

Министерство образования и молодежной политики
Свердловской области государственное автономное
профессиональное образовательное учреждение
Свердловской области
«Уральский радиотехнический колледж им. А.С. Попова»

ГАПОУ СО УРТК им. А. С. Попова
Специальность Компьютерные сети

Лабораторная работа №1

Программное обеспечение компьютерных сетей.

Екатеринбург 2022 г.

Лабораторная работа №1 “Программное обеспечение компьютерных сетей”

I

1) Запуск оболочки и выход из оболочки

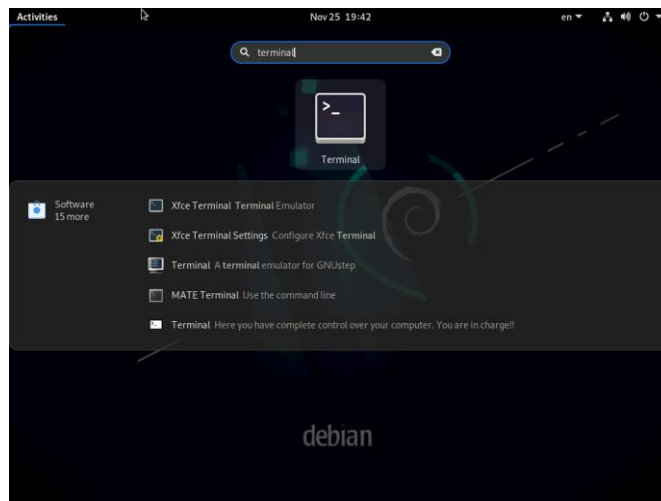


Рисунок 1. Запуск терминала

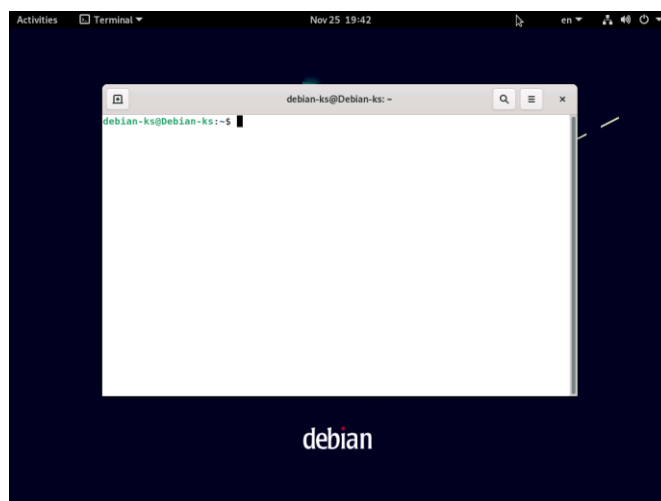


Рисунок 2. Терминал открылся

2) Переключение на виртуальный терминал и обратно

```
debian GNU/Linux 11 debian-ks tty5
debian-ks login: debian-ks
Password:
Linux Debian-ks 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
last login: Fri Nov 25 19:49:10 PST 2022 from 192.168.80.1 on pts/0
debian-ks@debian-ks:~$
```

Рисунок 3. CTRL+ALT+F7 и CTRL+ALT+F5 чтобы попасть в виртуальный терминал tty5



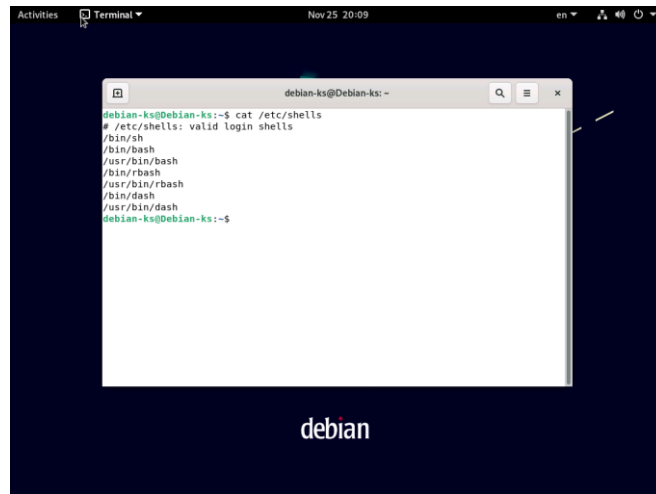
Рисунок 4. CTRL+ALT+F2 вернуться в графическую оболочку

3) Использование опций POSIX/BSD/GNU на примере одной команды “pwd”



Рисунок 5. PWD - позволяет вывести в терминал путь к текущей папке.

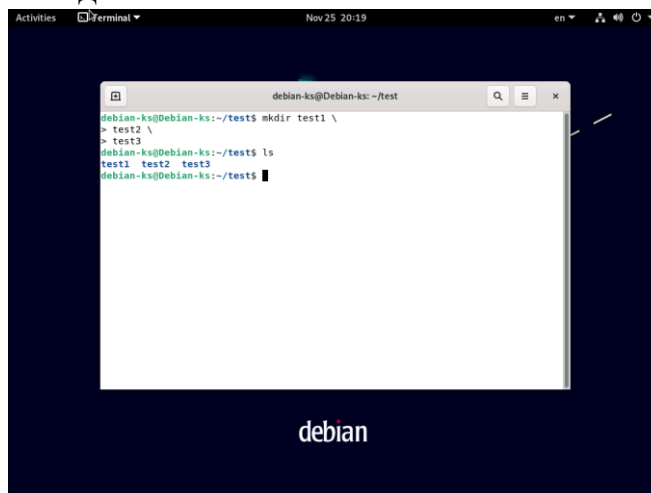
4) Запуск других оболочек и список доступных оболочек

