“Київський фаховий коледж зв’язку”

Циклова комісія Комп’ютерної та програмної інженерії

ЗВІТ ПО ВИКОНАННЮ

ЛАБОРАТОРНОЇ РОБОТИ №7

з дисципліни: «Операційні системи»

Тема: “Створення скриптових сценаріїв та визначення апаратної конфігурації системи”

Виконали

студенти

групи РПЗ-03

Команда 6: Sichkar Maxim,

Brytyuk Bohdan

Kyiv 2023

**The goal of the work:**

1. Getting practical skills for working with the Bash command shell.

2. Getting to know basic actions when working with script scenarios.

**Material provision of classes**

1. IBM PC type computer.

2. OS family Windows (Windows 7).

3. Virtual machine - Virtual Box (Oracle).

4. GNU/Linux operating system - CentOS.

5. Cisco network academy site netacad.com and its online Linux coursesTasks for preliminary preparation.

Tasks for preliminary preparation.

**1. Read the short theoretical information for the laboratory work and make a small dictionary basic English terms for command assignments and their parameters.**

**2. Study the materials of the online course of the Cisco Academy “NDG Linux Essentials”:**

- Chapter 11 - Basic Scripting

- Chapter 12 - Understanding Computer Hardware

**3. Take the test in the NDG Linux Essentials course on the following topics:**

- Chapter 11 Exam

- Chapter 12 Exam

**4. On the basis of the considered material, answer the following questions:**

**4.1. Describe the concept of a shell script.**

A shell script is a computer program written in a scripting language that runs within a Unix/Linux shell environment. It is essentially a set of commands that are executed sequentially by the shell. The shell script can contain various control structures such as loops, conditionals, and functions to perform specific tasks.

Shell scripts are commonly used for automating tasks, such as system administration tasks, file manipulation, and other repetitive tasks. They allow users to automate tasks that would otherwise require manual intervention, and can significantly increase productivity.

The shell script can be created using any text editor and saved with a ".sh" file extension. The first line of a shell script starts with the shebang, which specifies the interpreter to be used for executing the script. For example, #!/bin/bash specifies that the script is written in the Bash shell scripting language.

Shell scripts can be executed in a number of ways, including by typing the filename on the command line, or by calling it from another script or program. The script can also be made executable using the chmod command, which allows the user to run the script by typing its name on the command line, without specifying the shell interpreter.

**4.2. How are scripts created and edited, what do I need to do to run the script?**

Scripts can be created and edited using any text editor, such as Notepad, Sublime Text, Vim, or Emacs. To create a new script, simply open a text editor and start writing the commands you want to include in the script. Once you have written the script, you can save it with a filename and a ".sh" extension to indicate that it is a shell script.

To run the script, you need to make it executable. To do this, you can use the chmod command followed by the filename of the script. For example, if your script is named "myscript.sh", you can make it executable by running the following command in the terminal:

chmod +x myscript.sh

This command adds the execute permission to the script, allowing you to run it as a program. Once you have made the script executable, you can run it by typing the following command in the terminal:

./myscript.sh

This will execute the script and run the commands included in the script file. If you want to run the script from a different directory, you can specify the full path to the script instead of using "./myscript.sh".

It's important to note that the script needs to be written in a compatible scripting language for the shell environment you are using. For example, if you are using the Bash shell, your script should be written in Bash scripting language.

**4.3. What are the main components of a motherboard do you know?**

The motherboard is the main circuit board of a computer system, and it is responsible for connecting and communicating with various components of the computer system. The main components of a motherboard are:

* CPU Socket: The CPU (Central Processing Unit) socket is the place where the processor is installed on the motherboard.
* Chipset: The chipset is a collection of chips that control the communication between the CPU and other components of the motherboard, such as RAM, storage devices, and expansion cards.
* RAM Slots: The RAM (Random Access Memory) slots are used to install memory modules that provide temporary storage for data that the CPU is currently using.
* Expansion Slots: The expansion slots are used to add additional components to the motherboard, such as graphics cards, sound cards, and network cards.
* Power Connectors: The power connectors on the motherboard are used to provide power to the various components of the motherboard, including the CPU and expansion cards.
* BIOS (Basic Input/Output System): The BIOS is a small chip on the motherboard that provides the initial boot-up instructions for the computer system.
* CMOS Battery: The CMOS (Complementary Metal-Oxide-Semiconductor) battery is a small battery on the motherboard that provides power to the CMOS memory, which stores the BIOS settings and system clock.
* SATA and IDE Connectors: The SATA and IDE connectors are used to connect storage devices such as hard drives and optical drives to the motherboard.
* USB Connectors: The USB (Universal Serial Bus) connectors on the motherboard are used to connect external devices such as keyboards, mice, and printers.
* Audio and Network Connectors: The audio and network connectors on the motherboard are used to connect speakers, headphones, microphones, and network cables to the computer system.

