“Київський фаховий коледж зв’язку”

Циклова комісія Комп’ютерної інженерії

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №3**

з дисципліни: «Операційні системи»

**Тема:** **“Знайомство з інтерфейсом та можливостями ОС Linux”**

Виконали студенти

групи КСМ-13а

Команда “Viper”:

Малієнко А. М.,

Мішин А. О.

та Нерощин Д.О.

Перевірив викладач

Сушанова В.С.

Київ 2022

Київ 2022

**Мета роботи:**

**1. Знайомство з базовими командами CLI-режиму в Linux.**

**2. Знайомство з базовими текстовими командами в термінальному режимі роботи в різних ОС.**

**Матеріальне забезпечення занять**

**1. ЕОМ типу IBM PC.**

**2. ОС сімейства Windows (Windows 7).**

**3. Віртуальна машина – Virtual Box (Oracle).**

**4. Операційна система GNU/Linux – CentOS.**

**Завдання для попередньої підготовки**

***Готував матеріал студент Малієнко Аліна***

***1. Вивчіть матеріали онлайн-курсу академії Cisco “NDG Linux Essentials”:***

- Chapter 5 - Command Line Skills

- Chapter 6 - Getting Help

***2. Пройдіть тестування у курсі NDG Linux Essentials за такими темами:***

- Chapter 05 Exam

- Chapter 06 Exam

***3. Дайте визначення наступним поняттям:***

1. Command Execution: The interpreter processes and executes commands entered by the user or launched from scripts.

2. Program Execution: Users can run executable files, programs, and services from the command line.

3. File and Directory Operations: The interpreter provides the ability to manage files and directories, such as creating, moving, deleting, and changing access rights

4. Input/Output Streams: The interpreter manages input (stdin) and output (stdout) streams for user interaction and command execution result processing.

5. Variables and Aliases: Users can define variables and aliases to simplify command invocation and store data.

6. Scripts: The command interpreter can execute scripts that contain sequences of commands and instructions for task automation.

***Оболонка–(Shell)*** in programming refers to an interactive interface that allows interacting with the operating system or executing commands and programs through a text-based interface. Shells in programming are used for automating operations, processing data, executing commands, and running scripts.

In the context of a command-line or script, the shell executes commands entered by the user or contained in executable files. The shell interprets and carries out the commands, ensuring their execution in the operating system.

Examples of shells in programming include Bash, PowerShell, Command Prompt, Python shell, and many others. Shells help automate tasks, create scripts, and interact with the operating system through a text-based interface.

"Сommand" - instructions or operators that instruct a computer to perform a specific action or operation. Commands in programming are used to carry out specific tasks or control the execution of a program.

Commands can be part of the program code and contain instructions for calculations, data reading and writing, control of program execution flow, and much more. Depending on the programming language, commands may have different syntax and functionality.

***4. Дайте відповіді на наступні питання:***

***- Яку базову інформацію надає рядок запрошення prompt?***

The prompt provides basic information for the user, typically displaying the current directory (working directory), user information, system information, instructions, or other contextual information that helps the user interact with the command interpreter.

***- Для чого команді потрібні параметри та аргументи?***

Commands require parameters and arguments to specify the actions they should perform. Parameters provide additional information to modify the behavior of a command, while arguments are the specific inputs or data that a command operates on.

***- Яке призначення команд ls, які параметри та аргументи вона може мати? Наведіть 3 приклади.***

The “ls” command in Bash is used to list files and directories in the current directory. It can take various parameters and arguments to customize its behavior. Here are three examples:

• “ls: Lists files and directories in the current directory.

• “ls -l: Lists files and directories in long format, showing detailed information.

• “Is -a /path/to/directory”: Lists all files and directories in the specified directory, including hidden ones.

***- Яким чином можна використати історію команд, які переваги це надає?***

You can use command history by pressing the "Up" arrow key on your keyboard to cycle through previously entered commands. This allows you to quickly access and reuse commands without retyping them. The advantages of using command history include increased efficiency, reduced typing errors, and the ability to recall and repeat complex commands.