A terminal window titled "Terminal" with a dark blue background. The window shows the command prompt "debian-ks@Debian-ks:~\$" followed by the command "cat /etc/shells". The output of the command is displayed as a list of valid login shells: "/bin/sh", "/bin/bash", "/usr/bin/bash", "/bin/rbash", "/usr/bin/rbash", "/bin/dash", and "/usr/bin/dash". The prompt returns to "debian-ks@Debian-ks:~\$". The Debian logo is visible at the bottom of the terminal window.

```
debian-ks@Debian-ks:~$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
/bin/rbash
/usr/bin/rbash
/bin/dash
/usr/bin/dash
debian-ks@Debian-ks:~$
```

Рисунок 6. Выводим список оболочек, которые в данный момент установлены и доступны в системе

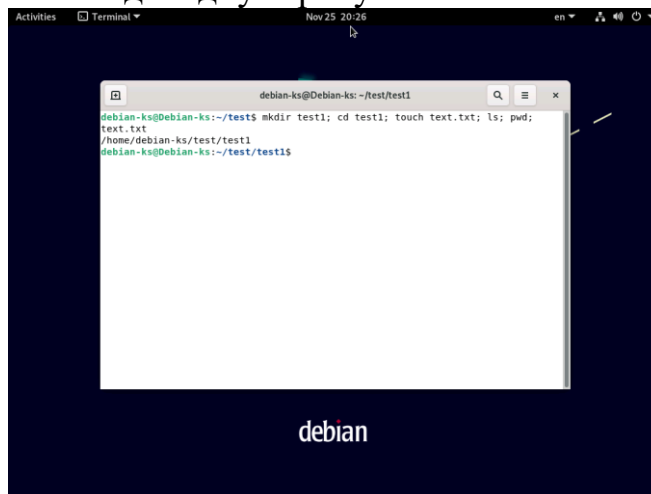
5) Ввод длинной команды



```
debian-ks@Debian-ks:~/test$ mkdir test1 \  
> test2 \  
> test3 \  
debian-ks@Debian-ks:~/test$ ls  
test1 test2 test3  
debian-ks@Debian-ks:~/test$
```

Рисунок 7. Обратный слэш в конце строки указывает терминалу рассматривать следующую строку как продолжение текущей

6) Ввод нескольких команд в одну строку



```
debian-ks@Debian-ks:~/test/test1$ mkdir test1; cd test1; touch text.txt; ls; pwd;  
text.txt  
/home/debian-ks/test/test1  
debian-ks@Debian-ks:~/test/test1$
```

Рисунок 8. Можно выполнить команды последовательно, используя точку с запятой после каждой команды

7) Применение конвейеров || и &&




```
debian-ks@Debian-ks:~/test/test1$ mkdir test1 && cd test1  
debian-ks@Debian-ks:~/test/test1$ cd ..  
debian-ks@Debian-ks:~/test$ mkdir test1 || cd test1  
mkdir: cannot create directory 'test1': File exists  
debian-ks@Debian-ks:~/test/test1$ mkdir test2 || cd test1  
debian-ks@Debian-ks:~/test/test1$
```

Рисунок 9. Зависимое выполнение команд. Символы && и || означают не что иное, как логическое И и ИЛИ. Символы && означают, что каждая

следующая команда выполняется только при успешном выполнении предыдущей. Символы || означают, что каждая следующая команда выполняется только при успешном выполнении предыдущей.

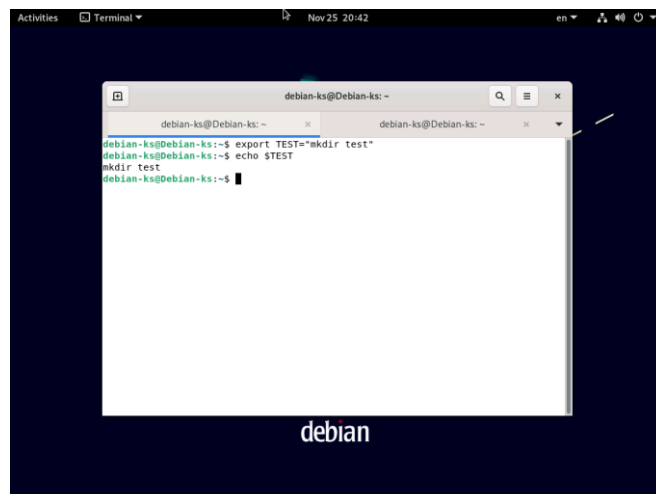
8) Установка переменной оболочки



```
debian-ks@Debian-ks:~$ export TEST=mkdir test
debian-ks@Debian-ks:~$
```

Рисунок 10. Создание переменной оболочки


9) Вывод переменной оболочки



```
debian-ks@Debian-ks:~$ export TEST=mkdir test
debian-ks@Debian-ks:~$ echo $TEST
mkdir test
debian-ks@Debian-ks:~$
```

Рисунок 11. Вывод содержимого переменной оболочки

10) Изменение значения переменной оболочки



```
debian-ks@Debian-ks:~$ export TEST=mkdir test
debian-ks@Debian-ks:~$ echo $TEST
mkdir test
debian-ks@Debian-ks:~$ export TEST=mkdir test15
debian-ks@Debian-ks:~$ echo $TEST
mkdir test15
debian-ks@Debian-ks:~$
```

Рисунок 12. Изменение значения переменной оболочки

11) Тест переменной оболочки и переменной окружения

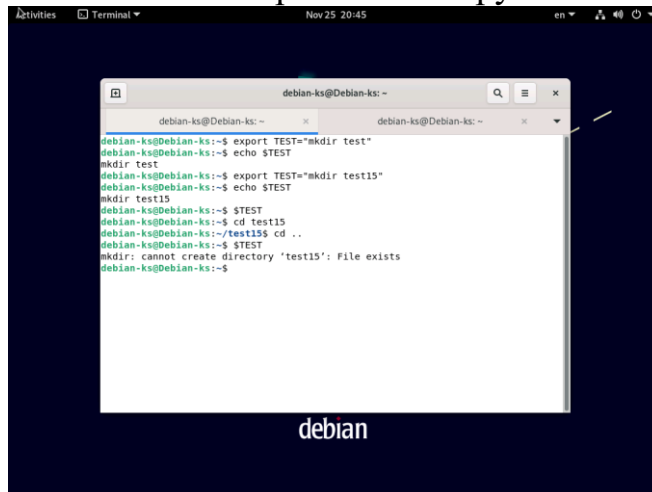


Рисунок 13. С помощью переменной окружения мы создали папку test15

12) Файлы настроек переменных окружения

.BASHRC файл переменных конкретного пользователя. .BASH_PROFILE файл переменных для пользователей SSH. /ETC/ENVIRONMENT файл для создания, редактирования и удаления каких-либо переменных окружения на системном уровне. /ETC/BASH.BASHRC файл для каждого локального пользователя. /ETC/PROFILE доступны любому удаленному пользователю.

