

# 1. Лабораторная работа №1. Основы интерфейса командной строки ОС GNU Linux

## 1.1. Цель работы

Приобретение практических навыков работы с операционной системой на уровне командной строки (организация файловой системы, навигация по файловой системе, создание и удаление файлов и директорий).

## 1.2. Теоретическое введение

### 1.2.1. Введение в GNU Linux

*Операционная система (ОС)* — это комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем. Сегодня наиболее известными операционными системами являются ОС семейства Microsoft Windows и UNIX-подобные системы.

*GNU Linux* — семейство переносимых, многозадачных и многопользовательских операционных систем, на базе ядра Linux, включающих тот или иной набор утилит и программ проекта GNU, и, возможно, другие компоненты. Как и ядро Linux, системы на его основе, как правило, создаются и распространяются в соответствии с моделью разработки свободного и открытого программного обеспечения (Open-Source Software). Linux-системы распространяются в основном бесплатно в виде различных дистрибутивов.

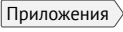
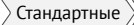


*Дистрибутив GNU Linux* — общее определение ОС, использующих ядро Linux и набор библиотек и утилит, выпускаемых в рамках проекта GNU, а также графическую оконную подсистему X Window System. Дистрибутив готов для конечной установки на пользовательское оборудование. Кроме ядра и, собственно, операционной системы дистрибутивы обычно содержат широкий набор приложений, таких как редакторы документов и таблиц, мультимедийные проигрыватели, системы для работы с базами данных и т.д. Существуют дистрибутивы, разрабатываемые как при коммерческой поддержке (Red Hat / Fedora, SLED / OpenSUSE, Ubuntu), так и исключительно усилиями добровольцев (Debian, Slackware, Gentoo, ArchLinux).

### 1.2.2. Введение в командную строку GNU Linux

Работу ОС GNU Linux можно представить в виде функционирования множества взаимосвязанных процессов. При загрузке системы сначала запускается ядро, которое, в свою очередь,

запускает оболочку ОС (от англ. shell «оболочка»). Взаимодействие пользователя с системой Linux (работа с данными и управление работающими в системе процессами) происходит в интерактивном режиме посредством командного языка. Оболочка операционной системы (или командная оболочка, интерпретатор команд) — интерпретирует (т.е. переводит на машинный язык) вводимые пользователем команды, запускает соответствующие программы (процессы), формирует и выводит ответные сообщения. Кроме того, на языке командной оболочки можно писать небольшие программы для выполнения ряда последовательных операций с файлами и содержащимися в них данными — сценарии (скрипты).

Из командных оболочек GNU Linux наиболее популярны `bash`, `csh`, `ksh`, `zsh`. Команда `echo $SHELL` позволяет проверить, какая оболочка используется. В качестве предустановленной командной оболочки GNU Linux используется одна из наиболее распространённых разновидностей командной оболочки — `bash` (Bourne again shell).

В GNU Linux доступ пользователя к командной оболочке обеспечивается через терминал (или консоль). Запуск терминала можно осуществить через главное меню    или нажав .

Интерфейс командной оболочки очень прост. Обычно он состоит из приглашения командной строки (строки, оканчивающейся символом `$`), по которому пользователь вводит команды:


```
iivanova@dk4n31:~$
```



Это приглашение командной оболочки, которое несёт в себе информацию об имени пользователя `iivanova`, имени компьютера `dk4n31` и текущем каталоге, в котором находится пользователь, в данном случае это домашний каталог пользователя, обозначенный как `~`.

Команды могут быть использованы с ключами (или опциями) — указаниями, модифицирующими поведение команды. Ключи обычно начинаются с символа `(-)` или `(--)` и часто состоят из одной буквы. Кроме ключей после команды могут быть использованы аргументы (параметры) — названия объектов, для которых нужно выполнить команду (например, имена файлов и каталогов). Например, для подробного просмотра содержимого каталога `documents` может быть использована команда `ls` с ключом `-l`:

```
iivanova@dk4n31:~$ ls -l documents
```

В данном случае `ls` — это имя команды, `l` — ключ, `documents` — аргумент. Команды, ключи и аргументы должны быть отделены друг от друга пробелом.

