

# **Отчёт по лабораторной работе №12**

**Программирование в командном процессоре ОС UNIX. Расширенное  
программирование**

Федюшина Ярослава Андреевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>4</b>	<b>Выводы</b>	<b>17</b>
<b>5</b>	<b>Клнтрольные вопросы</b>	<b>18</b>

## Список иллюстраций

3.1	создание файла . . . . .	8
3.2	код . . . . .	9
3.3	проверка программы . . . . .	10
3.4	создание нового файла . . . . .	10
3.5	код для map . . . . .	11
3.6	проверка кода . . . . .	12
3.7	map less . . . . .	13
3.8	создание нового файла . . . . .	14
3.9	код random . . . . .	15
3.10	проверка кода с random . . . . .	16

## **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 Задание

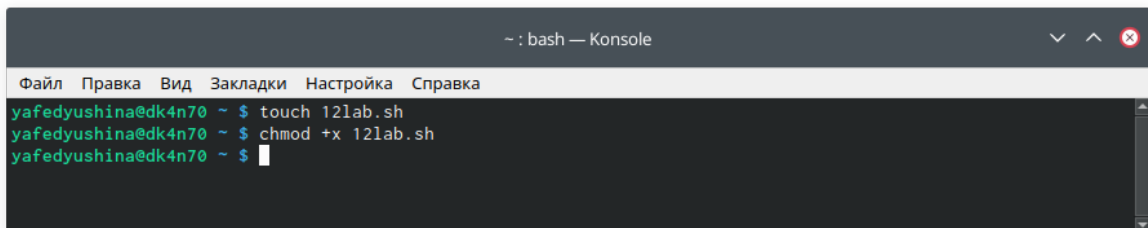
1. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени  $t_1$  дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени  $t_2 < t_1$ , также выдавая информацию 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 в 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 нём 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 псевдо случайные числа в диапазоне от

0 до 32767.

## 3 Выполнение лабораторной работы

Создаю файл для выполнения работы и написания кода

### 3.1



```
~ : bash — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
yafedyushina@dk4n70 ~ $ touch 12lab.sh
yafedyushina@dk4n70 ~ $ chmod +x 12lab.sh
yafedyushina@dk4n70 ~ $
```