13) История команд

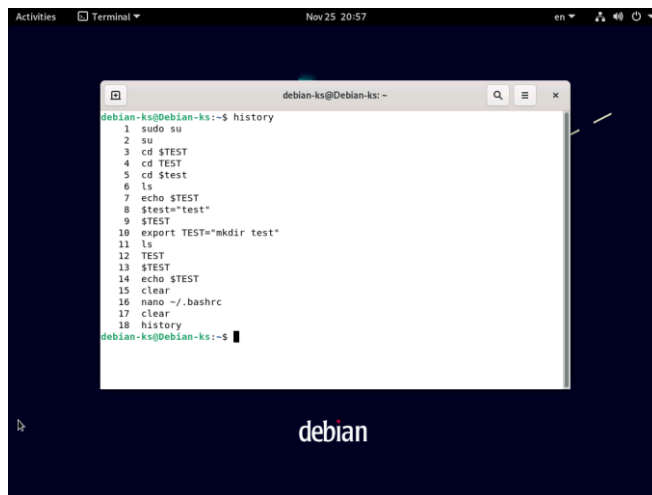


Рисунок 14. Историю команд можно посмотреть, используя команду history. Также команды хранятся в файле .bash_history

14) Автоматическое дополнение задается клавишей ... ТАВ

15) Псевдонимы

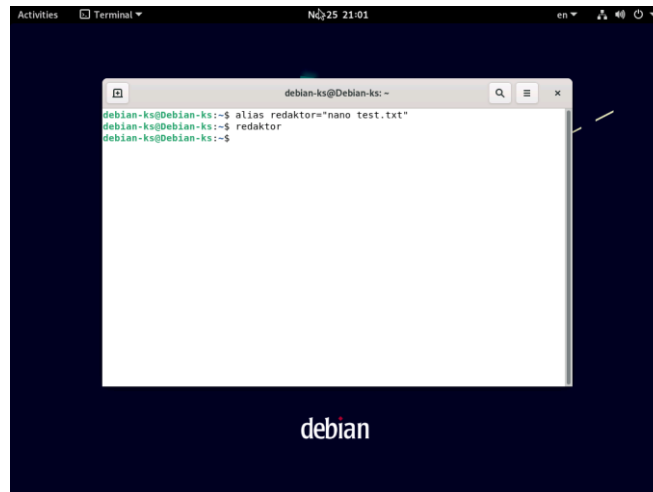


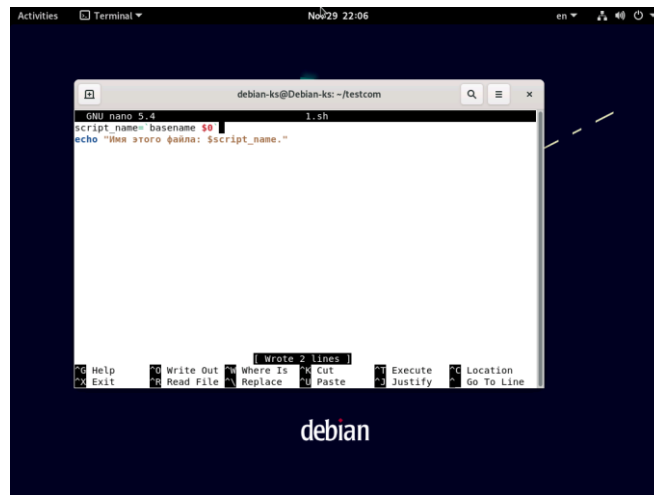
Рисунок 15. Можно создавать новые псевдонимы, просто выполняя эту команду в терминале. Но созданные таким образом алиасы `linux` будут работать только в этом терминале и только до его закрытия.



Рисунок 16. Результат алиаса `redaktor` создание и редактирование файла `test.txt`

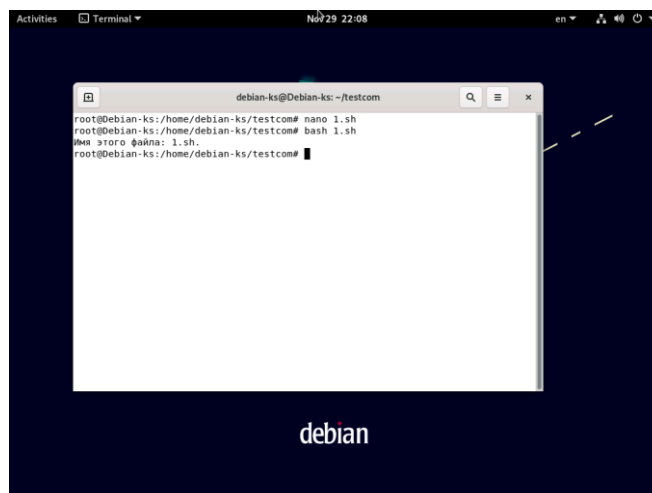
16) Командная подстановка -

Это подстановка результатов выполнения команды или даже серии команд; буквально, эта операция позволяет вызвать команду в другом окружении.



```
debian-ks@Debian-ks: ~/testcom
GNU nano 5.4 1.sh
script_name=basename $0
echo "Имя этого файла: $script_name."
[ Wrote 2 Lines ]
Help Write Out Where Is Cut Execute Location
Exit Read File Replace Paste Justify Go To Line
```