Ввод команды завершается нажатием клавиши , после чего команда передаётся оболочке на исполнение. Результатом выполнения команды могут являться сообщения о ходе выполнения команды или об ошибках. Появление приглашения командной строки говорит о том, что выполнение команды завершено.

Иногда в GNU Linux имена программ и команд слишком длинные, однако `bash` может завершать имена при их вводе в терминале. Нажав клавишу , можно завершить имя команды, программы или каталога. Например, предположим, что нужно использовать программу `mcedit`. Для этого наберите в командной строке `mc`, затем нажмите один раз клавишу . Если ничего не происходит, то это означает, что существует несколько возможных

вариантов завершения команды. Нажав клавишу `Tab` ещё раз, можно получить список имён, начинающихся с `mc`:

```
iivanova@dk4n31:~$ mc
mc      mcd      mcedit   mclasser mcookie  mcview
mcat    mcdiff   mcheck   mcomp    mcopy
iivanova@dk4n31:~$ mc
```

Более подробно о работе в операционной системе Linux см., например, в [13; 16].

1.2.3. Файловая структура GNU Linux: каталоги и файлы

Файловая система определяет способ организации, хранения и именования данных на носителях информации в компьютерах и представляет собой иерархическую структуру в виде вложенных друг в друга каталогов (директорий), содержащих все файлы. В ОС Linux каталог, который является “вершиной” файловой системы, называется **корневым каталогом**, обозначается символом `/` и содержит все остальные каталоги и файлы.

В большинстве Linux-систем поддерживается стандарт иерархии файловой системы (Filesystem Hierarchy Standard, FHS), унифицирующий местонахождение файлов и каталогов. Это означает, что в корневом каталоге находятся только подкаталоги со стандартными именами и типами данных, которые могут попасть в тот или иной каталог. Так, в любой Linux-системе всегда есть каталоги `/etc`, `/home`, `/usr/bin` и т.п. В табл. 1.1 приведено краткое описание нескольких каталогов.

Таблица 1.1. Описание некоторых каталогов файловой системы GNU Linux

Каталог	Описание
<code>/</code>	Корневая директория, содержащая всю файловую
<code>/bin</code>	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям (например: <code>cat</code> , <code>ls</code> , <code>cp</code> )
<code>/etc</code>	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
<code>/home</code>	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
<code>/media</code>	Точки монтирования для сменных носителей, таких как CD-ROM, DVD-ROM, flash
<code>/root</code>	Домашняя директория пользователя <code>root</code>
<code>/tmp</code>	Временные файлы

Каталог	Описание
/usr	Вторичная иерархия для данных пользователя; содержит большинство пользовательских приложений и утилит, используемых в многопользовательском режиме; может быть смонтирована по сети только для чтения и быть общей для нескольких машин

Обратиться к файлу, расположенному в каком-то каталоге, можно указав путь к нему. Существует несколько видов путей к файлу:

- **полный или абсолютный путь** — начинается от корня (/), образуется перечислением всех каталогов, разделённых прямым слешем (/), и завершается именем файла (например, полный путь к файлу `addition.txt` из каталога `user` в каталоге `home`, находящемся в корневом каталоге, будет иметь вид: `/home/user/documents/addition.txt`;
- **относительный путь** — так же как и полный путь, строится перечислением через (/) всех каталогов, но начинается от текущего каталога (каталога, в котором “находится” пользователь), т.е. пользователь, находясь в каталоге `user`, может обратиться к файлу `addition.txt`, указав относительный путь `documents/addition.txt`.

Таким образом, в Linux если имя объекта начинается с /, то системой это интерпретируется как полный путь, в любом другом случае — как относительный.

