

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики  
Кафедра информационных технологий

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №2  
по дисциплине  
«Операционные системы»  
на тему:  
«Работа с терминалом и оболочкой Bash»

Выполнила студентка группы МО32 \_\_\_\_\_ В.С. Переузник

Направление подготовки 02.03.03 Математическое обеспечение и  
администрирование информационных систем  
Курс 3

Отчет принял преподаватель кафедры ИТ \_\_\_\_\_ А.А. Полупанов

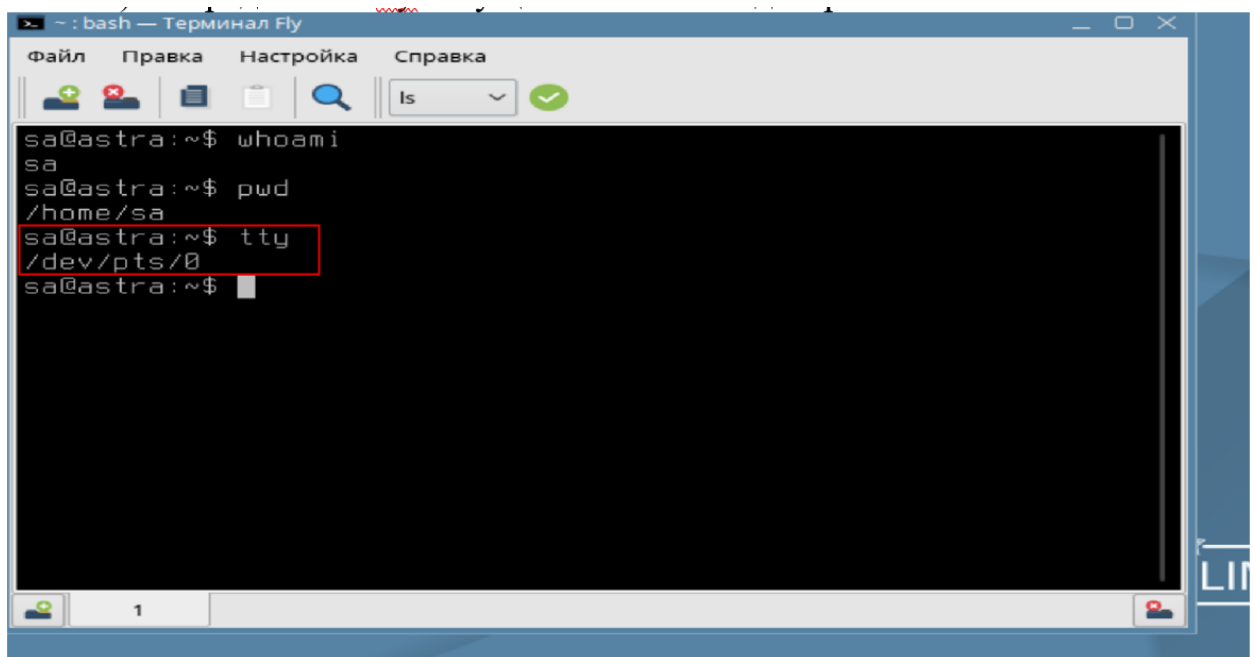
Краснодар  
2025

**Задача:** 1) Изучить основы работы с псевдотерминалом и командами; 2) Освоить работу с файловой системой и правами доступа; 3) Изучить переменные окружения и права доступа.

## Задание 1

### 1) Определение tty текущей сессии псевдотерминала

Командой `tty` была определена текущая сессия псевдотерминала. Как показано на рисунке 1.1, результатом выполнения команды стал вывод `/dev/pts/0`, что указывает на использование псевдотерминала типа PTS (Pseudo Terminal Slave).



```
sa@astra:~$ whoami
sa
sa@astra:~$ pwd
/home/sa
sa@astra:~$ tty
/dev/pts/0
sa@astra:~$
```

Рисунок 1.1 - Определение текущего терминала

### 2) Отправка эхо на файл псевдотерминала

Для демонстрации работы с псевдотерминалом было отправлено сообщение "Hello from Vika" в файл терминала, как представлено на рисунке 1.2. Это подтверждает, что псевдотерминалы в Linux представляются как файлы, и запись в них отображается в соответствующем терминале.

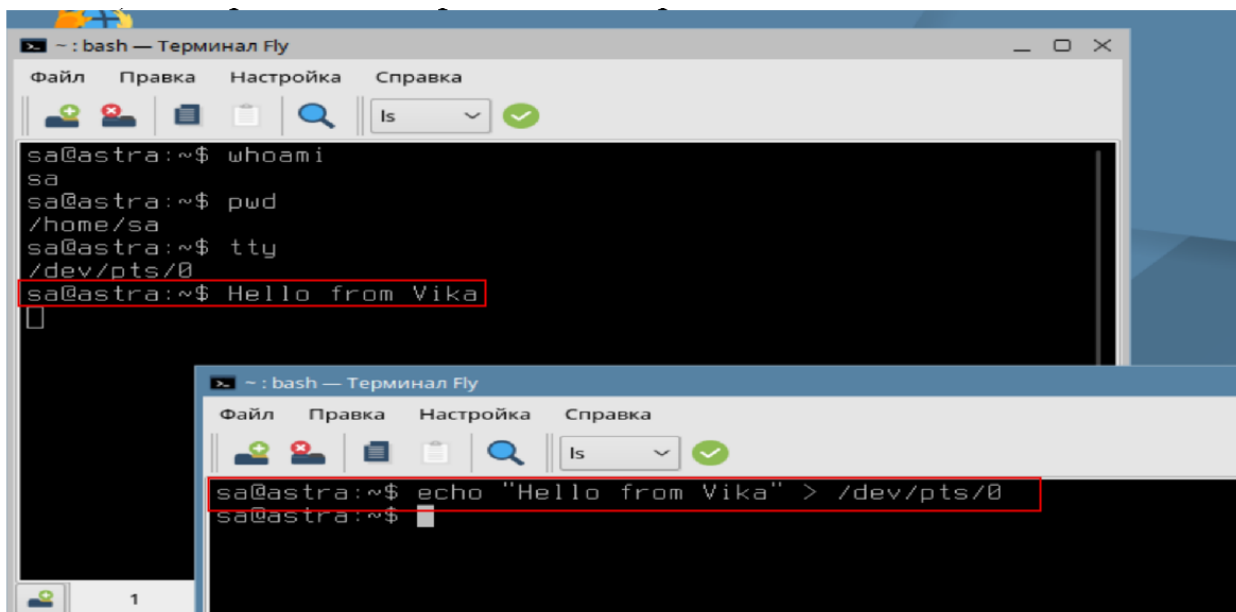


Рисунок 1.2 - Отправка сообщения в терминал

3) Вывод переменных `$COLUMNS` и `$LINES` (текущие ширина и высота терминала в пикселях)

С помощью команды `echo` были выведены переменные окружения, определяющие размер терминала. На рисунке 1.3 видно, что терминал имеет размер 62 колонки и 18 строк.