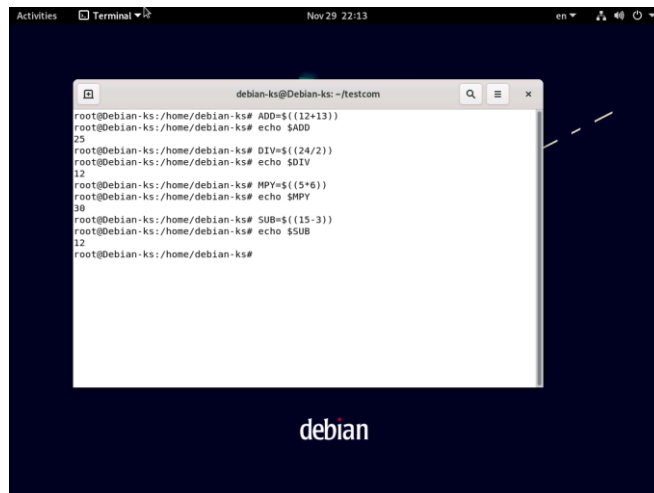
Рисунок 17. Создадим скрипт и попытаемся вывести название скрипта подстановкой переменной в echo



```
debian-ks@Debian-ks: ~/testcom
root@Debian-ks:~/home/debian-ks/testcom# nano 1.sh
root@Debian-ks:~/home/debian-ks/testcom# bash 1.sh
Имя этого файла: 1.sh
root@Debian-ks:~/home/debian-ks/testcom#
```

Рисунок 18. Результат выполнения скрипта подставил название файла с помощью переменной

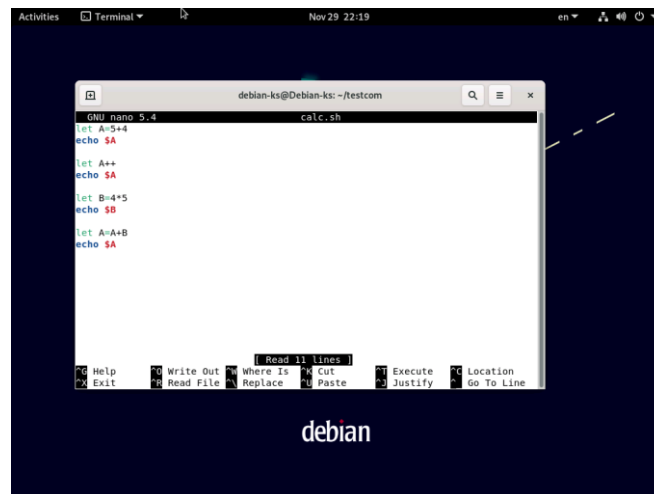
17) Арифметические выражения



```
root@Debian-ks:~/home/debian-ks# ADD=$((12+13))
root@Debian-ks:~/home/debian-ks# echo $ADD
25
root@Debian-ks:~/home/debian-ks# DIV=$((24/2))
root@Debian-ks:~/home/debian-ks# echo $DIV
12
root@Debian-ks:~/home/debian-ks# MPY=$((5*6))
root@Debian-ks:~/home/debian-ks# echo $MPY
30
root@Debian-ks:~/home/debian-ks# SUB=$((15-3))
root@Debian-ks:~/home/debian-ks# echo $SUB
12
root@Debian-ks:~/home/debian-ks#
```

debian

Рисунок 19. В bash существует множество способов выполнения арифметических операций. Пример в самом терминале




```
GNU nano 5.4 calc.sh
let A=4
echo $A
let A++
echo $A
let B=4*5
echo $B
let A=A+B
echo $A
```

Read 11 lines

Help Write Out Where Is Cut Execute Location
Exit Read File Replace Paste Justify Go To Line

debian

Рисунок 20. Реализуем скрипт с вычислениями

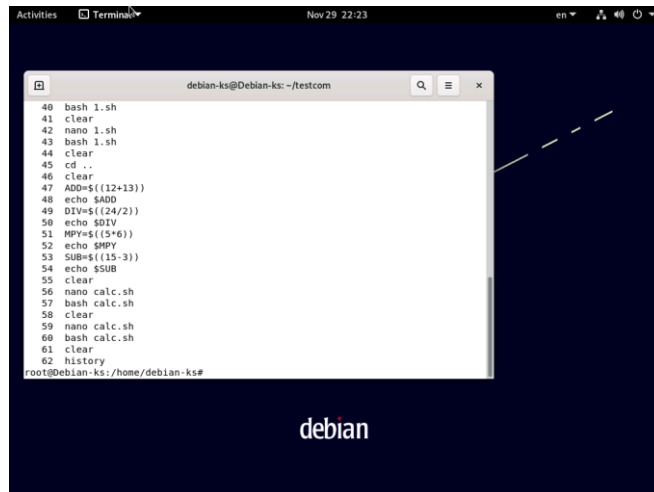


```
root@Debian-ks:~/home/debian-ks# nano calc.sh
root@Debian-ks:~/home/debian-ks# bash calc.sh
9
10
20
30
root@Debian-ks:~/home/debian-ks#
```

debian

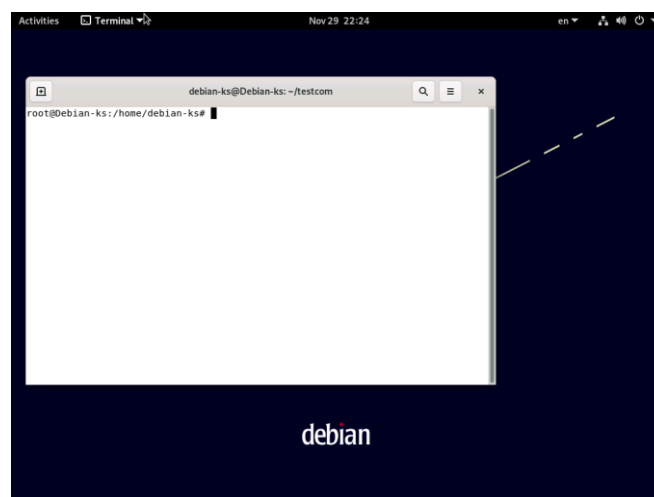
Рисунок 21. Результат выполнения скрипта