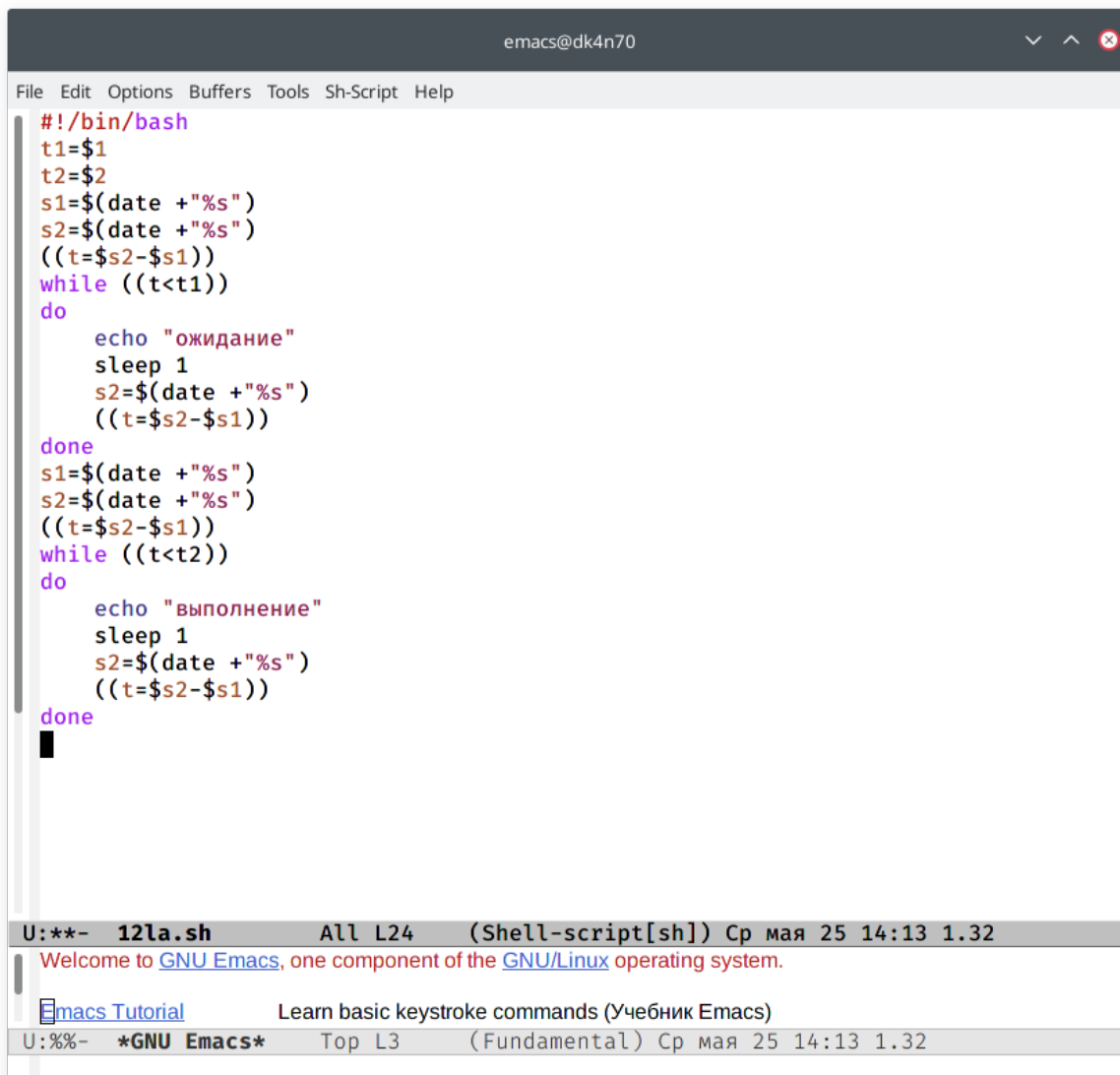
Рис. 3.1: создание файла

### 3.2

написание кода по заданию в emacs



### 3.3



The screenshot shows an Emacs window titled 'emacs@dk4n70'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The main buffer contains a shell script with the following content:

```
#!/bin/bash
t1=$1
t2=$2
s1=$(date +%S")
s2=$(date +%S")
((t=$s2-$s1))
while ((t<t1))
do
    echo "ожидание"
    sleep 1
    s2=$(date +%S")
    ((t=$s2-$s1))
done
s1=$(date +%S")
s2=$(date +%S")
((t=$s2-$s1))
while ((t<t2))
do
    echo "выполнение"
    sleep 1
    s2=$(date +%S")
    ((t=$s2-$s1))
done
```

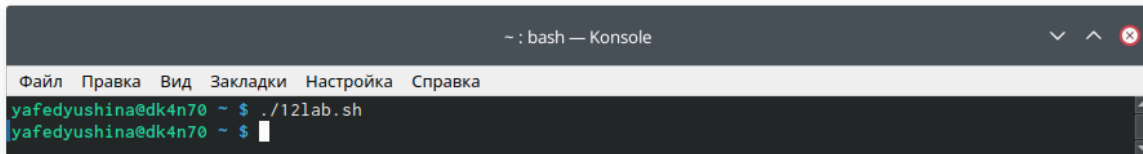
Below the script, there is a terminal buffer showing the command prompt 'U: \*\*~ 12la.sh' and the output 'Welcome to GNU Emacs, one component of the GNU/Linux operating system.' and 'Learn basic keystroke commands (Учебник Emacs)'. The terminal buffer also shows the command prompt 'U: %%~ \*GNU Emacs\*' and the output 'Top L3 (Fundamental) Cp мая 25 14:13 1.32'.