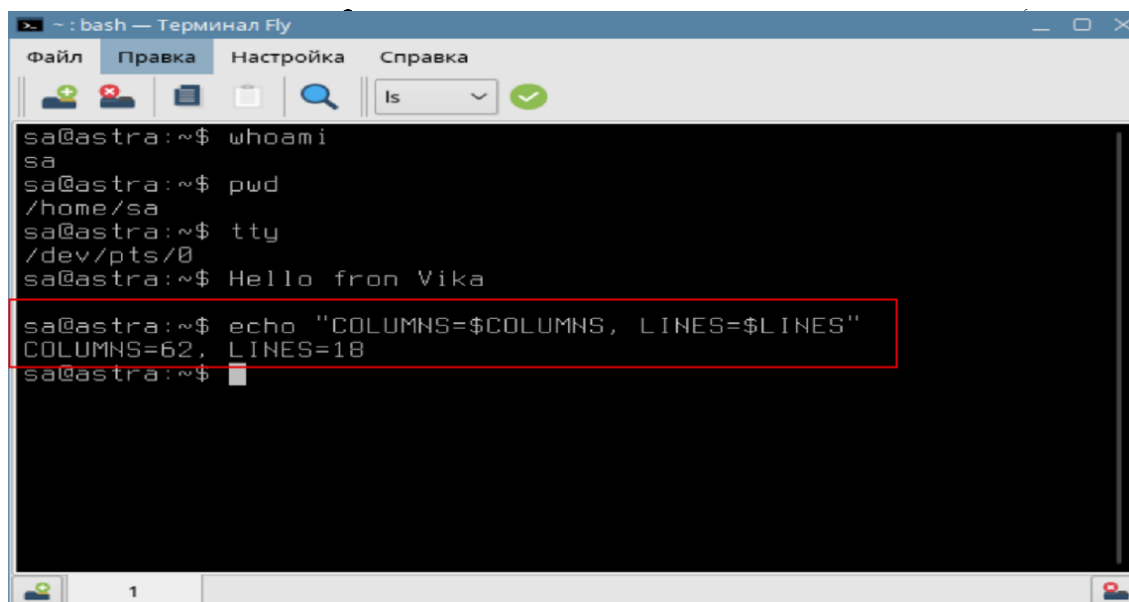
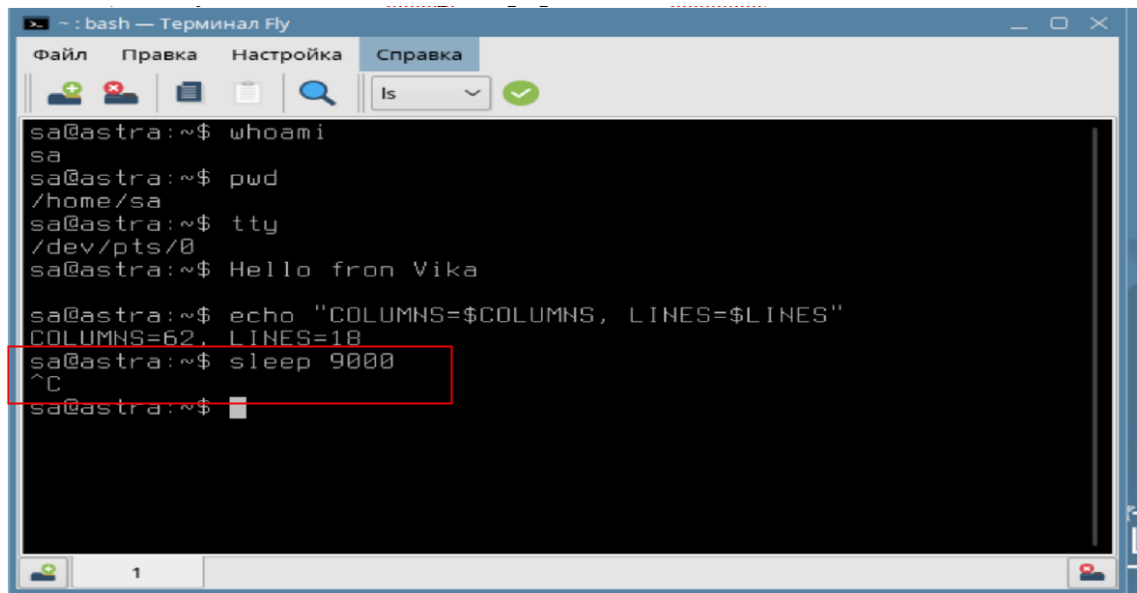


Рисунок 1.3 - Размеры терминала

#### 4) Запуск команды sleep и прерывание Ctrl+C

Была запущена команда sleep 9888 для создания длительного процесса, который затем был успешно прерван сочетанием клавиш Ctrl+C, что демонстрируется на рисунке 1.4. Символ ^C в выводе подтверждает отправку сигнала прерывания.



```
sa@astra:~$ whoami
sa
sa@astra:~$ pwd
/home/sa
sa@astra:~$ tty
/dev/pts/0
sa@astra:~$ Hello from Vika

sa@astra:~$ echo "COLUMNS=$COLUMNS, LINES=$LINES"
COLUMNS=62, LINES=18
sa@astra:~$ sleep 9888
^C
sa@astra:~$
```

Рисунок 1.4 - Прерывание команды

#### 5) Запуск mc и попытка закрытия Ctrl+C

После запуска файлового менеджера Midnight Commander (mc) была предпринята попытка закрыть его с помощью Ctrl+C, которая не увенчалась успехом, как показано на рисунке 1.5.