18) Команда clear



```
40 bash 1.sh
41 clear
42 nano 1.sh
43 bash 1.sh
44 clear
45 cd ..
46 clear
47 ADD=$((12+13))
48 echo $ADD
49 DIV=$((24/2))
50 echo $DIV
51 MP=$((5*6))
52 echo $MP
53 SUB=$((15-3))
54 echo $SUB
55 clear
56 nano calc.sh
57 bash calc.sh
58 clear
59 nano calc.sh
60 bash calc.sh
61 clear
62 history
root@debian-ks:/home/debian-ks#
```

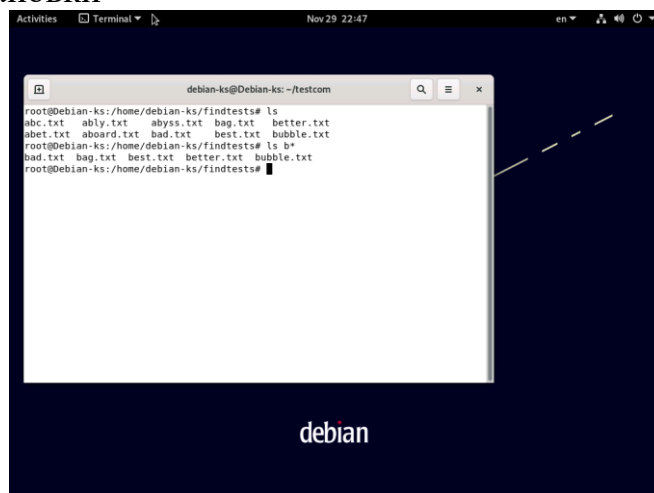
Рисунок 22. До использования команды clear



```
root@debian-ks:/home/debian-ks#
```

Рисунок 23. После использования команды clear

19) Шаблоны подстановки



```
root@debian-ks:/home/debian-ks/findtests# ls
abc.txt ably.txt abyss.txt bag.txt better.txt
abet.txt aboard.txt bad.txt best.txt bubble.txt
root@debian-ks:/home/debian-ks/findtests# ls b*
bad.txt bag.txt best.txt better.txt bubble.txt
root@debian-ks:/home/debian-ks/findtests#
```

Рисунок 24. Символ * — для замены нескольких символов (в том числе 0);

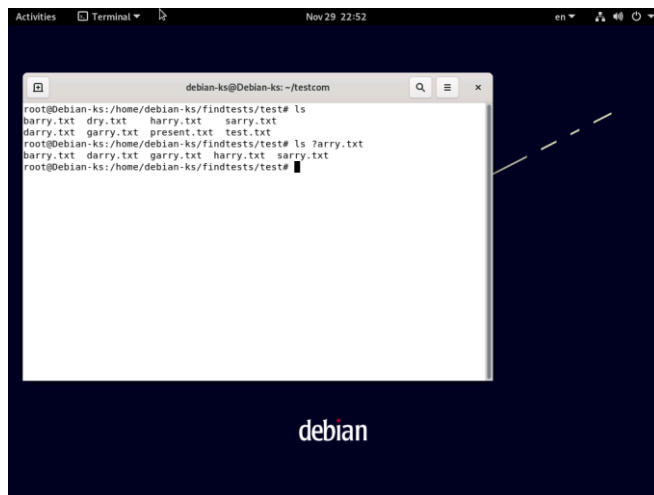


Рисунок 25. Символ ? — для замены одиночного символа;

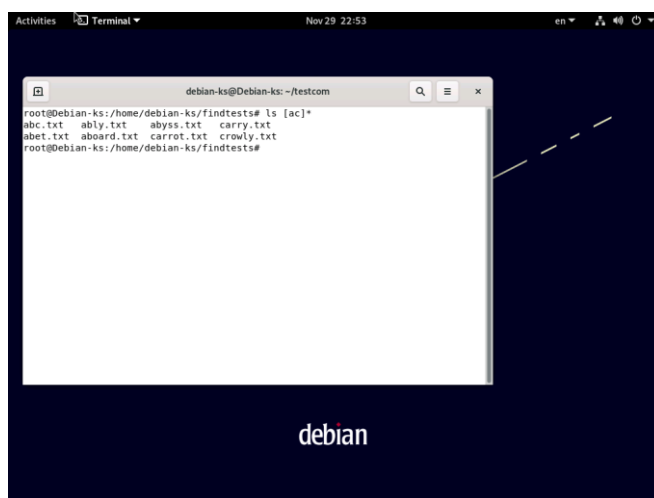


Рисунок 26. Символы [] — для замены определенного набора символов.

20) Help - справка man – подробная справка info – справка с гиперссылками



Рисунок 27. Результат команды help. Выводит команды и их описание. Справочник.

`help` — это встроенная команда в оболочке `bash` (и только в этой оболочке), которая документирует некоторые встроенные команды и ключевые слова этой оболочки. Это внутренняя система документации этой оболочки.

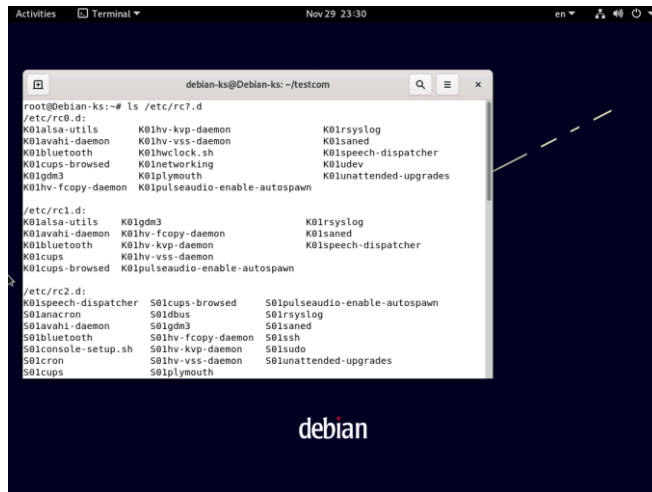
`man` — это общесистемная система документации, которая предоставляет краткие справочные руководства (страницы) для отдельных команд, функций API, концепций, синтаксиса файла конфигурации, форматов файлов, организованных в разделы (1 для пользовательских команд, 2 для системных вызовов...). Это традиционная система документации Unix.