Рис. 3.2: код

### 3.4

проверяю написанную мной программу

## 3.5



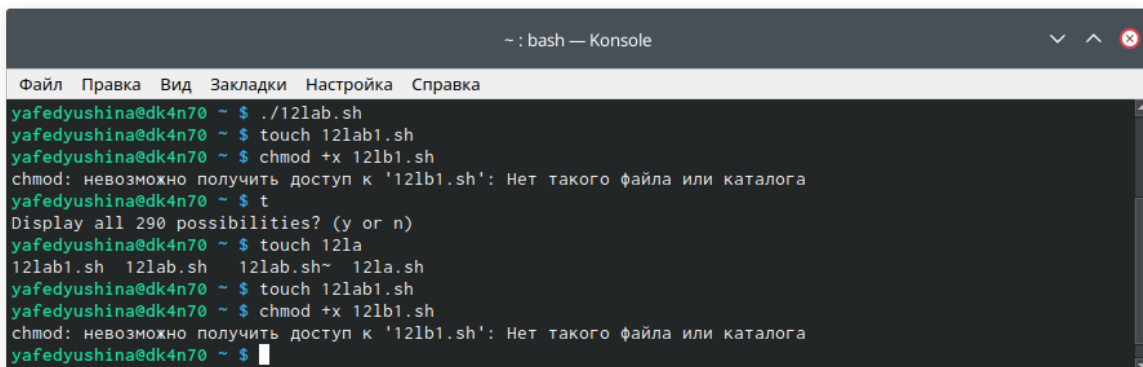
```
~ : bash — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
yafedyushina@dk4n70 ~ $ ./12lab.sh
yafedyushina@dk4n70 ~ $
```

Рис. 3.3: проверка программы

## 3.6

создаю новый файл для реализации команды map

## 3.7



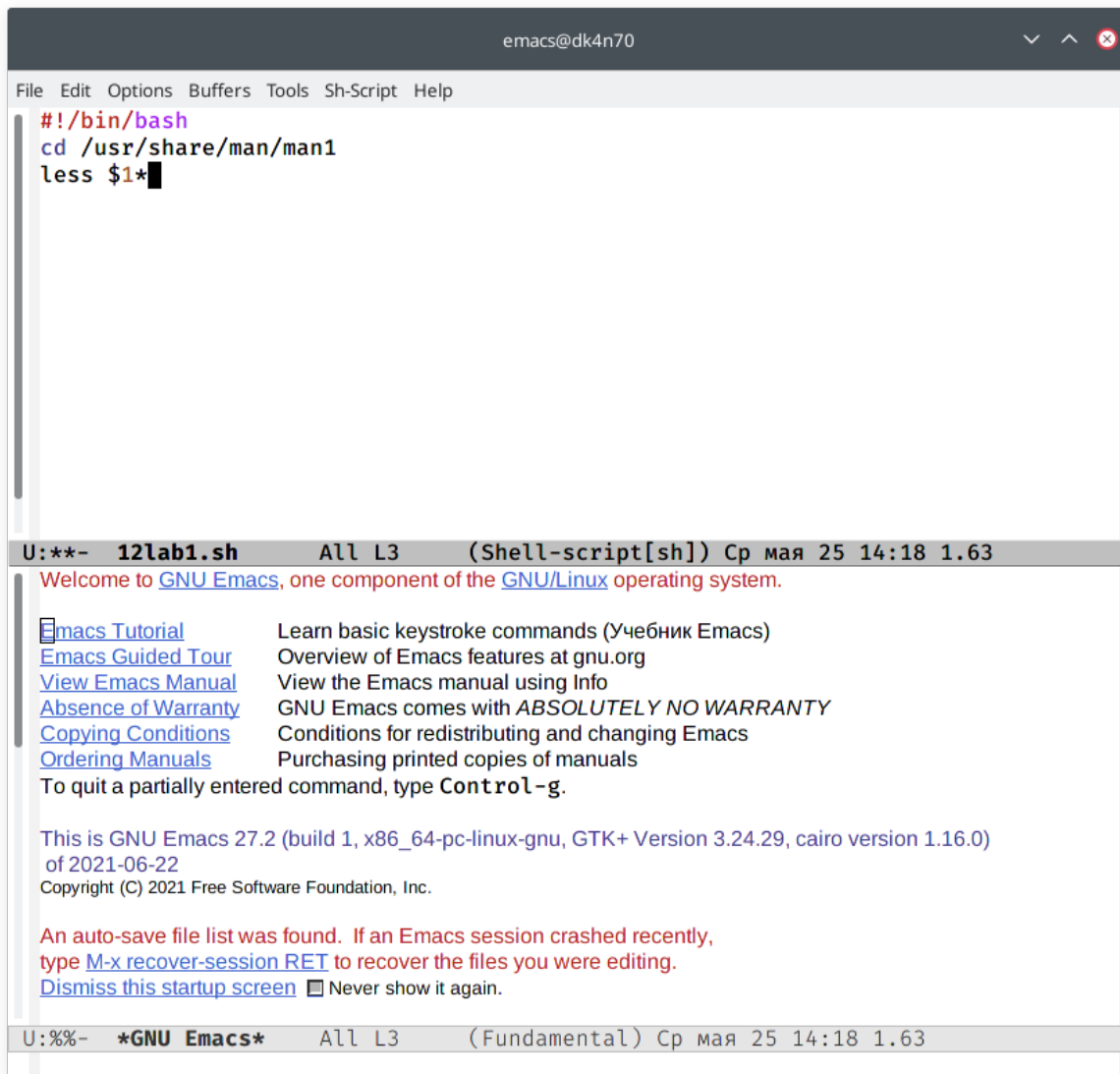
```
~ : bash — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
yafedyushina@dk4n70 ~ $ ./12lab.sh
yafedyushina@dk4n70 ~ $ touch 12lab1.sh
yafedyushina@dk4n70 ~ $ chmod +x 12lab1.sh
chmod: невозможно получить доступ к '12lab1.sh': Нет такого файла или каталога
yafedyushina@dk4n70 ~ $ t
Display all 290 possibilities? (y or n)
yafedyushina@dk4n70 ~ $ touch 12la
12lab1.sh 12lab.sh 12lab.sh~ 12la.sh
yafedyushina@dk4n70 ~ $ touch 12lab1.sh
yafedyushina@dk4n70 ~ $ chmod +x 12lab1.sh
chmod: невозможно получить доступ к '12lab1.sh': Нет такого файла или каталога
yafedyushina@dk4n70 ~ $
```