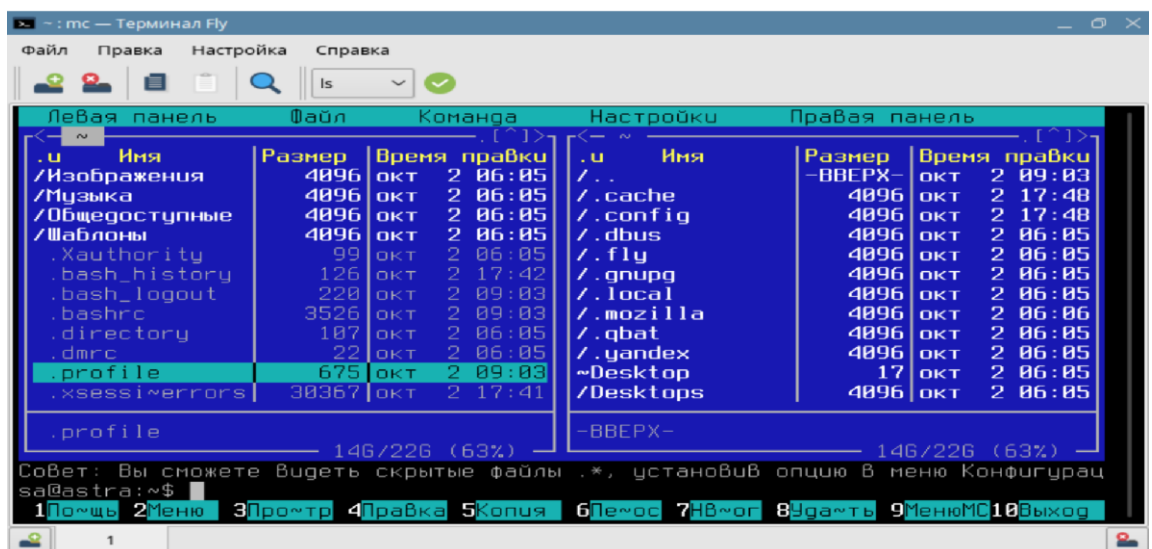


Рисунок 1.5 - Работа с Midnight Commander

## 6) Почему tc не закрывается через Ctrl + C ?

tc не закрывается по Ctrl+C, потому что эта программа использует это сочетание клавиш для других целей - в основном для копирования файлов. Когда работаешь в tc, он полностью перехватывает управление клавиатурой и создает свой собственный интерфейс. Ctrl+C в нем просто не предназначен для выхода - он занят другими задачами. Чтобы выйти из tc нормально, нужно нажать F10 или через меню найти пункт "Exit". Это стандартный способ закрытия для этой программы.

## Задание 2

### 1) Узнаем, в какой рабочей директории находимся.

Командой `pwd` была определена текущая рабочая директория. На рисунке 2.1 видно, что начальной директорией является `/home/sa`.

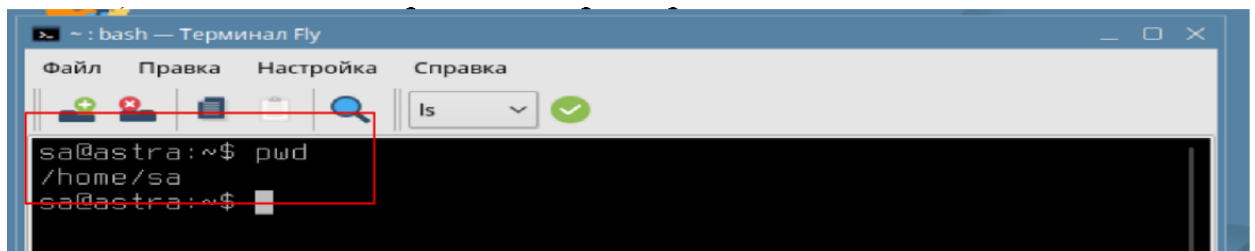


Рисунок 2.1 - Текущая директория

### 2) Переходим в директорию etc

С помощью команды `cd /etc` был осуществлен переход в системную директорию `etc`. Как показано на рисунке 2.2, в командной строке изменилось отображение текущего пути с `~` на `/etc`.

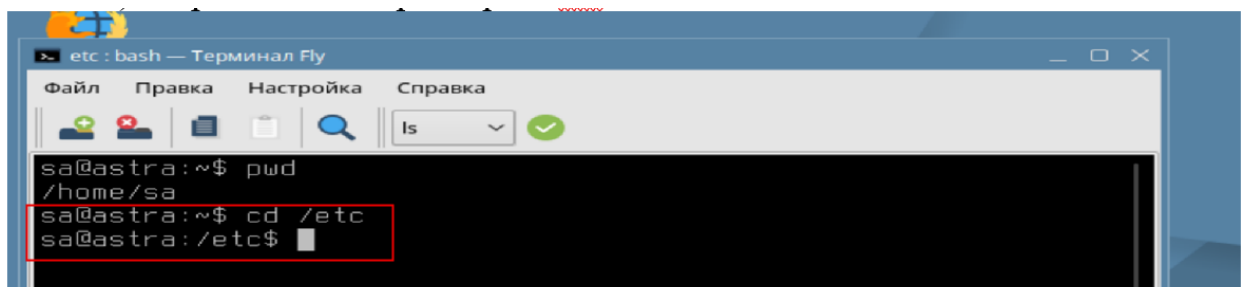
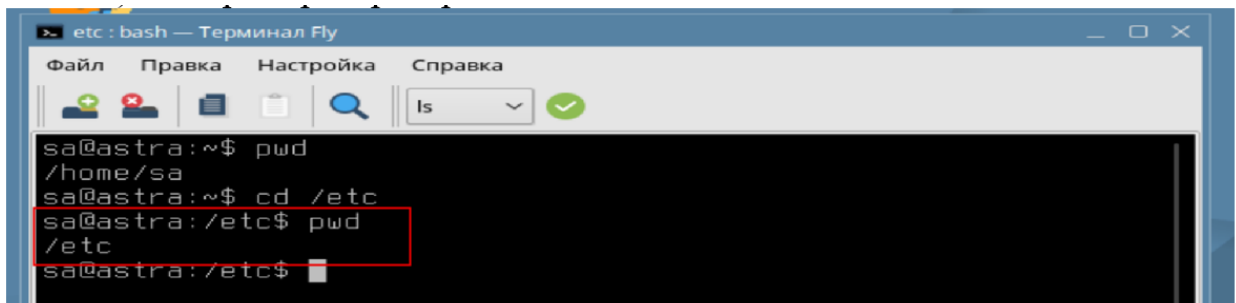


Рисунок 2.2 - Смена директории

Что изменилось: в командной строке теперь показывается /etc вместо ~ - так удобнее видеть, где работаешь

3) Еще раз проверим рабочий каталог.