`info` — это еще одна система документации, созданная в рамках проекта GNU. Это гипертекст со ссылками (предшествовал Интернету).

Информационное руководство похоже на цифровую книгу с концепцией оглавления и указателя (с возможностью поиска), который помогает находить информацию.

II

- 1) Составляем шаблон для любых имен файлов, вторая буква которых должна быть гласной буквой английского алфавита
`?[aeiouAEIOU]*.*`
- 2) Делаем тоже, что и в предыдущем задании, но так, чтобы вторая буква не была гласной, а могла быть любым другим символом
`?[^aeiouAEIOU]*.*`
- 3) Выводим с помощью шаблонов все имена файлов длиной шесть символов, последний из которых либо не цифра, либо буква A?
`?????[Aa].*`
- 4) Символ “ * ” внутри конструкции ...
Обозначает любое количество любых символов, в том числе и их отсутствие
- 5) Получаем список имен файлов в /etc, имена которых соответствуют шаблону `rc?.d`.



```
root@Debian-ks:~# ls /etc/rc?.d
/etc/rc0.d:
K01alsa-utils      K01hv-kvp-daemon      K01rsyslog
K01avahi-daemon    K01hv-vss-daemon      K01saned
K01bluetooth       K01hwclock.sh         K01speech-dispatcher
K01cups-browsed    K01networking         K01udev
K01qdm3            K01plymouth           K01unattended-upgrades
K01hv-fcopy-daemon K01pulseaudio-enable-autospawn

/etc/rc1.d:
K01alsa-utils      K01qdm3               K01rsyslog
K01avahi-daemon    K01hv-fcopy-daemon    K01saned
K01bluetooth       K01hv-kvp-daemon      K01speech-dispatcher
K01cups            K01hv-vss-daemon      K01pulseaudio-enable-autospawn
K01cups-browsed

/etc/rc2.d:
K01speech-dispatcher S01cups-browsed      S01pulseaudio-enable-autospawn
S01anacron           S01dbus              S01rsyslog
S01avahi-daemon     S01qdm3              S01saned
S01bluetooth        S01hv-fcopy-daemon  S01ssh
S01console-setup.sh S01hv-kvp-daemon    S01sudo
S01cron             S01hv-vss-daemon    S01unattended-upgrades
S01cups             S01plymouth
```

Рисунок 28. Результат команды `ls` с шаблоном `rc?.d`

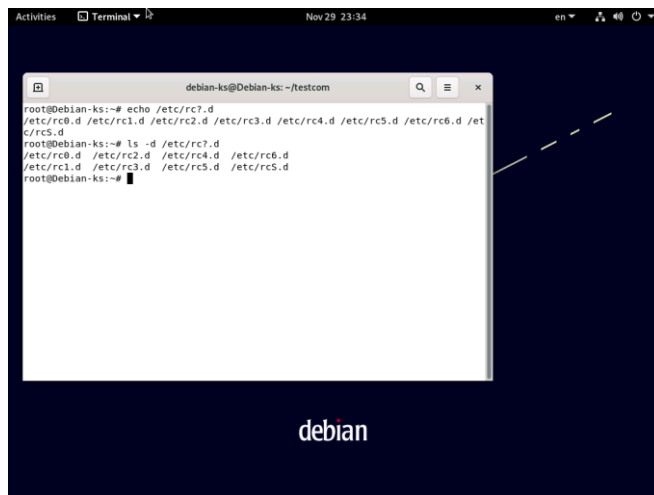
- 6) Делаем то же самое с помощью команды `echo`



```
root@Debian-ks:~# echo /etc/rc?.d
/etc/rc0.d /etc/rc1.d /etc/rc2.d /etc/rc3.d /etc/rc4.d /etc/rc5.d /etc/rc6.d /etc/rc7.d
root@Debian-ks:~#
```

Рисунок 29. Результат команды `echo` с шаблоном `rc?.d`

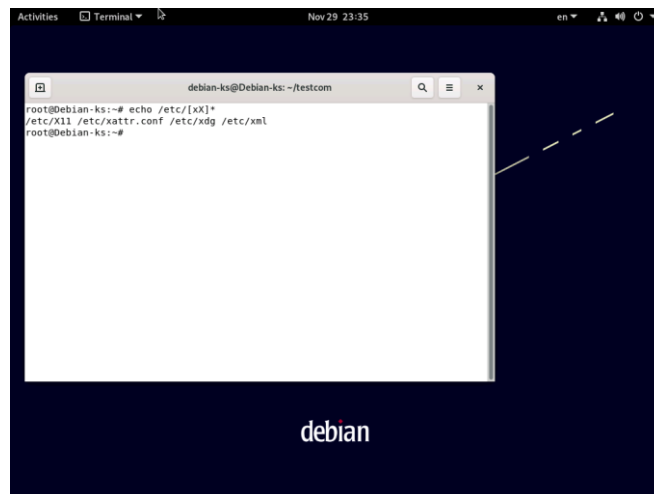
- 7) Делаем то же самое с помощью команды `ls` и `ls -d`



```
root@Debian-ks:~# echo /etc/rc?.d
/etc/rc0.d /etc/rc1.d /etc/rc2.d /etc/rc3.d /etc/rc4.d /etc/rc5.d /etc/rc6.d /etc/rc7.d
root@Debian-ks:~# ls -d /etc/rc?.d
/etc/rc0.d /etc/rc2.d /etc/rc4.d /etc/rc6.d
/etc/rc1.d /etc/rc3.d /etc/rc5.d /etc/rc5.d
root@Debian-ks:~#
```