В Linux любой пользователь имеет **домашний каталог**, который, как правило, имеет имя пользователя. В домашних каталогах хранятся документы и настройки пользователя. Для обозначения домашнего каталога используется знак тильды (~). При переходе из домашнего каталога знак тильды будет заменён на имя нового текущего каталога.

#### 1.2.4. Базовые команды `bash`

В операционной системе GNU Linux взаимодействие пользователя с системой обычно осуществляется с помощью командной строки посредством построчного ввода команд. Общий формат команд можно представить следующим образом:

```
<имя_команды><разделитель><аргументы>
```

Первые задачи, которые приходится решать в любой системе это — работа с данными (обычно хранящимися в файлах) и управление работающими в системе программами (процессами). Для получения достаточно подробной информации по каждой из команд используйте команду `man`, например:

```
user@dk4n31:~$ man ls
```

В таблице 1.2 приведены основные команды взаимодействия пользователя с файловой системой в GNU Linux посредством командной строки.

**Таблица 1.2.** Основные команды взаимодействия пользователя с файловой системой

Команда		Описание
pwd	<b>P</b> rint <b>W</b> orking <b>D</b> irectory	определение текущего каталога
cd	<b>C</b> hange <b>D</b> irectory	смена каталога
ls	<b>L</b> i <b>S</b> t	вывод списка файлов
mkdir	<b>M</b> a <b>K</b> e <b>D</b> IRectory	создание пустых каталогов
touch		создание пустых файлов
rm	<b>R</b> e <b>M</b> ove	удаление файлов или каталогов
mv	<b>M</b> o <b>V</b> e	перемещение файлов и каталогов
cp	<b>C</b> o <b>P</b> y	копирование файлов и каталогов
cat		вывод содержимого файлов

Более подробно о работе в `bash` см. в [2; 5; 6; 8].

**1.2.5. Полезные комбинации клавиш**

Для удобства и экономии времени при работе в терминале существует большое количество сокращённых клавиатурных команд.

Клавиши `↑` и `↓` позволяют увидеть историю предыдущих команд в `bash`. Количество хранимых строк определено в переменной окружения `HISTSIZE`.

Клавиши `←` и `→` перемещают курсор влево и вправо в текущей строке, позволяя редактировать команды.

Сочетания клавиш `Ctrl + a` и `Ctrl + e` перемещают курсор в начало и в конец текущей строки. Клавиши `Ctrl + k` удаляет всё от текущей позиции курсора до конца строки, а `Ctrl + w` или `Alt + Backspace` удаляют слово перед курсором.

Сочетание клавиш `Ctrl + d` в пустой строке служит для завершения текущего сеанса. Для завершения выполняющейся в данный момент команды можно использовать `Ctrl + c`. Также данное сочетание отменит редактирование командной строки и вернёт приглашение командной строки. `Ctrl + l` очищает экран.

### 1.3. Порядок выполнения работы

### 1.4. Техническое обеспечение

Лабораторная работа подразумевает работу с операционной системой ОС Linux на уровне командной строки. Выполнение работы возможно как в дисплейном классе факультета физико-математических и естественных наук РУДН, так и дома. Описание выполнения работы приведено для дисплейного класса со следующими характеристиками техники:

- Intel Core i3-550 3.2 GHz, 4 GB оперативной памяти, 8 GB свободного места на жёстком диске;
- ОС Linux Gentoo (<http://www.gentoo.ru/>).

#### 1.4.1. Перемещение по файловой системе

Откройте терминал. По умолчанию терминал открывается в домашнем каталоге пользователя, который обозначается символом `~`.

```
user@dk4n31:~$
```

Убедитесь, что Вы находитесь в домашнем каталоге. Если это не так, перейдите в него. Это можно сделать с помощью команды `cd` без аргументов.

```
user@dk4n31: /tmp$ cd
user@dk4n31: ~$
```

С помощью команды `pwd` узнайте полный путь к Вашему домашнему каталогу.

```
user@dk4n31: ~$ pwd
```

Команда `cd` позволяет сменить текущий каталог на другой, указав путь к нему в качестве параметра. Формат команды:

```
cd [путь_к_каталогу]
```

Команда `cd` работает как с абсолютными, так и с относительными путями.