Повторное использование команды pwd подтвердило успешную смену рабочей директории, что отображено на рисунке 2.3.

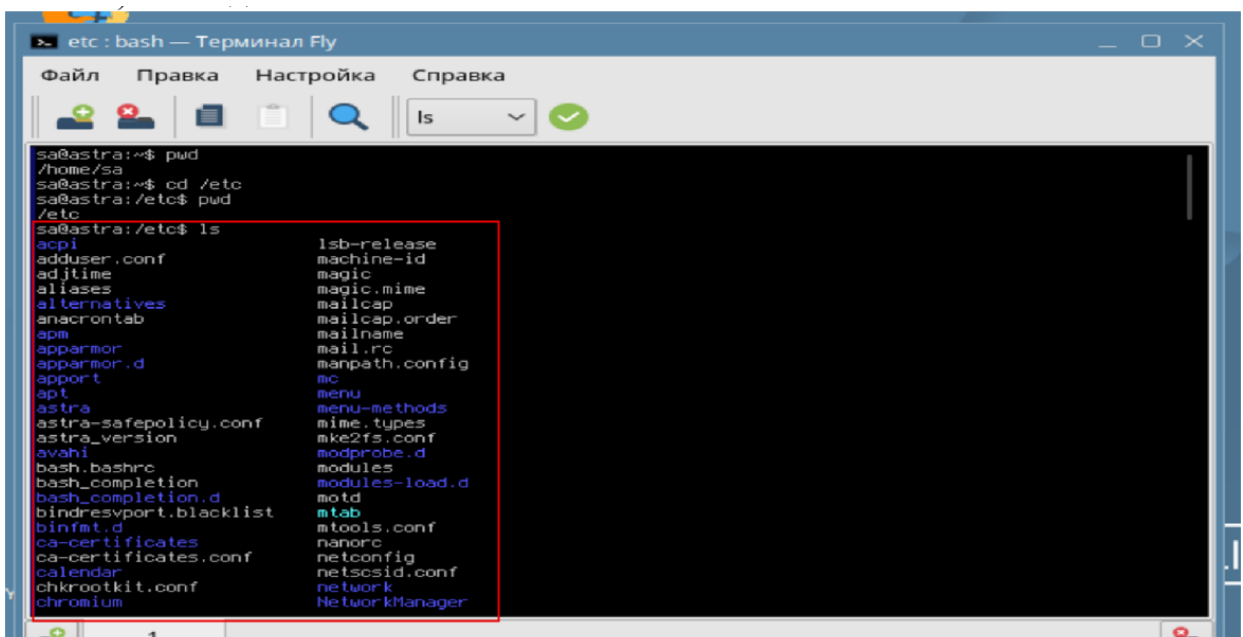


```
sa@astra:~$ pwd
/home/sa
sa@astra:~$ cd /etc
sa@astra:/etc$ pwd
/etc
sa@astra:/etc$
```

Рисунок 2.3 - Подтверждение смены директории

4) Выведем список всех объектов.

Командой ls был получен список файлов и директорий в текущем каталоге /etc. На рисунке 2.4 представлен частичный вывод, содержащий многочисленные конфигурационные файлы системы.



```
sa@astra:~$ pwd
/home/sa
sa@astra:~$ cd /etc
sa@astra:/etc$ pwd
/etc
sa@astra:/etc$ ls
acpi                                lsb-release
adduser.conf                       machine-id
adjtime                            magic
aliases                           magic.mime
alternatives                       mailcap
anacrontab                         mailcap.order
apm                                mailname
apparmor                           mail.rc
apparmor.d                         manpath.config
appport                            mc
apt                                 menu
astra                              menu-methods
astra-safepolicy.conf              mime.types
astra_version                      mke2fs.conf
avahi                              modprobe.d
bash.bashrc                        modules
bash_completion                   modules-load.d
bash_completion.d                 motd
bindresvport.blacklist            mtab
binfmt.d                          mtools.conf
ca-certificates                   nanorc
ca-certificates.conf              netconfig
calendar                          netcsid.conf
chkrootkit.conf                   network
chromium                          NetworkManager
```

Рисунок 2.4 - Содержимое директории /etc

Там много...

#### 5) Смотрим hostname.

Командой `cat hostname` было выведено содержимое файла с именем хоста системы. Рисунок 2.5 демонстрирует, что команда `cat` только читает файл, не изменяя его содержимое.

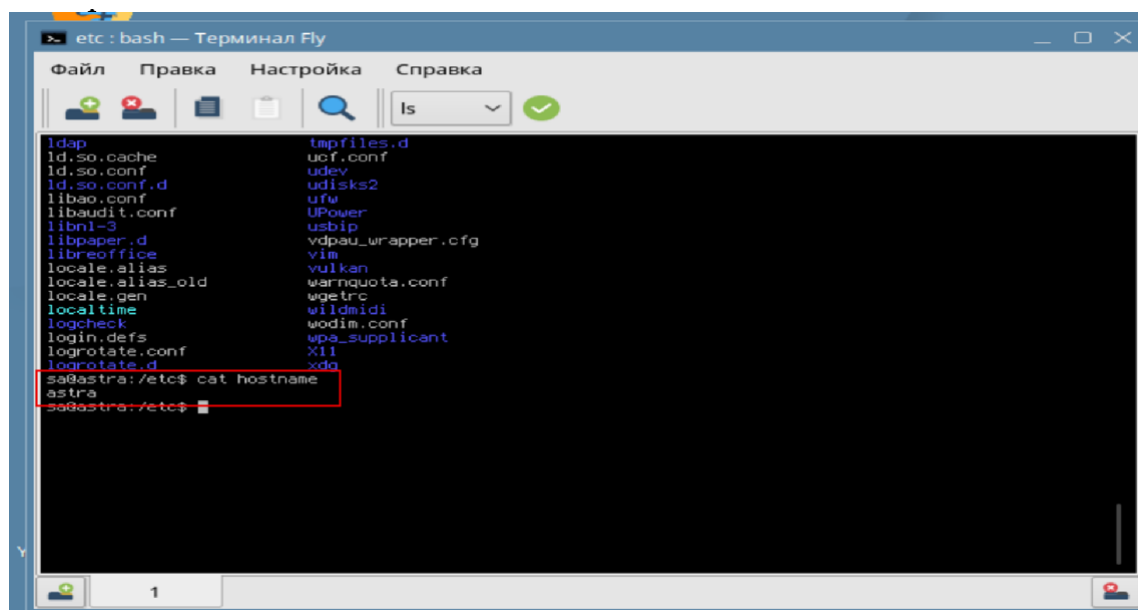


Рисунок 2.5 - Просмотр имени хоста

- Файл изменился? - Нет, `cat` только читает файл, не изменяет его
- Куда вывела содержимое? - На экран (в стандартный вывод)
- Безопасна ли команда? - Да, она только читает, но надо быть осторожными с большими файлами