***- Яке призначення команди echo?***

The “echo” command is used to display messages or text on the terminal. It is often used for printing output, displaying variables, or creating simple scripts. For example, “echo” "Hello, World!" will display "Hello, World!" on the terminal.

***- Охарактеризуйте поняття змінної в оболонці Bash, які типи змінних вона підтримує?***

In the Bash shell, a variable is a symbol that represents a value or text. Bash supports various types of variables, including:

• Scalar variables: Hold single values, such as numbers or strings.

• Array variables: Store multiple values in indexed arrays.

• Associative arrays: Store key-value pairs.

• Environment variables: Used to configure the shell's behavior and provide information to running programs.

***5. Підготувати в електронному вигляді початковий варіант звіту:***

- Титульний аркуш, тема та мета роботи

1. - Словник термінів
2. - Відповіді на п.4 та п.5 з завдань для попередньої підготовки
3. - Яке призначення команд env, export та unset?

The “env” command displays the current environment variables, while the “export” command is used to set or modify environment variables. Environment variables are global variables that are accessible to all processes running in the shell session. The “unset” command is used to remove environment variables, effectively unsetting their values. These commands are essential for configuring the shell's behavior and providing information to programs

man: The “man” command is used to display the manual pages (documentation) for various commands and system functions. You can use it as follows:

man command\_name

Replace “command\_name” with the name of the command you want to learn more about. You can navigate through the manual pages using arrow keys, and press “q” to exit.

“--help” option: Many command-line utilities and programs support the “—help” option, which provides a brief summary of the command's usage and available options. You can use it like this:

command\_name –help

Replace “command\_name” with the name of the command you want to get help for.

“Help” (for shell built-ins): If you're looking for help on shell built-in commands (commands that are part of the shell itself, such as “cd” or” echo”), you can use the “help” command followed by the built-in command's name. For example:

“help cd”

**Хід роботи**

***Готував матеріал студент Нерощин Д.***

**Хід роботи.**

**1. Опрацюйте всі приклади команд, що представлені у лабораторній роботі курсу NDG Linux Essentials -**

**Lab 5: Command Line Skills та Lab 6: Getting Help. Створіть таблицю для опису цих команд\*\*\***

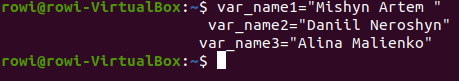
|  |  |
| --- | --- |
| **Назва команди** | **Її призначення та функціональність** |
| **ls** | **Displays information about directories and files. By default, without**  **arguments displays information for the current directory** |
| **ls -l** | **Using the -l option in the ls command allows you to display information**  **about the files located in the current working directory in a long**  **format, which provides more extensive additional information** |
| **ls -l /tmp** | **Using the /tmp argument in combination with the -l option in the ls command**  **command allows you to display detailed information about the files in the /tmp directory.** |
| ls -l /home | **to display detailed information about files in the /home directory** |
| whoami | **the command displays the same information as in the first part of the prompt** |
| uname | **command displays information about the current system; to see the name of the kernel you are using** |
| pwd | **The pwd command is used to display the current "location" or current "working" directory** |
| echo Hi  history | **Execute a new command and then execute the history command** |
| history 5 | **To view a limited number of commands, the history command can take a number as a parameter to display just that many recent entries** |
| !9 | **To run the command again, enter an exclamation point and the history list number** |
| echo Hello  Student | **The echo command can be used to print text and the value of a variable, as well as to show how the shell environment extends metacharacters** |
| **which** | **Use the which command to determine if an executable file, in this case named date, is located in the directory specified in the PATH value.** |
| **type command** | **The type command can be used to determine information about the type of command** |
| **which ls** | **Use the which command to display the full path to the ls command** |
| **echo Today is `date`** | **Run the following command to use back quotes `** |
| **echo Today is $(date)** | **You can also place $( before the command and ) after the command to perform a command substitution** |
| **echo D\***  **echo "D\*"** | **Double quote characters will affect wildcard characters by disabling their special meaning** |
| **false || echo Fail Or**  **true || echo Nothing to see here** | **The "or" symbols that separate the following commands demonstrate how an error before the "or" operator causes the command after it to be executed** |
| **man date** | **To learn more about the commands, go to the command's manual page using the man command** |
| **/file** | **Start searching forward for the word "file" by entering** |
| **man -k password** | **In some cases, you may not remember the exact name of the command. In these cases, you can use the -k option for the man command and provide the keyword argument** |
| **apropos password** | **is another way to view summaries of help pages by keyword** |
| **Whatis**  **passwd** | **Instead of using man -f to display all sections of the help page for a name, you can also use the whatis command** |
| **locate crontab** | **The easiest way to find a file is to use the locate command** |
| **whereis passwd** | **You may simply want to find where the team is located** |