Рисунок 30. Результат команды ls -d с шаблоном rc?.d

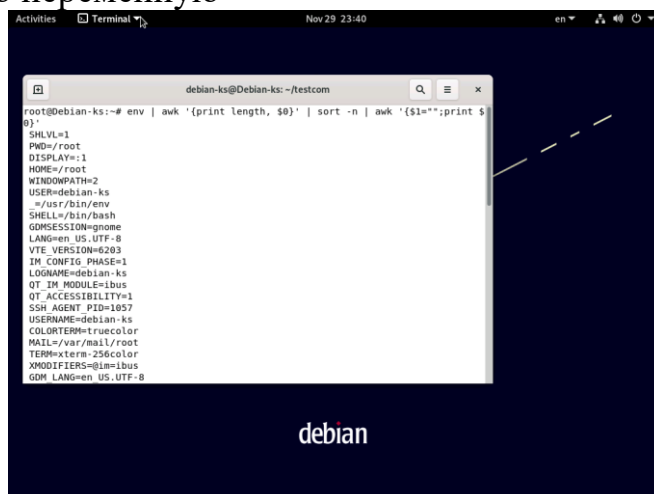
- 8) С помощью файлового шаблона подстановки и команды echo получаем список файлов в каталоге /etc, имена которых начинаются либо с символа x, либо с X



```
root@Debian-ks:~# echo /etc/[xX]*
/etc/X11 /etc/xattr.conf /etc/xdg /etc/xml
root@Debian-ks:~#
```

Рисунок 31. Результат команды echo с шаблоном [xX]*

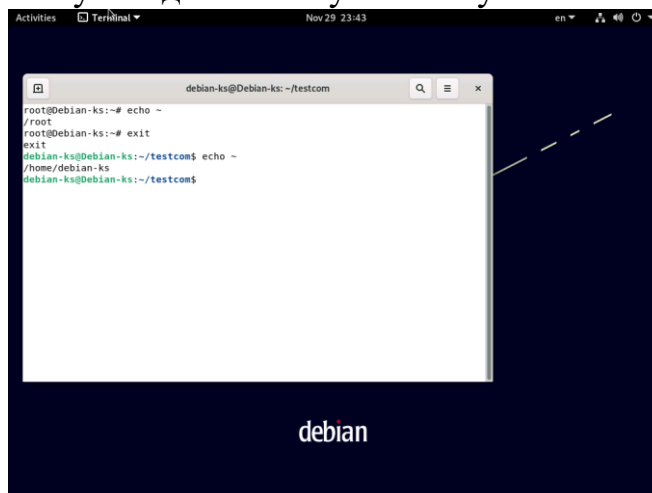
- 9) Посмотрим, какие переменные окружения заданы в нашей системе. Найдем самую короткую переменную



```
root@Debian-ks:~# env | awk '{print length, $0}' | sort -n | awk '{$1=""; print $0}'
SHLVL=1
PWD=/root
DISPLAY=:1
HOME=/root
WINDOWPATH=2
USER=debian-ks
=/usr/bin/env
SHELL=/bin/bash
GDMSESSION=gnome
LANG=en_US.UTF-8
VTE_VERSION=4203
IM_CONFIG_PHASE=1
LOGNAME=debian-ks
QT_IM_MODULE=ibus
QT_ACCESSIBILITY=1
SSH_AGENT_PID=1057
USERNAME=debian-ks
COLORTERM=truecolor
MAIL=/var/mail/root
TERM=xterm-256color
XMODIFIERS=@im=ibus
GDM LANG=en_US.UTF-8
```

Рисунок 32. Выведем все переменные окружения, заданные в системе и отсортируем их по возрастанию

10) Посмотрим, каков путь к домашнему каталогу




```
Activities Terminal Nov 29 23:43 en
debian-ks@Debian-ks: ~/testcom
root@Debian-ks:~# echo ~
/root
root@Debian-ks:~# exit
exit
debian-ks@Debian-ks:~/testcom$ echo ~
/home/debian-ks
debian-ks@Debian-ks:~/testcom$
```

debian

Рисунок 33. Посмотрим путь к домашнему каталогу пользователей root и debian-ks с помощью команды echo

11) Проверяем, как отреагирует система, если вручную изменить переменную окружения OLDPWD

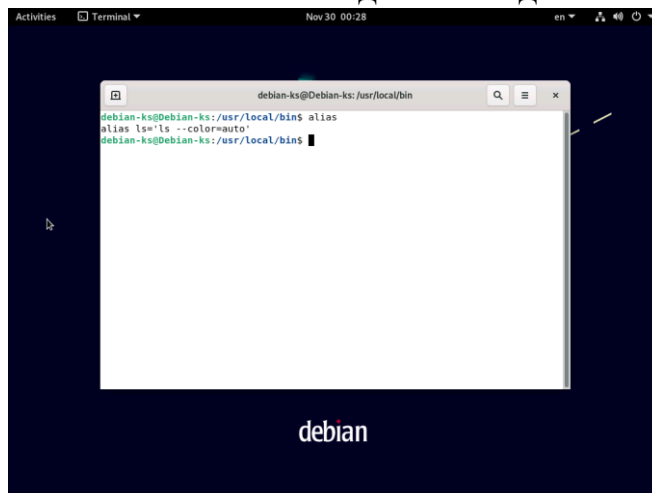


```
Activities Terminal Nov 30 00:26 en
debian-ks@Debian-ks: /usr/local/bin
debian-ks@Debian-ks:~$ export OLDPWD="/usr/local/bin"
debian-ks@Debian-ks:~$ cd -
/usr/local/bin
debian-ks@Debian-ks: /usr/local/bin$
```

debian

Рисунок 34. Изменим переменную OLDPWD чтобы попасть не в домашний каталог, а в папку bin