## 6) Делаем копию hostname

Была создана резервная копия файла hostname с помощью перенаправления вывода. Как видно на рисунке 2.6, файл hostname.old был сохранен в домашней директории пользователя.

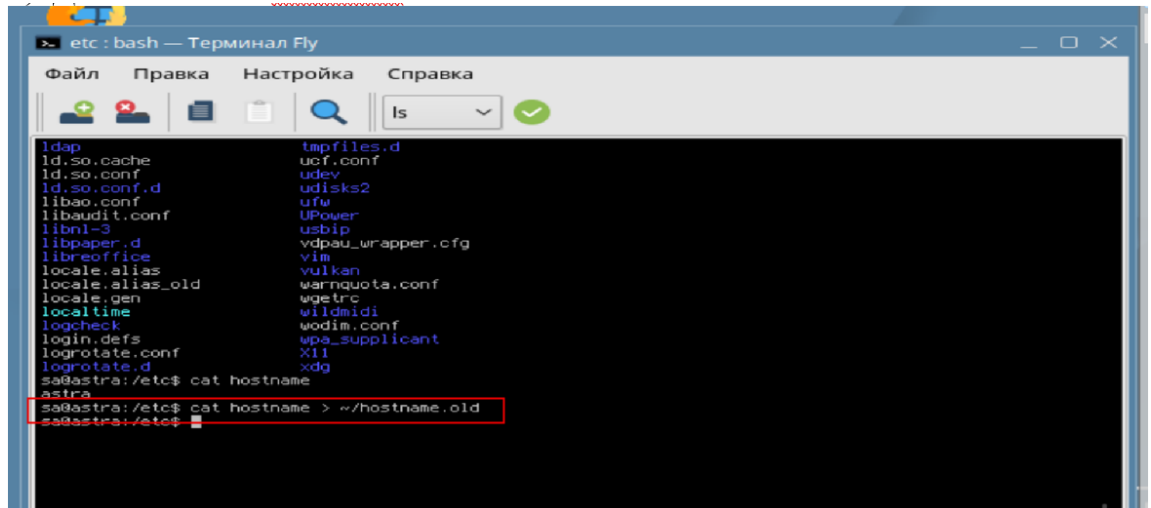


Рисунок 2.6 - Создание резервной копии

Куда сохранился? - В домашнюю директорию (~ означает домашнюю папку)

## 7) Проверяем бэкап

Содержимое резервной копии было проверено командой `cat < ~/hostname.old`, что подтвердило корректность создания копии (рисунок 2.7).

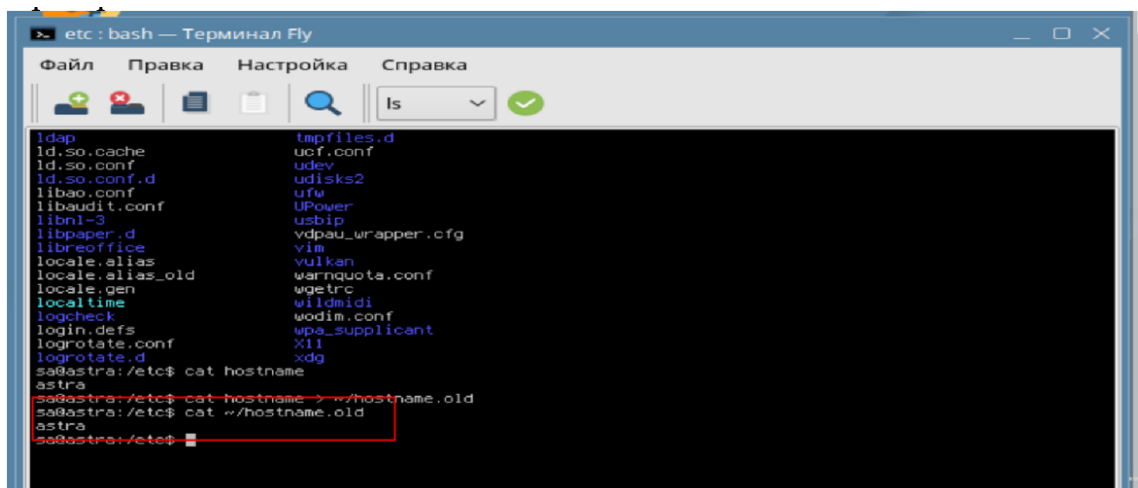


Рисунок 2.7 - Проверка резервной копии



## 8) Меняем имя хоста с помощью редактора nano

Была предпринята попытка редактирования файла `hostname` с помощью редактора nano без прав суперпользователя. Как показано на рисунке 2.8, система не позволила сохранить изменения из-за недостаточных прав доступа.