***Готував матеріал студент Мішин А.***

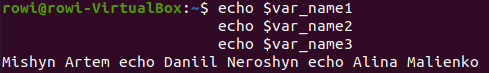
**2. Робота в в терміналі (закріплення практичних навичок) обов&#39;язково представити свої скріншоти:**

**2.1. Робота зі змінними (Variables) та псевдонімами (Aliases) в терміналі:**

**- Створіть змінні, що будуть містити Ваші імена та прізвища $var\_name1, $var\_name2, $var\_name3**

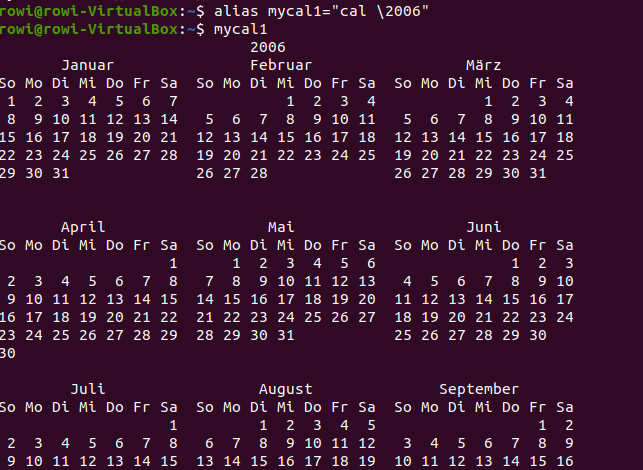


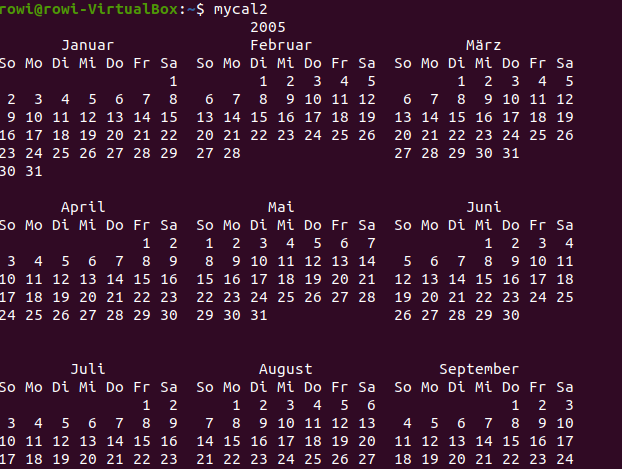
**- За допомогою команди echo виведіть імена студентів вашої команди**

