ' ' -f $((rand+1)))
  echo $letter
}
count=$((RANDOM%30))
for (( i=1;i<=count;i++ ));
do
  echo -n $(random_letter)
done
echo ""
```

[ Прочитано 12 строк ]

<b>^G</b> Справка	<b>^O</b> Записать	<b>^W</b> Поиск	<b>^K</b> Вырезать	<b>^T</b> Выполнить	<b>^C</b> Позиция	<b>M-U</b> Отмена
<b>^X</b> Выход	<b>^R</b> ЧитФайл	<b>^_\</b> Замена	<b>^U</b> Вставить	<b>^J</b> Выровнять	<b>^/</b> К строке	<b>M-E</b> Повтор

Рис. 6: Рисунок 6

```
vanechaeva@vanechaeva: ~  
f  
g  
d  
vanechaeva@vanechaeva:~$ nano rand.sh  
vanechaeva@vanechaeva:~$ ./rand.sh  
sgsicxgxdoykvanechaeva@vanechaeva:~$ nano rand.sh  
vanechaeva@vanechaeva:~$ ./rand.sh  
jfie  
vanechaeva@vanechaeva:~$ ./rand.sh  
chigsjy  
vanechaeva@vanechaeva:~$ ./rand.sh  
avduxhuvdknxupyhyvhhbljyacbbk  
vanechaeva@vanechaeva:~$ ./rand.sh  
dyv  
vanechaeva@vanechaeva:~$ ./rand.sh  
rwntqpdxcdfbinicsllpl  
vanechaeva@vanechaeva:~$ ./rand.sh  
wqphmjmbcxzswnzjt  
vanechaeva@vanechaeva:~$ ./rand.sh  
bgaxt  
vanechaeva@vanechaeva:~$ ./rand.sh  
izsozrygjmwr  
vanechaeva@vanechaeva:~$ ./rand.sh  
air  
vanechaeva@vanechaeva:~$ ./rand.sh  
lfzhjyrm  
vanechaeva@vanechaeva:~$ ./rand.sh  
ycpqntmkcwtqzajp  
vanechaeva@vanechaeva:~$
```

Рис. 7: Рисунок 7

## Выводы

В ходе данной работы мною были изучены основы программирования в оболочке ОС UNIX и я научилась писать более сложные файлы с использованием логических управляющих конструкций и циклов.

# Контрольные вопросы

1. Найдите синтаксическую ошибку в следующей строке: `while [ $ 1 != "exit"]`

Синтаксическая ошибка в данной строке заключается в том, что вместо квадратных скобок `[ ]` необходимо использовать круглые скобки `(( ))`.

2. Как объединить (конкатенация) несколько строк в одну?

Для объединения (конкатенации) нескольких строк в одну можно использовать оператор `+=`.

3. Найдите информацию об утилите `seq`. Какими иными способами можно реализовать её функционал при программировании на `bash`?

`seq` - это утилита командной строки, которая генерирует последовательность чисел. Она может использоваться для создания циклов и других задач, которые требуют генерации последовательностей чисел. Пример использования:

4. Какой результат даст вычисление выражения `$((10/3))`?

Результатом вычисления выражения `$((10/3))` будет целое число 3. При делении целых чисел результат также является целым числом, и любая дробная часть отбрасывается.

5. Укажите кратко основные отличия командной оболочки `zsh` от `bash`.

Некоторые отличия командной оболочки `zsh` от `bash`:

zsh имеет более продвинутую систему автодополнения и подсказок в командной строке. zsh поддерживает более широкий диапазон символов при именовании файлов и переменных. zsh имеет более продвинутую систему расширения команд и возможности настройки.

6. Проверьте, верен ли синтаксис данной конструкции `for ((a=1; a <= LIMIT; a++))`

Синтаксис данной конструкции корректен, если переменная `LIMIT` заранее определена. Если `LIMIT` не определена, то при выполнении скрипта будет возникать ошибка.

7. Сравните язык `bash` с какими-либо языками программирования. Какие преимущества у `bash` по сравнению с ними? Какие недостатки?

Bash является языком командной строки, который используется для автоматизации и автоматизации выполнения задач в операционной системе Linux. В отличие от некоторых языков программирования, таких как C++ или Java, Bash имеет более простой синтаксис и специализ