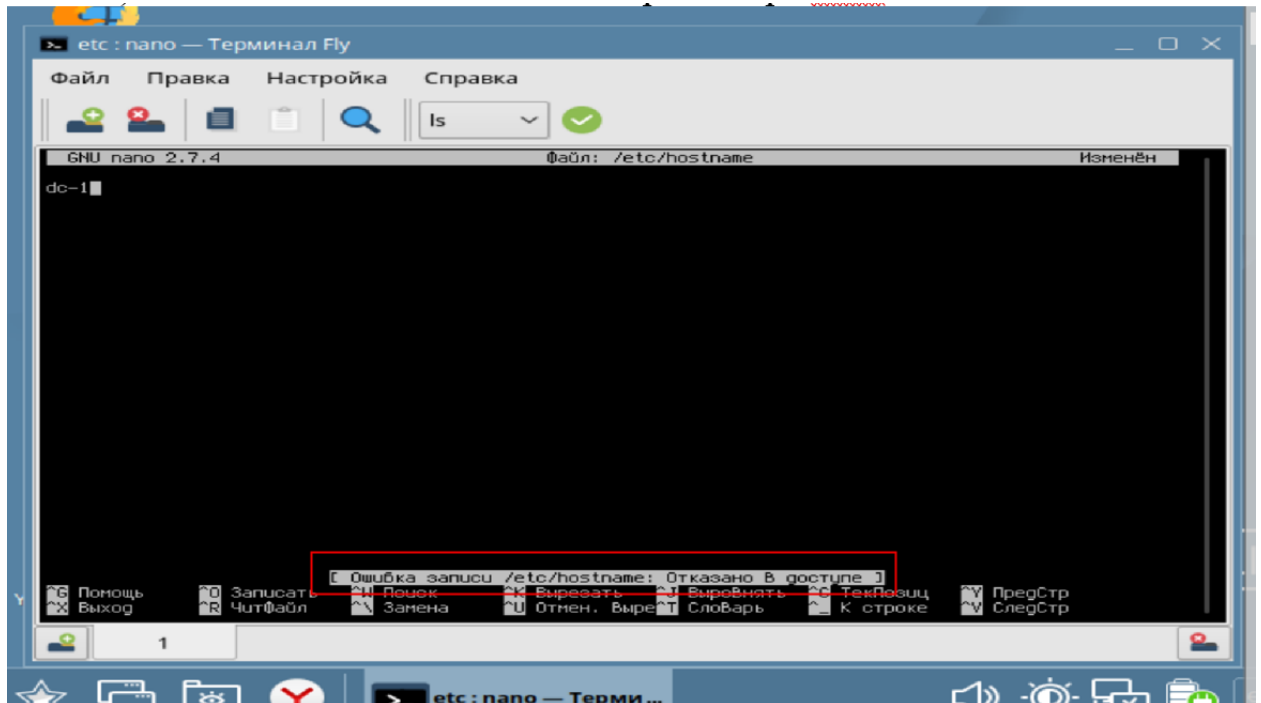


Рисунок 2.8 - Попытка редактирования без прав

Почему подчеркивает красным цветом [ File „/etc/hostname“ is unwritable ]? - У меня не подчеркивалось красным, а просто вывелось сообщение. Но вообще красное подчеркивание или сообщение об ошибке появляется потому, что у обычного пользователя нет прав изменять системные файлы. Файл `/etc/hostname` принадлежит администратору системы, и его можно редактировать только с помощью `sudo`. Без прав суперпользователя система защищает важные файлы от случайных изменений, поэтому nano не позволяет сохранить изменения и показывает ошибку.

## 9) Повторяем с правами суперпользователя

После получения прав суперпользователя через `sudo` была успешно изменена конфигурация файла `hostname`, как демонстрируется на рисунке 2.9.

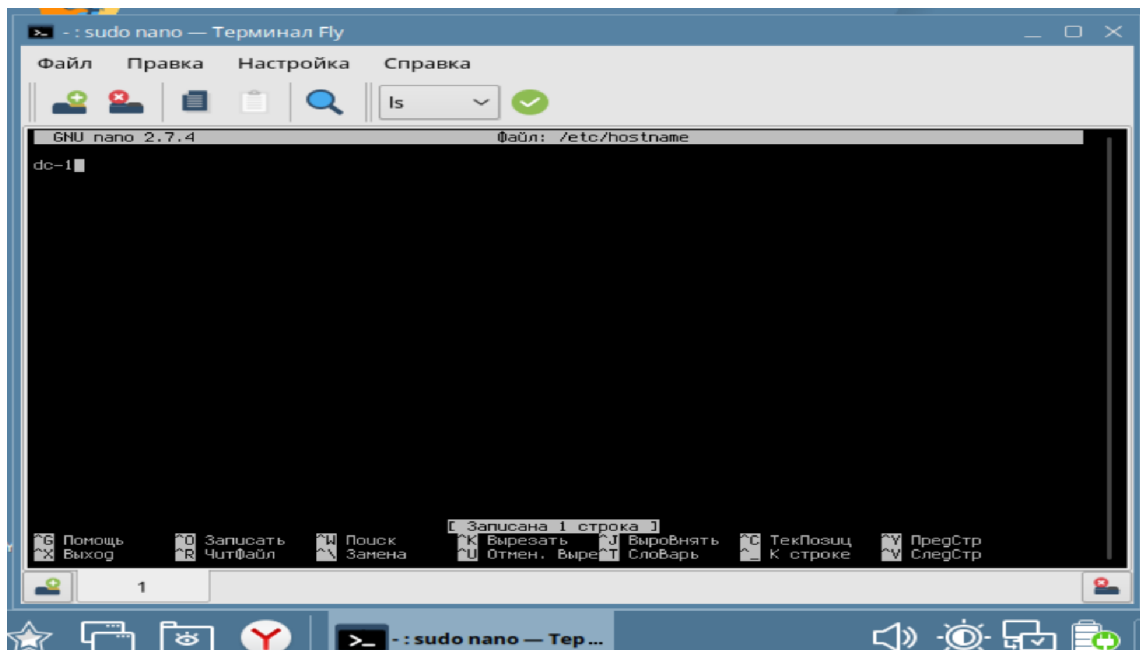


Рисунок 2.9 - Редактирование с правами root

### Задание 3

#### 1) Приветствие текущему пользователю

С помощью команды echo было выведено персонализированное приветствие с использованием переменной окружения \$USER. На рисунке 3.1 показано применение переменных окружения в командной строке.