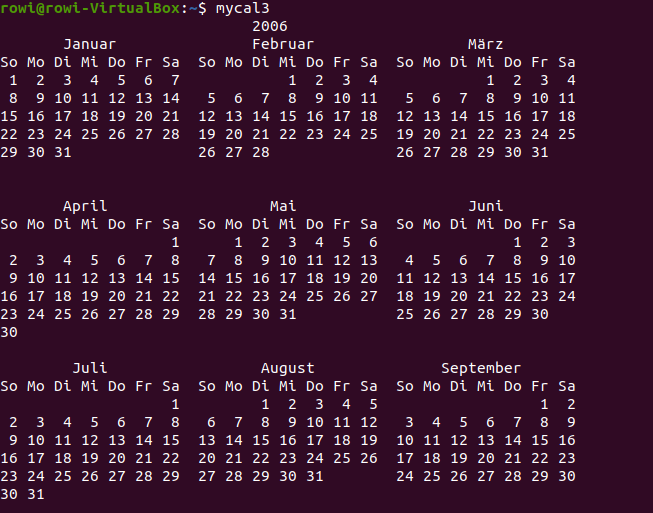


**- Створіть псевдоніми mycal1, mycal2, mycal3 для команди cal для автоматичного виведення**

**календарю вашого року народження**

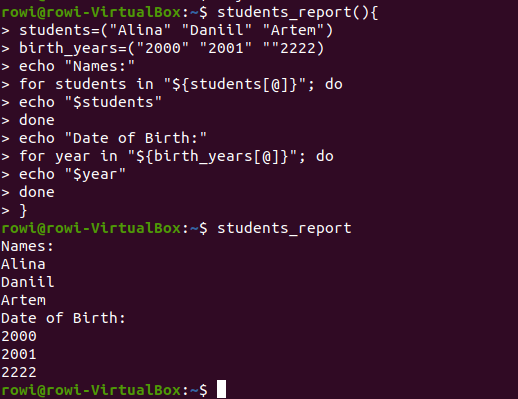






**2.2. Робота з функціями (Functions) в терміналі:**

**- Створіть функцію students\_report, що порядково буде виводити спочатку імена студентів Вашої команди, а потім роки їх народження**



**2.3. Робота з лапками (Quoting) в терміналі. Виведіть в командному рядку наступні речення:**

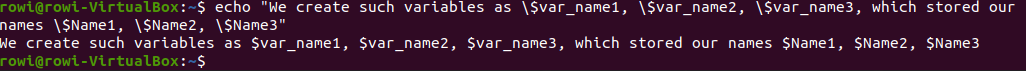
**- “We create such variables as $var\_name1, $var\_name2, $var\_name3, which stored our names Name1,**

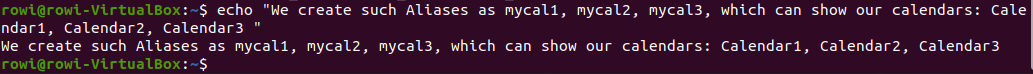
**Name2, Name3” (у реченні спочатку виводимо назви змінних, а потім їх вміст)**