Перейдите в подкаталог Документы Вашего домашнего каталога указав относительный путь

```
user@dk4n31: ~$ cd Документы
user@dk4n31: ~/Документы$
```

Перейдите в каталог `local` – подкаталог `usr` корневого каталога указав абсолютный путь к нему (`/usr/local`):

```
user@dk4n31: ~$ cd /usr/local
user@dk4n31: ~/usr/local$
```

Обратите внимание абсолютный путь **всегда** начинается от корневого каталога (т.е. с символа /).

Можно использовать комбинацию 'cd -' для возвращения в последний посещённый пользователем каталог. А 'cd ..' используется для перехода на один каталог выше по иерархии. Введите последовательно эти команды. В каком каталоге Вы находитесь?

Команда ls выдаёт список файлов указанного каталога и имеет следующий синтаксис:

```
ls [опции] [каталог] [каталог...]
```

Для просмотра списка файлов текущего каталога может быть использована команда ls без аргументов.

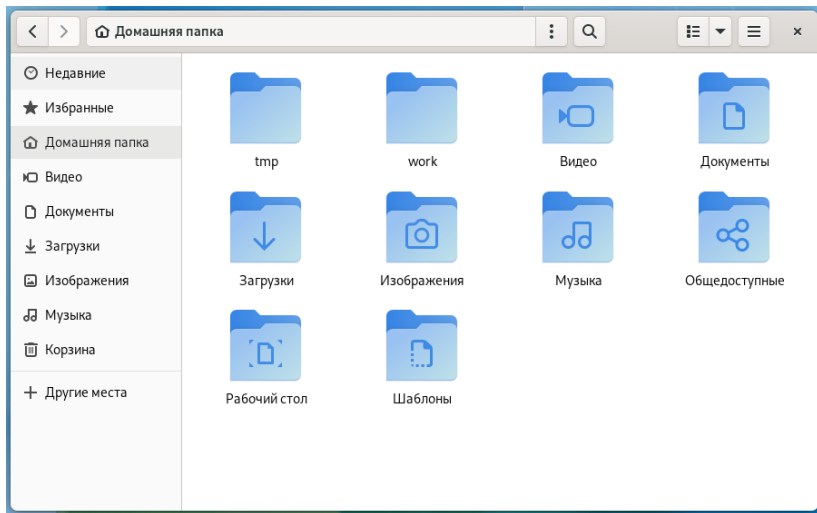
Перейдите в домашний каталог

```
user@dk4n31:~$ cd ~
```

Выведите список файлов Вашего домашнего каталога.

```
user@dk4n31:~$ ls
```

Откройте домашний каталог с помощью файлового менеджера графического окружения Вашей ОС (рис. 1.1): Обзор >> Файлы >> Домашняя папка или Компьютер >> Домашняя папка или Места >> Домашняя папка или введя в терминале команду nautilus.



**Рис. 1.1.** Домашняя папка

Убедитесь в том, что список файлов полученных с помощью команды ls совпадает с файлами, отображающимися в графическом файловом менеджере.

Также как и команда `cd`, команда `ls` работает как с абсолютными, так и с относительными путями.

Выведите список файлов подкаталога Документы Вашего домашнего каталога указав относительный путь

```
user@dk4n31:~$ ls Документы
```

Выведите список файлов каталога `/usr/local` указав абсолютный путь к нему:

```
user@dk4n31:~$ ls /usr/local
```

Для данной команды существует довольно много опций (ключей), ниже дано описание некоторых из них.