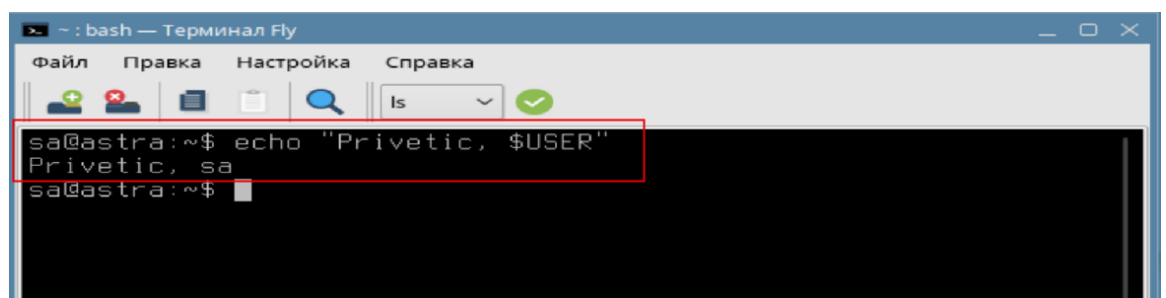


Рисунок 3.1 - Приветствие пользователю

#### 2) Посмотрим, какие есть глобальные переменные окружения.

Командой `env` были получены глобальные переменные окружения системы. Рисунок 3.2 демонстрирует частичный вывод, содержащий многочисленные переменные конфигурации.

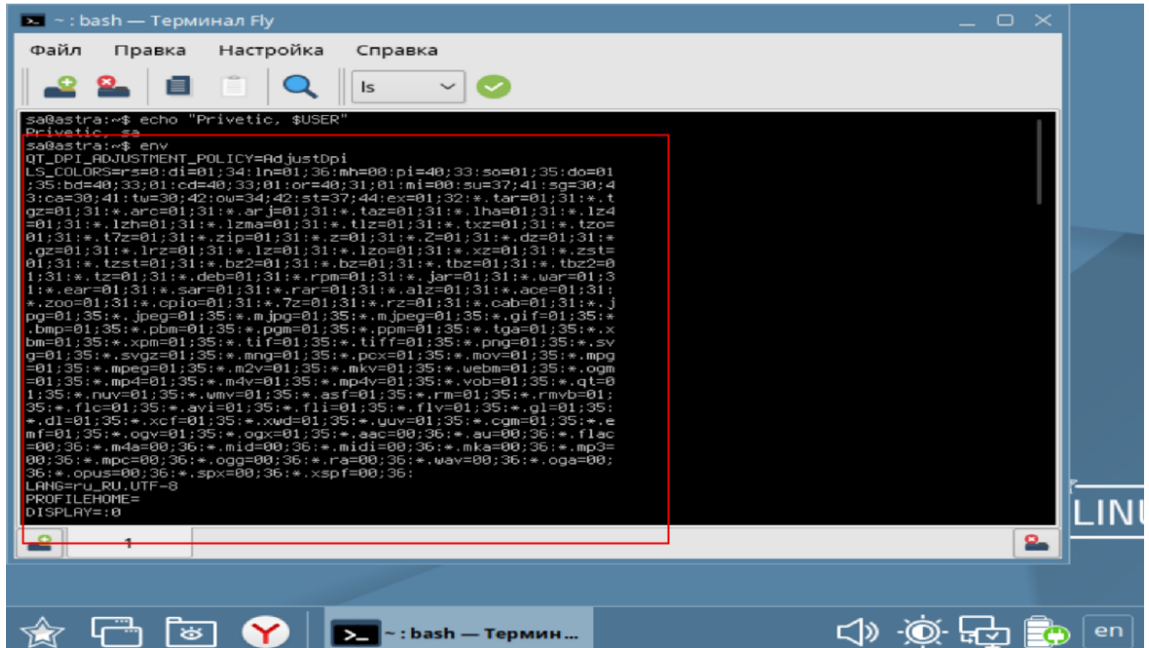


Рисунок 3.2 - Глобальные переменные окружения

3) Выведем все переменные текущей сессии (команда `set`).

Команда `set` вывела все переменные текущей сессии, включая локальные и глобальные. Как показано на рисунке 3.3, вывод содержит значительно больше информации по сравнению с командой `env`.

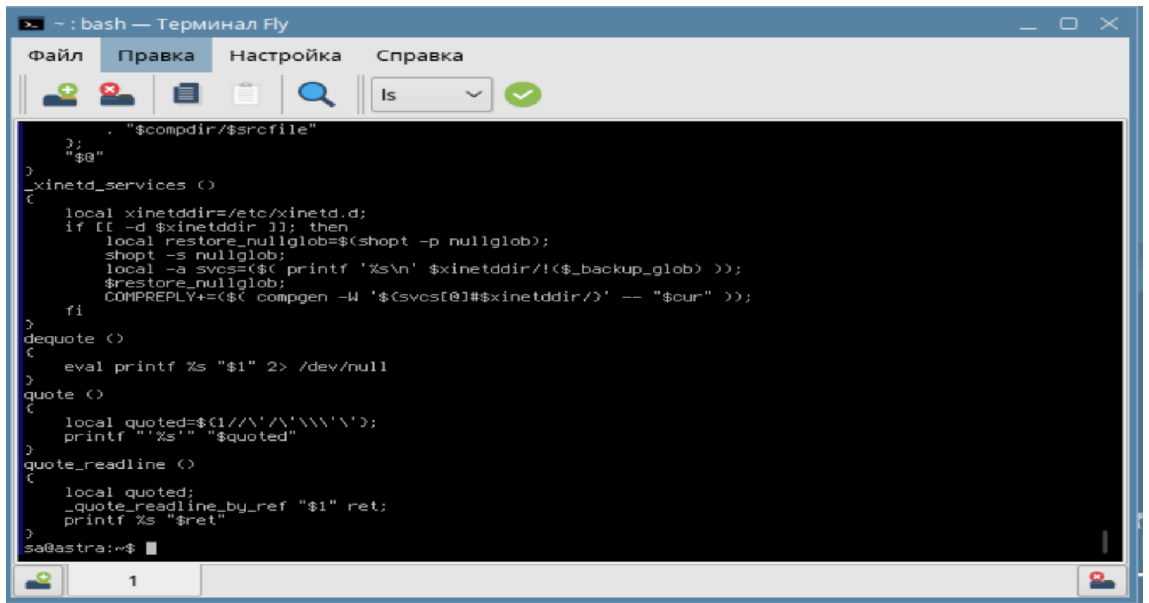


Рисунок 3.3 - Все переменные сессии

В таком формате выводит, но стирает с терминала, все что было до нее. Так как команда `set` выводит очень большое количество переменных, из-за чего терминал быстро прокручивает вывод и создается впечатление, что предыдущий текст стирается. Однако вся история команд сохраняется в буфере терминала (`history`).

#### 4) Найдем размер терминала.

С помощью конвейера и команды `grep` были отфильтрованы переменные `LINES` и `COLUMNS`. На рисунке 3.4 видно, что терминал имеет размер 104 колонки и 31 строка.