**- “We create such Aliases as mycal1, mycal2, mycal3, which can show our calendars: Calendar1,**

**Calendar2, Calendar3” (у реченні спочатку виводимо назву команди-псевдонімів, потім вивід**

**цих команд).**

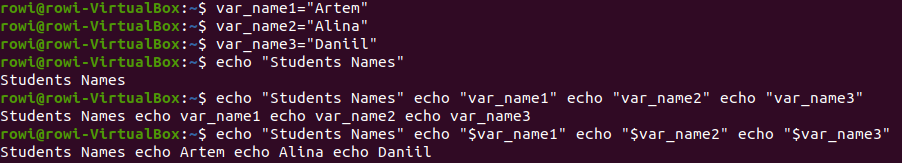


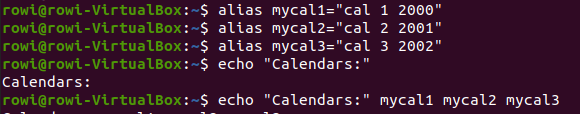


**2.4. Робота з інструкціями керування (Control Statements) в терміналі:.**

**- Чи можна завдання 2.1 та 2.2 ходу роботи виконати через інструкції керування без написання**

**окремої функції, як це буде виглядати?**





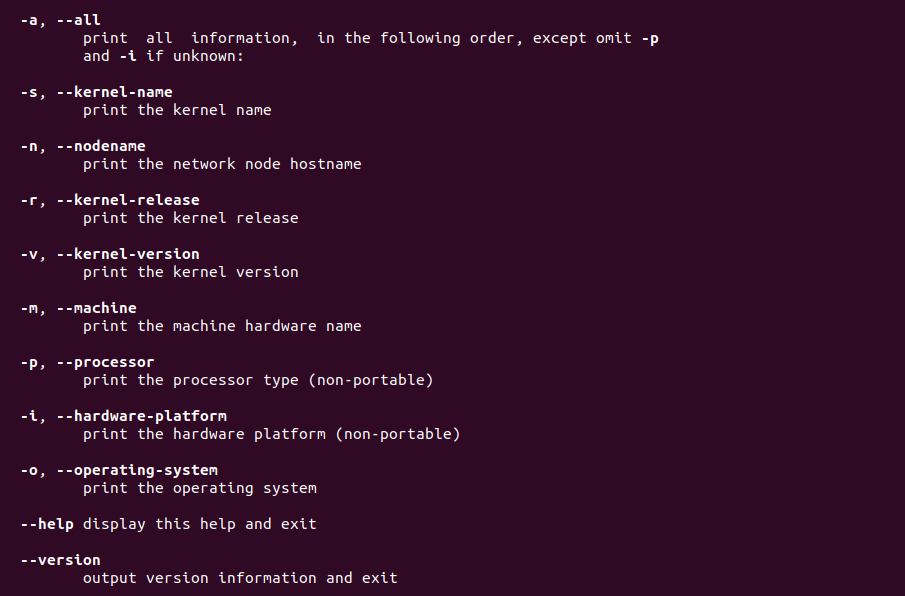
**2.5. Робота з командами довідки (Man Pages) в терміналі:.**

**- На прикладі команди uname продемонструйте як отримати довідку. На основі отриманої**

**додаткової інформації наведіть 5 різних варіантів виводу результату інформації по даній**

**команді з використанням 5 різних параметрів (Options)**

**Ось усі варіанти виводу результату команди uname :**



***Готував матеріал студент Малієнко Аліна***

1. **What types of commands exist in the Bash shell?**

Bash is an improved version of the well-known sh command shell (Bourne shell). Its main purpose is to provide interaction between the user and the operating system kernel. This interaction can take place in two ways. First, by using commands typed in the terminal (command line interface). Secondly, by running pre-prepared script files (shell scripts) for execution.

To provide these capabilities, bash has its own system of commands and language constructs. Internal commands, or those integrated into the shell, are located in the computer's RAM during operation, and external commands are stored as files in the system directory, usually in /usr/bin or /bin.

1.Built-in Commands: These are commands that are built directly into the shell itself. For example, cd, echo, alias, export, unset, and so on. These commands are executed without creating a new process.

2. External Commands: These are programs or scripts that are located on the system as separate executable files. To execute an external command, the shell starts a new process.

3.Pipeline Commands: Bash allows you to combine multiple commands together using the | (pipe) operator. For example, command1 | command2 passes the output of command1 as the input of command2.

4.I/O Redirection Commands: The shell allows you to redirect the input and output of a command using the >, >>, <, |, and so on operators. For example, command > output.txt redirects the output of a command to the file output.txt.

5.Process Control Commands: These commands allow you to start, pause, terminate, and control processes in the shell. For example, ps, kill, bg, fg.

6.Text Processing Commands: Bash provides a number of text processing commands, such as grep, sed, awk, which allow you to search, replace, and analyze texts.

Internal commands, in particular, allow you to perform the following actions:

* - Perform operations on the file system;
* - Working with variables;
* - Scenario management;
* - Manage running processes;

- Provides I/O from peripheral devices.

Bash scripts or shell scripts allow you to automate the system management process, from automating the process of updating packages to almost complete control over the operating system. Their advantages over some of their "competitors" include:

* - "Transparency" of the code for the user, as commands are stored in text format;
* - Structured scenario, i.e. it is possible to specify the sequence of commands;
* - Compatible with most \*nix systems.

**2.What are environmental variables? What are they? How can they be viewed in the terminal?**

I would like to highlight the following elements:

/ETC/PROFILE - is one of the system files. It is available to all users and the entire system even when logged in remotely. The only limitation for it is that the parameters are not accepted when opening the standard "Terminal", that is, in this location, no values from this configuration will work.