Рис. 3.4: создание нового файла

## 3.8

пишу сам код для реализации задумки

## 3.9



The screenshot shows the Emacs editor window titled "emacs@dk4n70". The menu bar includes "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". The terminal session shows the following commands and output:

```
#!/bin/bash
cd /usr/share/man/man1
less $1*
```

The status bar at the bottom of the terminal session reads: "U:\*\*~ 12lab1.sh All L3 (Shell-script[sh]) Ср мая 25 14:18 1.63".

The Emacs startup screen displays the following text:

Welcome to [GNU Emacs](#), one component of the [GNU/Linux](#) operating system.

<a href="#">Emacs Tutorial</a>	Learn basic keystroke commands (Учебник Emacs)
<a href="#">Emacs Guided Tour</a>	Overview of Emacs features at gnu.org
<a href="#">View Emacs Manual</a>	View the Emacs manual using Info
<a href="#">Absence of Warranty</a>	GNU Emacs comes with <i>ABSOLUTELY NO WARRANTY</i>
<a href="#">Copying Conditions</a>	Conditions for redistributing and changing Emacs
<a href="#">Ordering Manuals</a>	Purchasing printed copies of manuals

To quit a partially entered command, type **Control-g**.

This is GNU Emacs 27.2 (build 1, x86\_64-pc-linux-gnu, GTK+ Version 3.24.29, cairo version 1.16.0) of 2021-06-22  
Copyright (C) 2021 Free Software Foundation, Inc.

An auto-save file list was found. If an Emacs session crashed recently, type **M-x recover-session RET** to recover the files you were editing.  
[Dismiss this startup screen](#) ☐ Never show it again.