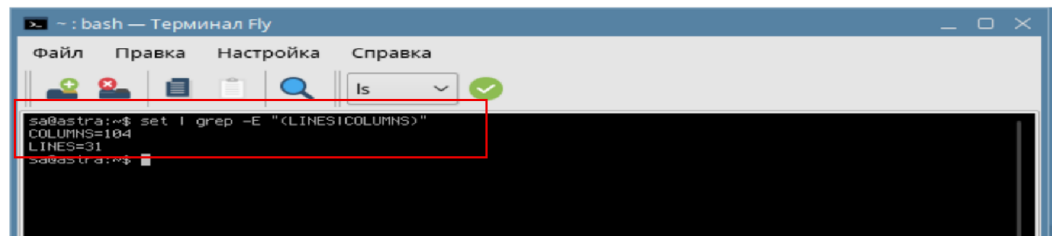


Рисунок 3.4 - Переменные размера терминала

#### 5) Попробуем перезагрузить без root.

Была предпринята попытка выполнить перезагрузку системы командой `reboot` без прав суперпользователя. Как демонстрируется на рисунке 3.5, команда не была найдена в PATH обычного пользователя.

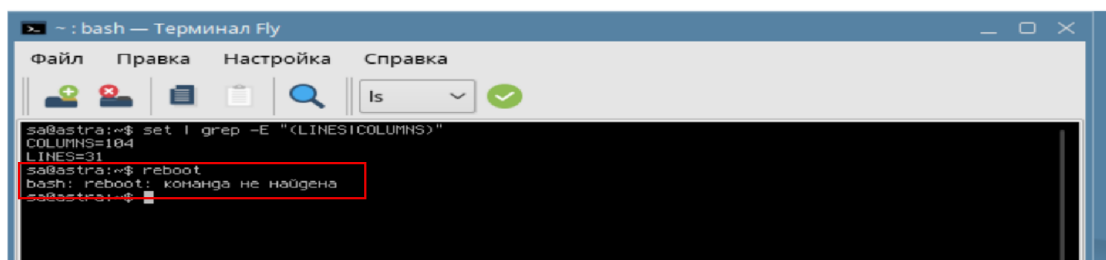
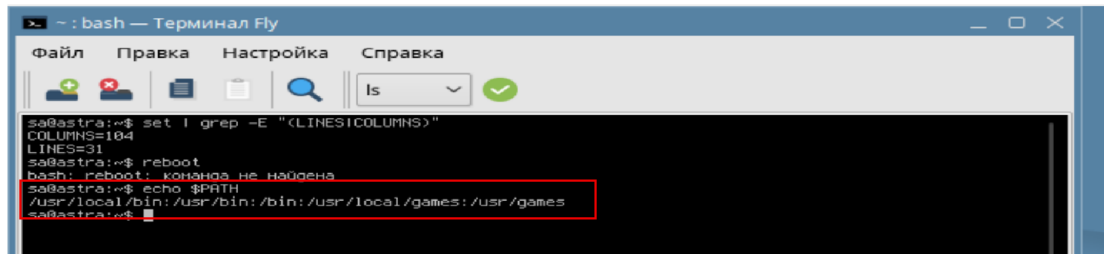


Рисунок 3.5 - Попытка перезагрузки

Почему не получилось? - обычный пользователь не может выполнять системные команды, влияющие на всю систему.

5.1) Посмотрим, какие бинарные файлы может запускать обычный пользователь

Командой `echo $PATH` был проанализирован список директорий, в которых обычный пользователь может выполнять команды. Рисунок 3.6 показывает отсутствие системных директорий `/sbin` и `/usr/sbin` в `PATH`.

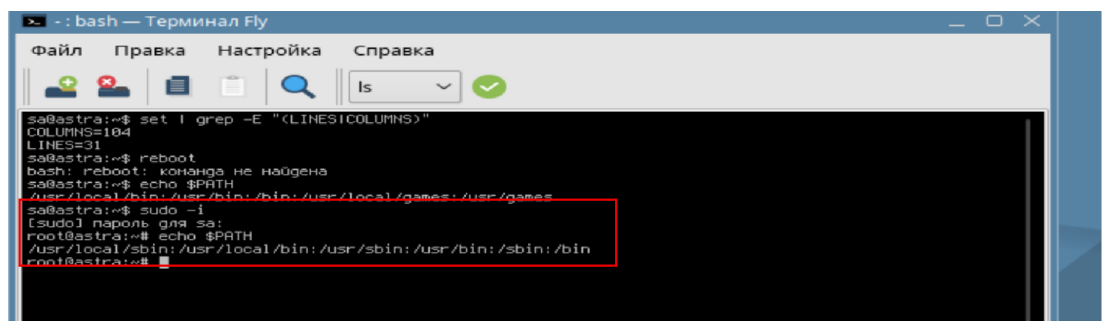


```
sa@astra:~$ set | grep -E "(LINES|COLUMNS)"
COLUMNS=104
LINES=31
sa@astra:~$ reboot
bash: reboot: команда не найдена
sa@astra:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
sa@astra:~$
```

Рисунок 3.6 - `PATH` обычного пользователя

5.2) Посмотрим, какие файлы может запускать `root` пользователь

После перехода в сессию `root` была проверена переменная `PATH`. На рисунке 3.7 видно, что у пользователя `root` в `PATH` включены системные директории `/sbin` и `/usr/sbin`.



```
sa@astra:~$ set | grep -E "(LINES|COLUMNS)"
COLUMNS=104
LINES=31
sa@astra:~$ reboot
bash: reboot: команда не найдена
sa@astra:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
sa@astra:~$ sudo -i
[sudo] пароль для sa:
root@astra:~# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
root@astra:~#
```

Рисунок 3.7 - `PATH` пользователя `root`

Почему пользователь не может найти команду `reboot`, хотя она есть у `root`? - Команда `reboot` находится в `/sbin`, которая не входит в `$PATH` обычного пользователя. У `root` эти пути добавлены в переменную `PATH`, поэтому он видит эти команды. Системные команды в специальных папках доступны только администратору для безопасности.

**Вывод:** В ходе лабораторной работы были освоены основы работы с терминалом Linux, включая управление процессами, работу с файловой системой, изучение механизмов прав доступа и переменных окружения.