/ETC/ENVIRONMENT - is a broader analog of the previous configuration. It operates at the system level, has the same options as the previous file, but now without any restrictions, even when connected remotely.

/ETC/BASH.BASHRC - file is for local use only; it will not function in a remote session or connection via the Internet. It is performed for each user separately when creating a new terminal session.

.BASHRC - refers to a specific user, is stored in his home directory and is executed every time the terminal is launched again.

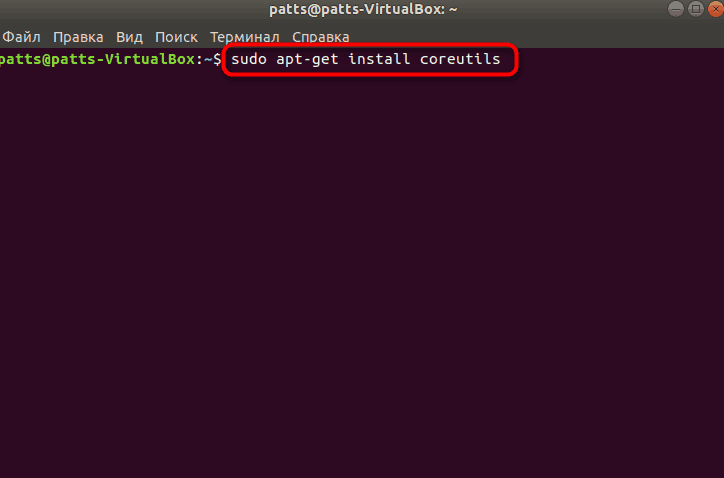
.BASH\_PROFILE - the same as .BASHRC, only for remote interaction, for example, when using SSH.

Environment variables in operating systems based on the Linux kernel are those variables that contain textual information used by other programs at startup. They usually include general system parameters of both the graphical and command shells, information about user settings, the location of certain files, and much more. The values of such variables are indicated, for example, by numbers, symbols, paths for folders or files. Thanks to this, many applications quickly access certain settings, and it becomes possible for the user to change or create new options.

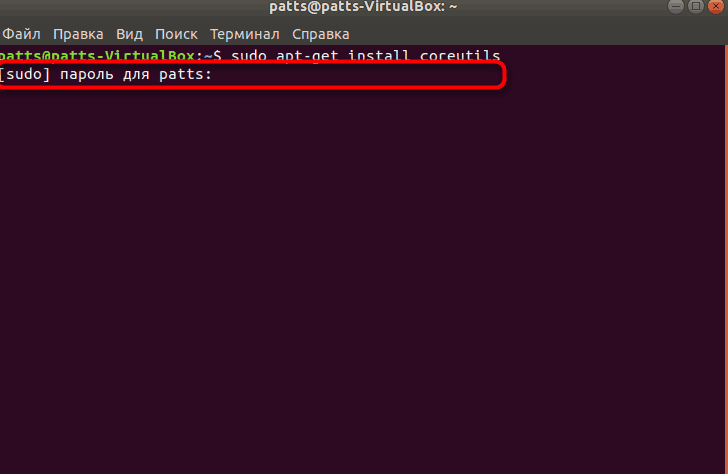
You can easily view all the system and user variables and their definitions in Linux with just one command that displays the list. To do this, you only need to perform a few simple actions through the standard console.



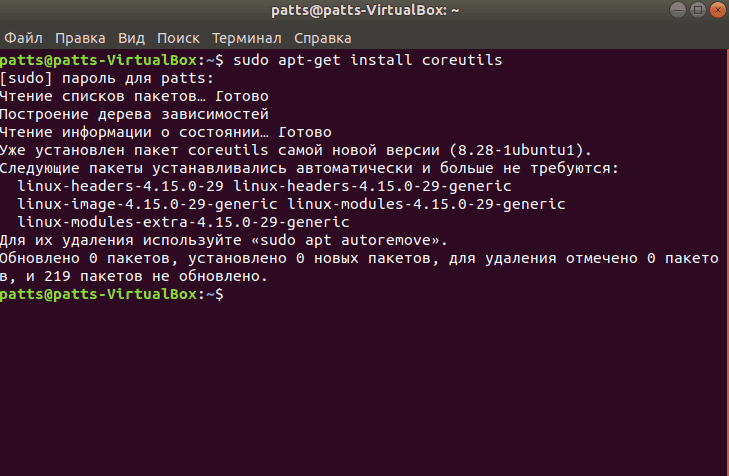
Run the command sudo apt-get install coreutils to check if this utility is available on your system and install it immediately if necessary.