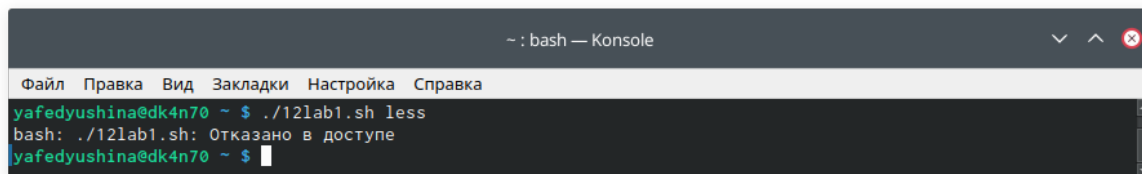
The status bar at the bottom of the Emacs window reads: "U:%%- \*GNU Emacs\* All L3 (Fundamental) Ср мая 25 14:18 1.63".

Рис. 3.5: код для man

## 3.10

проверяю новую программу на работу

## 3.11

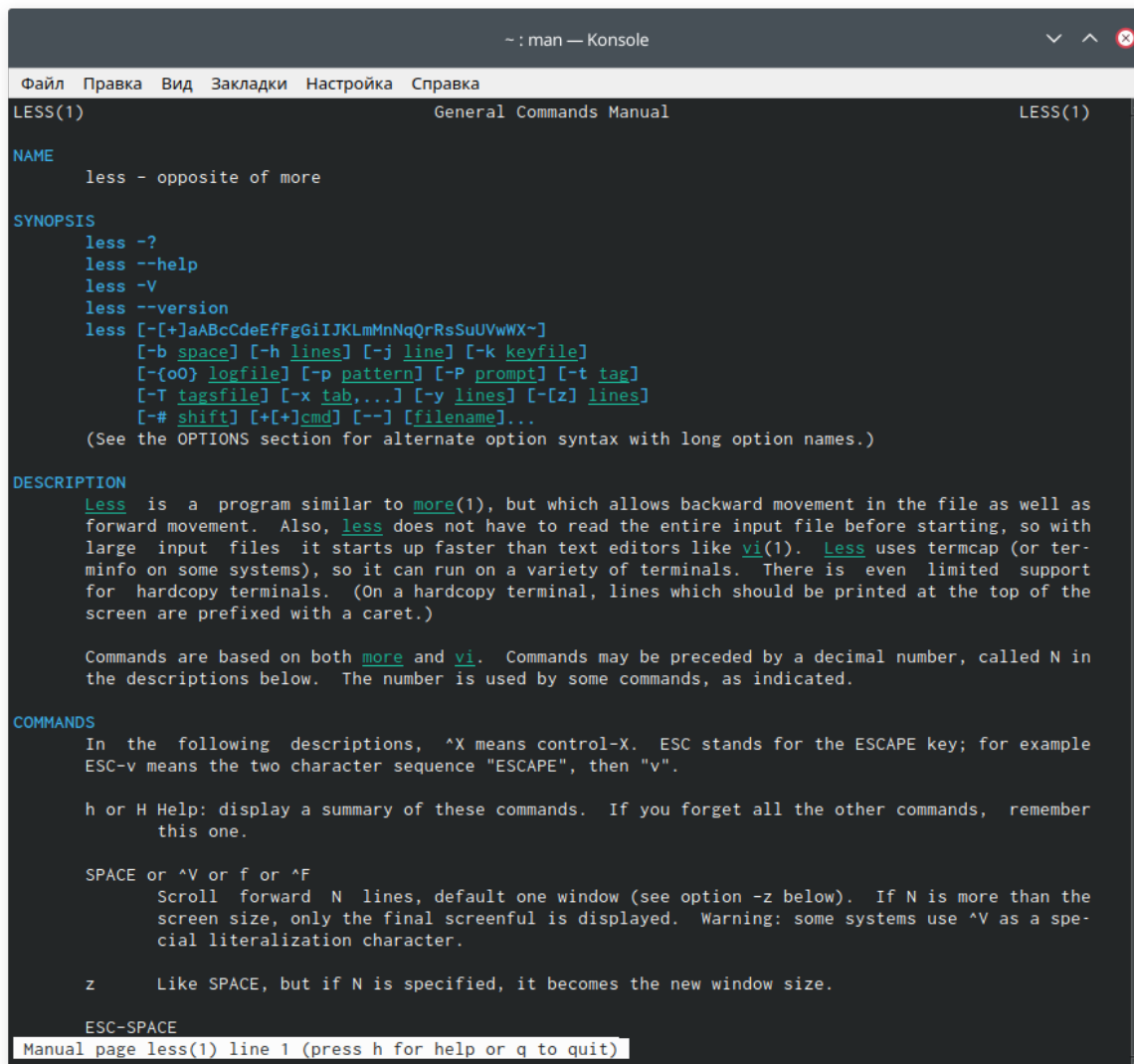


The screenshot shows a terminal window titled '~ : bash — Konsole'. The window has a menu bar with 'Файл', 'Правка', 'Вид', 'Закладки', 'Настройка', and 'Справка'. The terminal content shows a user 'yafedyushina@dk4n70' in the home directory (~) running the command './121lab1.sh less'. The system responds with 'bash: ./121lab1.sh: Отказано в доступе' (permission denied). The prompt returns to the user, and a cursor is visible at the end of the line.

```
~ : bash — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
yafedyushina@dk4n70 ~ $ ./121lab1.sh less
bash: ./121lab1.sh: Отказано в доступе
yafedyushina@dk4n70 ~ $
```

Рис. 3.6: проверка кода

## 3.12



```
~ : man — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
LESS(1)                                     General Commands Manual  LESS(1)

NAME
    less - opposite of more

SYNOPSIS
    less -?
    less --help
    less -V
    less --version
    less [-[+]aABcCdeEfFgGiIJKLmMnNqQrRsSuUVvWwX~]
        [-b space] [-h lines] [-j line] [-k keyfile]
        [-{oO} logfile] [-p pattern] [-P prompt] [-t tag]
        [-T tagsfile] [-x tab,...] [-y lines] [-z] lines
        [-# shift] [+{+}cmd] [--] [filename]...
    (See the OPTIONS section for alternate option syntax with long option names.)

DESCRIPTION
    Less is a program similar to more(1), but which allows backward movement in the
    file as well as forward movement. Also, less does not have to read the entire
    input file before starting, so with large input files it starts up faster than
    text editors like vi(1). Less uses termcap (or terminfo on some systems), so
    it can run on a variety of terminals. There is even limited support for
    hardcopy terminals. (On a hardcopy terminal, lines which should be printed
    at the top of the screen are prefixed with a caret.)

    Commands are based on both more and vi. Commands may be preceded by a decimal
    number, called N in the descriptions below. The number is used by some
    commands, as indicated.

COMMANDS
    In the following descriptions, ^X means control-X. ESC stands for the
    ESCAPE key; for example ESC-v means the two character sequence "ESCAPE",
    then "v".

    h or H Help: display a summary of these commands. If you forget all the
    other commands, remember this one.

    SPACE or ^V or f or ^F
        Scroll forward N lines, default one window (see option -z below). If
        N is more than the screen size, only the final screenful is displayed.
        Warning: some systems use ^V as a special literalization character.

    z
        Like SPACE, but if N is specified, it becomes the new window size.

    ESC-SPACE
    Manual page less(1) line 1 (press h for help or q to quit)
```