Таблица 1.3. Опции команды `ls`

Ключ	Описание
-a	вывод списка всех файлов, включая скрытые файлы (в Linux названия скрытых файлов начинаются с точки)
-R	рекурсивный вывод списка файлов и подкаталогов
-h	вывод для каждого файла его размера
-l	вывод дополнительной информации о файлах (права доступа, владельцы и группы, размеры файлов и время последнего доступа)
-i	вывод уникального номера файла (inode) в файловой системе перед каждым файлом
-d	обработка каталогов, указанных в командной строке, так, как если бы они были обычными файлами, вместо вывода списка их файлов

Примеры

- команда `ls -R` рекурсивно выводит список содержимого текущего каталога;
- команда `ls -is images/ ..` выводит список файлов каталога `images` и родительского по отношению к текущему каталога, при этом для каждого файла указан номер `inode` и его размер в килобайтах;
- команда `ls -l images/*.png` выводит список всех файлов в каталоге `images`, чьи имена заканчиваются на `.png`, включая скрытый файл `.png`, если таковой существует.

Включите в отчет примеры использования команды `ls` с разными ключами.



### 1.4.2. Создание пустых каталогов и файлов

Для создания каталогов используется команда `mkdir`. Её синтаксис имеет вид:

```
mkdir [опции] <каталог> [каталог...]
```

Создайте в домашнем каталоге подкаталог с именем `parentdir`

```
user@dk4n31:~$ cd
user@dk4n31:~$ mkdir parentdir
```

С помощью команды `ls` проверьте, что каталог создан.

Создайте подкаталог в существующем каталоге:

```
user@dk4n31:~$ mkdir parentdir/dir
```

При задании нескольких аргументов создаётся несколько каталогов:

```
user@dk4n31:~$ cd parentdir
user@dk4n31:~$ mkdir dir1 dir2 dir3
```

Если требуется создать подкаталог в каталоге, отличном от текущего, то путь к нему требуется указать в явном виде:

```
user@dk4n31:~$ mkdir ~/newdir
```

Эта команда должна создать каталог `newdir` в домашнем каталоге (`~`). Проверьте это с помощью команды

```
user@dk4n31:~$ ls ~
```

Опция – `parents` (краткая форма `-p`) позволяет создавать иерархическую цепочку подкаталогов, создавая все промежуточные каталоги. Создайте следующую последовательность вложенных каталогов `newdir/dir1/dir2` в домашнем каталоге

```
user@dk4n31:~$ mkdir -p ~/newdir/dir1/dir2
```

Для создания файлов может быть использована команда `touch`, которая имеет следующий синтаксис:

```
touch [опции] файл [файл...]
```

Создайте файл `test.txt` в каталоге `~/newdir/dir1/dir2`

```
user@dk4n31:~$ touch ~/newdir/dir1/dir2/test.txt
```

Проверьте наличие файла с помощью команды

```
user@dk4n31:~$ ls ~/newdir/dir1/dir2
```

### 1.4.3. Перемещение и удаление файлов или каталогов

Команда `rm` удаляет файлы и (или) каталоги и имеет следующий синтаксис:

```
rm [опции] <файл|каталог> [файл|каталог...]
```

Опции команды `rm`:

- `-r` или `-R`: рекурсивное удаление (это обязательная опция для удаления любого каталога, пустого или содержащего файлы и (или) подкаталоги);
- `-i`: запрос подтверждения перед удалением;
- `-v`: вывод подробной информации при выполнении команды;
- `-f`: принудительное удаление файлов или каталогов.

Для удаления пустых каталогов можно воспользоваться командой `rmdir`.