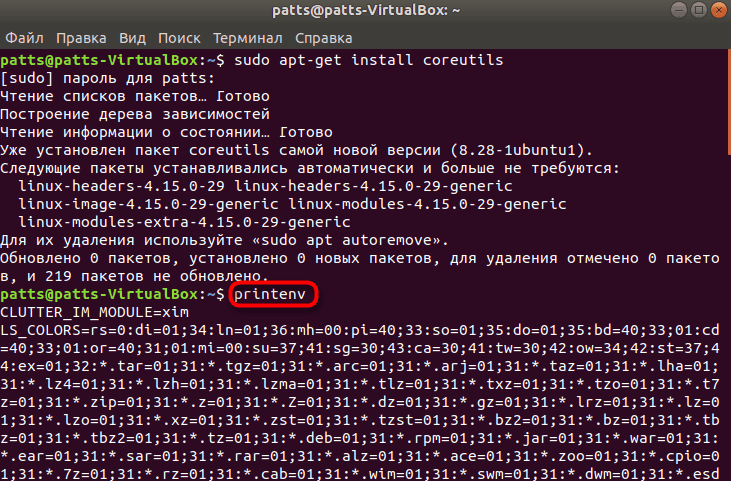
Enter the password for the root user account, the characters you enter will not be displayed.



You will be notified when new files are added or available in the libraries.



Now use one of the commands of the installed Coreutils utility to open the list of all environment variables. Write printenv and press the Enter key.



**3.Describe the variable $PS1. How to view its contents in the terminal?**

The Bash query configuration is stored in the PS1 variable. To save the contents of the PS1 variable to a new variable, run the following command:



Now you can set the PS1 variable to different values for your experiment. For example, the first line will set the prompt to the basic prompt "user $", and the second line will set the prompt to the basic query "user: working\_directory $".

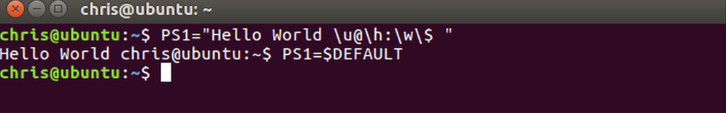


If you want to return to the default tooltip, just run the following command.



You can add any characters or text to the variable. So, you can use "Hello World" for the default tooltip prefix:





To view the contents of the $PS1 shell variable in the terminal, you can use the echo command. Here is an example:



This command displays the current contents of the $PS1 variable, which is responsible for the appearance of your command line (prompt) in the shell.

### 4.How can I change the value of the $PS1 variable? What will happen in the bash prompt line (the prompt line before each command)? How can I change the value of this variable not for the current session, but by default?

***Change the bash invitation for all time***

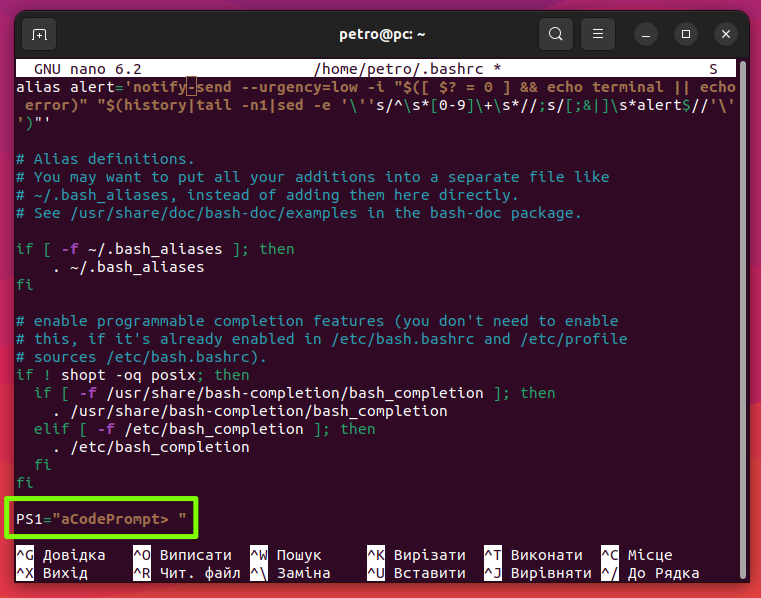
Open the bash configuration file for editing:

sudo nano ~/.bashrc

In this file, you will see several different settings. Some of them are currently inactive and are just comments - blue-green lines preceded by a # sign. White lines are used by the shell to customize its behavior.

Scroll to the end of the configuration file and add the following line:

PS1="aCodePrompt> "

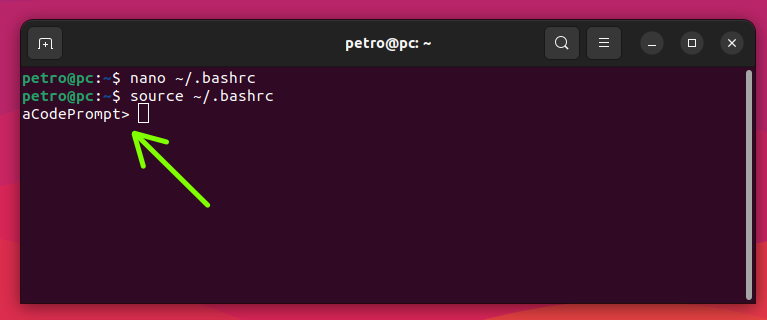


You can replace the aCodePrompt> part with any string of text you want.

Save the file (Ctrl+O followed by Enter) and exit the editor (Ctrl+X).

Update bash to apply the changes:source ~/.bashrc

As a result of executing these commands, the shell prompt should change to the following:

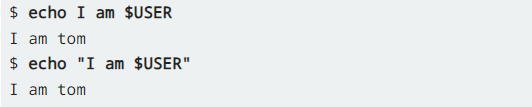


**5.What are quotation marks used for in the Bash shell?**

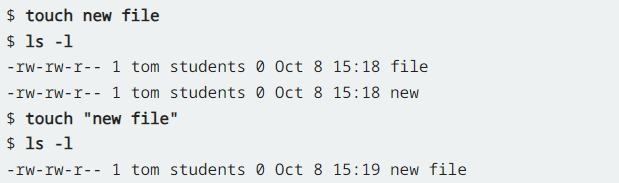
***Double quotes***

Double quotes tell the shell to treat text between the quotes ("...") as normal characters. All special characters are meaningless, except for $ (dollar sign), \ (backslash), and ` (backquote). This means that variables, command substitution, and arithmetic functions can still be used.

***For example, double quotes do not affect the replacement of the $USER variable:***



***The space, on the other hand, loses its meaning as an argument separator:***



As you can see, in the first example, the touch command creates two separate files because it interprets the two strings as separate arguments. In the second example, the command interprets both lines as one argument, so it creates only one file. However, it is best to avoid spaces in file names. Instead, you can use an underscore (\_) or a period (.).

***Single quotes***

Single quotes have no exceptions for specific characters, unlike double quotes. They cancel any special meaning of each character. Let's take one of the first examples from above:



When you use single quotes, you see a different result:



Now the command displays the exact string without replacing the variable.

**Conclusions.**

In the course of the laboratory work, I have studied ... , in more detail theoretically studied the issue of familiarity with the basic commands of the CLI mode in Linux, familiarity with basic text commands in terminal mode in different OS. Material support of classes

1. Computer such as IBM PC.

2. Windows family OS (Windows 7).

3. Virtual machine - Virtual Box (Oracle).

4. GNU/Linux operating system - CentOS.