Рис. 3.7: man less

## 3.13

создаю новый текстовый файл и делаю его исполняемым

## 3.14

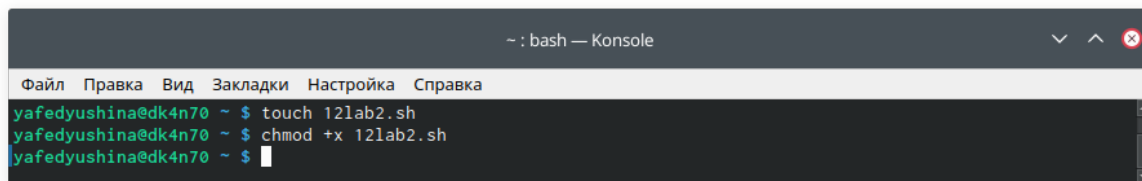


Рис. 3.8: создание нового файла

## 3.15

пишу код, генерирующий случайную последовательность букв латинского алфавита

## 3.16



The screenshot shows an Emacs window titled 'emacs@dk4n70'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The main text area contains a shell script in Bash. The script sets variables M=10, c=1, and d=1, then echoes '10 случайных слов'. It enters a while loop that continues as long as c is not equal to M+1. Inside the loop, it uses a for loop to iterate from i=1 to i=10, printing a random character from the set [0-9a-z] using printf. It then increments c and d. The script ends with a cursor on a new line.

```
#!/bin/bash
M=10
c=1
d=1
echo
echo "10 случайных слов"
while ((c!=($M+1)))
do
    echo $(for((i=1; i<=10; i++));
    do
        printf '%s' "${RANDOM:0:1}";
    done
    tr '0-9' '[a-z]'
    echo $d
    ((c+=1))
    ((d+=1))
done
█
```

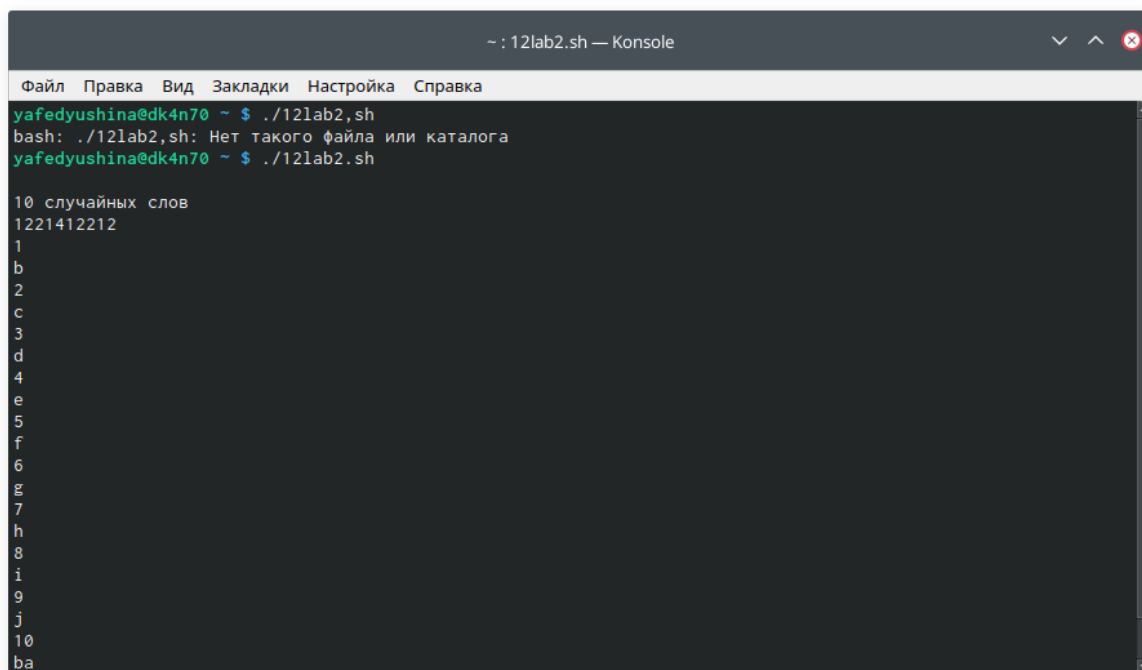
The status bar at the bottom shows the current buffer is '12lab2.sh', line 18, column 18, in a shell script. It also displays the date and time: 'Ср мая 25 14:25 0.72'. Below the status bar, there is a message: 'Welcome to GNU Emacs, one component of the GNU/Linux operating system.' and a link to 'Emacs Tutorial' with the text 'Learn basic keystroke commands (Учебник Emacs)'. The bottom-most status bar shows the Emacs version: '\*GNU Emacs\*', the current buffer: 'Top L3', and the date and time: '(Fundamental) Ср мая 25 14:25 0.72'.