Запросив подтверждения на удаление каждого файла в текущем каталоге, удалите в подкаталоге `/newdir/dir1/dir2/` все файлы с именами, заканчивающимися на `.txt`:

```
user@dk4n31:~$ rm -i ~/newdir/dir1/dir2/*.txt
```

Рекурсивно удалите из текущего каталога без запроса подтверждения на удаление каталог `newdir`, а также файлы, чьи имена начинаются с `dir` в каталоге `parentdir`:

```
user@dk4n31:~$ rm -R ~/newdir ~/parentdir/dir*
```

Команда `rm` удаляет файлы безвозвратно, и не существует способа для их восстановления.

Команда `mv` служит для перемещения файлов и каталогов и имеет следующий синтаксис:

```
mv [опции] <файл|каталог> [файл|каталог...] <назначение>
```

Некоторые опции:

- `-f`: принудительное выполнение операции (предупреждение не будет выводиться даже при перезаписи существующего файла);
- `-i`: запрашивается подтверждение перед перезаписью существующего файла;
- `-v`: подробный режим, который сообщает обо всех изменениях и действиях при выполнении команды.

Команда `cp` копирует файлы и каталоги и имеет следующий синтаксис:

```
cp [опции] <файл|каталог> [файл|каталог...] <назначение>
```

Некоторые опции команды `cp`:

- `-R`: рекурсивное копирование; является обязательной опцией для копирования каталогов;
- `-i`: запрос подтверждения перед перезаписью любых файлов;
- `-f`: заменяет любые существующие файлы без запроса подтверждения;

- -v: подробный режим, сообщает обо всех изменениях и действиях.

Для демонстрации работы команд `cp` и `mv` приведем следующие примеры.

Создайте следующие файлы и каталоги в домашнем каталоге:

```
user@dk4n31:~$ cd
user@dk4n31:~$ mkdir -p parentdir1/dir1 parentdir2/dir2 parentdir3
user@dk4n31:~$ touch parentdir1/dir1/test1.txt parentdir2/dir2/test2.txt
```

Используя команды `cp` и `mv` файл `test1.txt` скопируйте, а `test2.txt` переместите в каталог `parentdir3`:

```
user@dk4n31:~$ mv parentdir1/dir1/test1.txt parentdir3
user@dk4n31:~$ cp parentdir2/dir2/test2.txt parentdir3
```

С помощью команды `ls` проверьте корректность выполненных команд

```
user@dk4n31:~$ ls parentdir3
test1.txt test2.txt
user@dk4n31:~$ ls parentdir1/dir1
user@dk4n31:~$ ls parentdir2/dir2
test2.txt
```

Также команда `mv` может быть использована для переименования файлов и каталогов, а команда `cp` позволяет сделать копию файла с новым именем

Переименуйте файл `test1.txt` из каталога `parentdir3` в `newtest.txt`, запрашивая подтверждение перед перезаписью:

```
user@dk4n31:~$ ls parentdir3
test1.txt test2.txt
user@dk4n31:~$ cp parentdir3/test2.txt parentdir3/subtest2.txt
user@dk4n31:~$ mv -i parentdir3/test1.txt parentdir3/newtest.txt
user@dk4n31:~$ ls parentdir3
newtest.txt subtest2.txt test2.txt
```

Переименуйте каталог `dir1` в каталоге `parentdir1` в `newdir`:

```
user@dk4n31:~$ cd parentdir1
user@dk4n31:~/parentdir1$ ls
dir1
user@dk4n31:~/parentdir1$ mv dir1 newdir
user@dk4n31:~/parentdir1$ ls
newdir
```

### 1.4.4. Команда cat: вывод содержимого файлов

Команда cat объединяет файлы и выводит их на стандартный вывод (обычно это экран):

```
user@dk4n31:~$ cat /etc/hosts
#
# /etc/hosts: static lookup table for host names
#
#<ip-address>    <hostname.domain.org>    <hostname>
127.0.0.1        localhost.localdomain    localhost
# End of file
```

### 1.5. Задание для самостоятельной работы

1. Воспользовавшись командой `pwd`, узнайте полный путь к своей домашней директории.
2. Введите следующую последовательность команд

```
cd
mkdir tmp
cd tmp
pwd
cd /tmp
pwd
```

Объясните, почему вывод команды `pwd` при переходе в каталог `tmp` дает разный результат.