**4.4. Briefly describe for which devices the concepts of MBR and GPT are used?**

MBR (Master Boot Record) and GPT (GUID Partition Table) are partitioning schemes used to organize and manage the partitions on a hard drive.

MBR is an older partitioning scheme that is used for BIOS-based computer systems. It has a limit of four primary partitions and a maximum disk size of 2 terabytes. MBR is commonly used for Windows-based systems and some Linux distributions.

GPT is a newer partitioning scheme that is used for UEFI-based computer systems. It can support up to 128 partitions and a maximum disk size of 9.4 zettabytes. GPT is becoming more common as more systems switch to UEFI boot mode, and it is required for booting from a disk larger than 2 terabytes.

Both MBR and GPT are used for hard disk drives (HDDs) and solid-state drives (SSDs) used in desktop and laptop computers. They are not used for other devices such as USB drives, memory cards, or external hard drives, which typically use a different partitioning scheme.

**4.5. What is the essence of the mounting operation, why is it needed?**

Mounting is the process of making a file system accessible at a certain point in the directory tree of an operating system. In simpler terms, it means connecting a storage device such as a hard drive, USB drive, or CD-ROM to a directory in the file system so that the operating system can access the files and folders stored on that device.

The essence of the mounting operation is to make the file system on the storage device available for use by the operating system and other applications running on the computer. Without mounting, the file system on the storage device would not be visible or accessible to the operating system, and it would not be possible to read or write data to the storage device.

Mounting is needed because it allows the operating system to manage and organize files stored on different storage devices in a unified and consistent manner. By mounting a storage device at a specific directory in the file system, the files and folders on that device become part of the overall file system hierarchy, and they can be accessed and managed just like any other file or folder on the computer.

Mounting also allows the operating system to control access to the storage device and ensure that multiple applications do not access the same file or folder simultaneously, which could cause data corruption or loss.

**5. Prepare the initial version of the report electronically:**

- Title page, topic and purpose of the work

- Glossary of terms

- Answers to p. 4.1 and p. 4.5 from tasks for preliminary preparation

Progress.

**1. Initial work in CLI mode in Linux OS of the Linux family:**

1.1. Start the VirtualBox virtual machine, select CentOS and run it. Log in

under user: CentOS, password for login: reverse (if you run LR in 401 aud.) and run

terminal.

1.2. Start the Ubuntu\_PC virtual machine (if you are doing the LR tasks through the netacad academy)

1.3. Start your Linux family operating system (if you are working on your own PC and its

installed) and launch the terminal.

2. Work through all the command examples presented in the labs of the NDG Linux Essentials course - Lab 11: Basic Scripting and Lab 12: Understanding Computer Hardware. Create a table to describe these teams\*\*\*

|  |  |
| --- | --- |
| The name of the command | Its purpose and functionality |
| lscpu | lscpu is a powerful and flexible utility for understanding the CPU architecture of a system, and it can be useful for system administrators, software developers, and anyone else who needs to understand the processing power and capabilities of a system. |
| head | The head command is a common command-line utility in Unix and Unix-like operating systems that is used to display the first few lines of a file or output stream. Its purpose is to allow users to quickly preview the contents of a file without having to open it in a text editor or viewer.  The head command can also be used in combination with other commands in pipelines to display the first few lines of the output of a command. For example, ls -l | head -n 5 would display the first 5 lines of the output of the ls -l command. |
| lspci | lspci is a command-line utility in Linux and other Unix-like operating systems that is used to display information about all PCI (Peripheral Component Interconnect) buses and devices connected to them in the system. The utility is typically used to obtain information about hardware devices connected to the system, such as network adapters, sound cards, graphic cards, and USB controllers.  The information provided by lspci can be useful for identifying and troubleshooting hardware issues in the system. For example, if a network adapter is not working, running lspci can provide information about the make and model of the adapter, which can be used to search for a suitable driver or firmware update.  lspci can be run with root or administrator privileges, and it is typically pre-installed in most Linux distributions. You can run man lspci to see the manual page and all available options. |
| lsusb | lsusb is a command-line utility used in Linux and Unix systems to list the USB devices currently connected to the system. The lsusb command stands for "List USB".  When lsusb is executed, it communicates with the USB controller in the system and retrieves information about each connected USB device. This information includes the device vendor and product ID, the USB bus and device number, the USB version, the device speed, and other details about the USB device.  The output of the lsusb command is displayed in a table format, with each USB device represented as a row in the table. The columns in the table provide information about the device ID, vendor, and product information, as well as the USB bus and device numbers.  lsusb is commonly used by system administrators and users to diagnose and troubleshoot issues with USB devices, such as verifying if a device is connected to the system, determining if the correct drivers are installed, or identifying issues with USB port connectivity. |
| lsmod | lsmod is a Linux command used to display the currently loaded kernel modules. Kernel modules are pieces of code that can be dynamically loaded and unloaded into the kernel at runtime. They are used to add support for new hardware devices, add new features to the kernel, or modify the behavior of the kernel.  When you run the lsmod command, it will display a list of all the currently loaded kernel modules along with some information about each module, such as the module name, the number of instances of the module in use, and the list of other modules that depend on it.  The lsmod command can be useful for debugging and troubleshooting hardware-related issues, as well as identifying which kernel modules are currently loaded and in use. Additionally, it can help you determine if a particular module is causing a conflict with other modules or with the kernel itself. |