12) Посмотрим, какие в нашей системе заданы псевдонимы команд

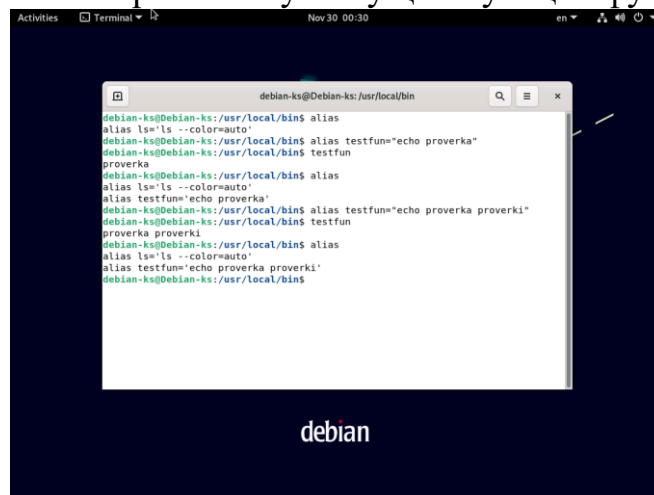


```
Activities Terminal Nov 30 00:28 en
debian-ks@Debian-ks: /usr/local/bin
debian-ks@Debian-ks: /usr/local/bin$ alias
alias ls='ls --color=auto'
debian-ks@Debian-ks: /usr/local/bin$
```

debian

Рисунок 35. Посмотрим какие псевдонимы команд заданы в нашей системе

13) Проверяем, как отреагирует система, если при создании псевдонима, в качестве его имени выбрать имя уже существующей функции



```
debian-ks@Debian-ks: /usr/local/bin
alias ls='ls --color=auto'
debian-ks@Debian-ks: /usr/local/bin alias testfun='echo proverka'
debian-ks@Debian-ks: /usr/local/bin testfun
proverka
debian-ks@Debian-ks: /usr/local/bin alias
alias ls='ls --color=auto'
alias testfun='echo proverka'
debian-ks@Debian-ks: /usr/local/bin alias testfun='echo proverka proverki'
debian-ks@Debian-ks: /usr/local/bin testfun
proverka proverki
debian-ks@Debian-ks: /usr/local/bin alias
alias ls='ls --color=auto'
alias testfun='echo proverka proverki'
debian-ks@Debian-ks: /usr/local/bin
```

Рисунок 36. Как видно на рисунке, при попытке указать уже существующее имя во время создания псевдонима, он её перезапишет

14) Создаем текстовый файл следующего содержания:

```
1+2
6*4
97% 12
43215/43*100
```



```
debian-ks@Debian-ks: ~
GNU nano 5.4 calc.txt
1+2
6*4
97% 12
43215/43*100
Wrote 4 lines
Help Write Out Where Is Cut Execute Location
Exit Read File Replace Paste Justify Go To Line
```

Рисунок 37. Создадим текстовый файл

15) Посчитаем все примеры из файла с помощью одной команды

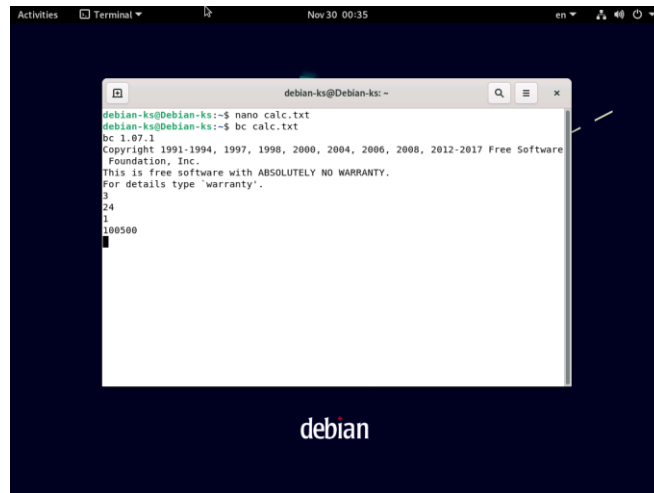


Рисунок 38. Используем команду `bc` чтобы посчитать все примеры из файла `calc.txt`

- 16) Проверим, есть ли разница по времени выполнения команд в виртуальном терминале и графическом
Разница неощутима
- 17) Проверим, как работает условный конвейер `&&`. Можно ли использовать его в псевдониме?

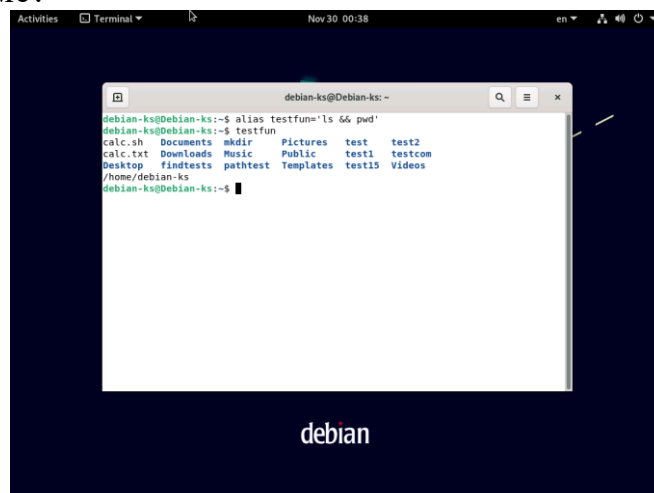


Рисунок 39. Создание и выполнение псевдонима с использованием условного конвейера `&&`



Рисунок 40. Попытка использовать условный конвейер && в имени псевдонима приводит к ошибке

18) Проверим, как работает условный конвейер ||. Можно ли использовать его в псевдониме?



Рисунок 41. Создание и выполнение псевдонима с использованием условного конвейера ||. Попытка использовать условный конвейер || в имени псевдонима также приводит к ошибке