Рис. 3.9: код random

## 3.17

проверяю программу на работу

## 3.18



```
~ : 12lab2.sh — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
yafedyushina@dk4n70 ~ $ ./12lab2.sh
bash: ./12lab2.sh: Нет такого файла или каталога
yafedyushina@dk4n70 ~ $ ./12lab2.sh

10 случайных слов
1221412212
1
b
2
c
3
d
4
e
5
f
6
g
7
h
8
i
9
j
10
ba
```

Рис. 3.10: проверка кода с random

## 3.19



## 4 Выводы

В ходе выполнения лабораторной работы №12 я изучила основы программирования в оболочке ОС Linux, а так же научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов

## 5 Кнтрольные вопросы

1. `while [$1 != "exit"]` В данной строчке допущены следующие ошибки:

- не хватает пробелов после первой скобки [ и перед второй скобкой ]
- выражение `$1` необходимо взять в `"`, потому что эта переменная может содержать пробелы. Таким образом, правильный вариант должен выглядеть так: `while [ "$1" != "exit" ]`

2. Чтобы объединить несколько строк в одну, можно воспользоваться несколькими способами:

- Первый: `VAR1="Hello," VAR2=" World" VAR3="VAR1VAR2" echo "$VAR3"`  
Результат: Hello, World
- Второй: `VAR1="Hello," VAR1+= " World" echo "$VAR1"` Результат: Hello, World

3. Команда `seq` в Linux используется для генерации чисел от ПЕРВОГО до ПОСЛЕДНЕГО шага INCREMENT. Параметры:

- `seq LAST`: если задан только один аргумент, он создает числа от 1 до LAST с шагом шага, равным 1. Если LAST меньше 1, значение не выдает.
- `seq FIRST LAST`: когда заданы два аргумента, он генерирует числа от FIRST до LAST с шагом 1, равным 1. Если LAST меньше FIRST, он не выдает никаких выходных данных.
- `seq FIRST INCREMENT LAST`: когда заданы три аргумента, он генерирует числа от FIRST до LAST на шаге INCREMENT. Если LAST меньше, чем FIRST, он не производит вывод.

- `seq -f «FORMAT» FIRST INCREMENT LAST`: эта команда используется для генерации последовательности в форматированном виде. `FIRST` и `INCREMENT` являются необязательными.
  - `seq -s «STRING» ПЕРВЫЙ ВКЛЮЧЕНО`: Эта команда используется для `STRING` для разделения чисел. По умолчанию это значение равно `/n`. `FIRST` и `INCREMENT` являются необязательными.
  - `seq -w FIRST INCREMENT LAST`: эта команда используется для выравнивания ширины путем заполнения начальными нулями. `FIRST` и `INCREMENT` являются необязательными.
4. Результатом данного выражения `$((10/3))` будет 3, потому что это целочисленное деление без остатка.
  5. Отличия командной оболочки `zsh` от `bash`:
    - В `zsh` более быстрое автодополнение для `cd` с помощью `Tab`
    - В `zsh` существует калькулятор `zcalc`, способный выполнять вычисления внутри терминала
    - В `zsh` поддерживаются числа с плавающей запятой
    - В `zsh` поддерживаются структуры данных «хэш»
    - В `zsh` поддерживается раскрытие полного пути на основе неполных данных
    - В `zsh` поддерживается замена части пути
    - В `zsh` есть возможность отображать разделенный экран, такой же как разделенный экран `vim`
  6. `for ((a=1; a <= LIMIT; a++))` синтаксис данной конструкции верен, потому что, используя двойные круглые скобки, можно не писать `$` перед переменными `()`.
  7. Преимущества скриптового языка `bash`:
    - Один из самых распространенных и ставится по умолчанию в большинстве дистрибутивах Linux, MacOS

- Удобное перенаправление ввода/вывода
- Большое количество команд для работы с файловыми системами Linux
- Можно писать собственные скрипты, упрощающие работу в Linux
- Недостатки скриптового языка bash:
- Дополнительные библиотеки других языков позволяют выполнить больше действий
- Bash не является языком общего назначения
- Утилиты, при выполнении скрипта, запускают свои процессы, которые, в свою очередь, отражаются на скорости выполнения этого скрипта
- Скрипты, написанные на bash, нельзя запустить на других операционных системах без дополнительных действий