3. Пользуясь командами `cd` и `ls`, посмотрите содержимое **корневого каталога, домашнего каталога**, каталогов `/etc` и `/usr/local`.
4. Пользуясь изученными консольными командами, в своём домашнем каталоге создайте каталог `temp` и каталог `labs` с подкаталогами `lab1`, `lab2` и `lab3` одной командой. В каталоге `temp` создайте файлы `text1.txt`, `text2.txt`, `text3.txt`. Пользуясь командой `ls`, убедитесь, что все действия выполнены успешно (каталоги и файлы созданы).
5. С помощью любого текстового редактора (например, редактора `mc`) запишите в файл `text1.txt` свое имя, в файл `text2.txt` фамилию, в файл `text3.txt` учебную группу. Выведите на экран содержимое файлов, используя команду `cat`.

Для открытия текстового редактора в командной строке необходимо указать его название и имя редактируемого файла. Например `bash user@dk4n31:~/temp$ mc text1.txt`

1. Скопируйте все файлы, чьи имена заканчиваются на `.txt`, из каталога `~/temp` в каталог `labs`. После этого переименуйте файлы каталога `labs` и переместите их: `text1.txt` переименуйте в `firstname.txt` и переместите в подкаталог `lab1`,

text2.txt в lastname.txt в подкаталог lab2, text3.txt в id-group.txt в подкаталог lab3. Пользуясь командами ls и cat, убедитесь, что все действия выполнены верно.

2. Удалите все созданные в ходе выполнения лабораторной работы файлы и каталоги.

## 1.6. Содержание отчёта

Отчёт должен включать:

- Титульный лист с указанием номера лабораторной работы и ФИО студента.
- Формулировка цели работы.
- Описание результатов выполнения лабораторной работы:
  - описание выполняемого задания;
  - скриншоты (снимки экрана), фиксирующие выполнение заданий лабораторной работы;
  - комментарии и выводы по результатам выполнения заданий.
- Описание результатов выполнения заданий для самостоятельной работы:
  - описание выполняемого задания;
  - скриншоты (снимки экрана), фиксирующие выполнение заданий;
  - комментарии и выводы по результатам выполнения заданий.
- Выводы, согласованные с целью работы.

Отчёт по выполнению лабораторной работы оформляется в любом текстовом процессоре (OpenOffice, Libreoffice и др.) с последующей конвертацией в формат pdf.

## 1.7. Вопросы для самопроверки

1. Дайте определение командной строки. Приведите примеры.
2. Как получить информацию об интересующей вас команде?
3. Чем относительный путь к файлу отличается от абсолютного?
4. Как определить абсолютный путь к текущей директории?
5. При помощи каких команд можно удалить файл и каталог? Можно ли это сделать одной и той же командой?
6. Как можно запустить нескольких команд в одной строке? Приведите примеры.
7. Какая информация выводится на экран о файлах и каталогах, если используется опция -l в команде ls?
8. Каким образом отобразить информацию о скрытых файлах? Приведите примеры.
9. Какая клавиша или комбинация клавиш служит для автоматического дополнения вводимых команд?

## Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. *Newham C.* Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658>.
6. *Robbins A.* Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. *Zarrelli G.* Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. *Колдаев В. Д., Лупин С. А.* Архитектура ЭВМ. — М. : Форум, 2018.
10. *Куляс О. Л., Никитин К. А.* Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. *Новожилов О. П.* Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
13. *Робачевский А., Немнюгин С., Стесик О.* Операционная система UNIX. — 2-е изд. — БХВ-Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
14. *Столяров А.* Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: [http://www.stolyarov.info/books/asm\\_unix](http://www.stolyarov.info/books/asm_unix).
15. *Таненбаум Э.* Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
16. *Таненбаум Э., Бос Х.* Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).