\*\*\*Screenshots of the execution of commands in the terminal may not be presented, it is enough to briefly describe the commands in the table.

**3. Create script scripts with the output of text messages for the user (demonstrate screenshots):**

- the script should output a greeting to the current user indicating the current date and information about the current system;

- the script should output information about the hardware configuration of the current system (use commands discussed in Lab 12).

**Control questions**

1. **How can scripts handle variables and create branched and looping scripts?**

Scripts can handle variables and create branched and looping scripts using programming constructs such as conditional statements and loops.

Variables in a script are used to store values that can be used later in the script. To create a variable in a script, you can simply assign a value to a name, like this:

variable\_name=value

To use the value of a variable, you can refer to its name using the $ symbol, like this:

echo $variable\_name

Conditional statements, such as if-else and case statements, are used to execute different parts of the script based on certain conditions. For example, the following if statement will execute a command only if a condition is true:

if [ $variable\_name -gt 10 ]

then

echo "Variable is greater than 10"

fi

Loops, such as for and while loops, are used to execute a block of code repeatedly. For example, the following for loop will execute a command for each value in a list:

for i in 1 2 3 4 5

do

echo $i

done

By combining these programming constructs, you can create more complex scripts that handle variables and perform branched and looping operations. For example, you can use an if statement inside a loop to execute a command only for certain values of a variable:

for i in 1 2 3 4 5

do

if [ $i -lt 3 ]

then

echo $i

fi

done

This script will print the values 1 and 2, because the if statement will only execute for values less than 3.

1. **What is the difference between arch and lscpu commands?**

The arch and lscpu commands are both used to display information about the system architecture and processor, but they provide different types of information.

The arch command displays the architecture of the system, which can be either 32-bit or 64-bit. It is a simple command that outputs the system architecture of the machine it is run on, like this:

$ arch

x86\_64

This output indicates that the system architecture is 64-bit.

On the other hand, the lscpu command displays detailed information about the processor(s) on the system, including its architecture, clock speed, number of cores, cache size, and other details. It provides much more detailed information than the arch command.

1. **Which command can be used to get information about the status of RAM usage by the current system?**

The free command is commonly used to get information about the status of RAM usage by the current system.

The free command displays the total amount of memory, the amount of memory used, the amount of memory available for new processes, and other related statistics.

The free command also provides additional options to display memory statistics in different units, such as bytes or megabytes, and to display the information in a more human-readable format. You can run man free to see the manual page and all available options.

1. **What commands can be used to view the connection status of peripheral devices in terminals?**

* lsusb: This command is used to display information about USB buses and the devices connected to them. It can display information such as the device ID, manufacturer, and model.
* lspci: This command is used to display information about PCI buses and the devices connected to them. It can display information such as the device ID, manufacturer, and model.
* dmesg: This command displays the system log, which includes information about device drivers and devices being detected and configured by the system.
* lsblk: This command displays information about block devices (such as hard drives and USB storage devices), including their connection status and mount points.
* lsmod: This command displays information about kernel modules, which are used to manage hardware devices and their drivers.

1. **What are the features of the gparted screen saver?Conclusion: we get practical skills for working with the Bash command shell, acquaintance with basic commands for data archiving and compression, get to know the basic actions when working with text in the terminal.**

My conclusion seems to be describing the skills and knowledge that can be gained through learning and practicing Bash command shell. These skills include data archiving and compression using commands such as tar, gzip, and bzip2, as well as working with text in the terminal using commands such as grep, sed, and awk. By acquiring these skills, you can become more proficient in using the terminal for various tasks, such as system administration, software development, and data analysis.

**Conclusion:**