“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»**

Виконала студентка

групи РПЗ-13а

Балджі В.В.

Перевірив викладач

Сушанова В.С.

Київ 2024

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

**Мета роботи:**

1. Отримання практичних навиків роботи з командною оболонкою Bash.
2. Знайомство знайомство з базовими діями при роботі зі скриптовими сценаріями.

**Матеріальне забезпечення занять:**

1. ЕОМ типу IBM PC.

2. ОС сімейства Windows та віртуальна машина Virtual Box (Oracle).

3. ОС GNU/Linux (будь-який дистрибутив).

4. Сайт мережевої академії Cisco netacad.com та його онлайн курси по Linux

**Завдання для попередньої підготовки:**

1. \*Прочитайте короткі теоретичні відомості до лабораторної роботи та зробіть невеликий словник базових англійських термінів з питань призначення команд та їх параметрів.

|  |  |
| --- | --- |
| **Термін англійською** | **Термін українською** |
| merits | переваги, заслуги |
| steep learning curve | стрімкий процес навчання |
| accordingly | відповідно |
| shebang (or crunchbang) | *#!* |
| loop | петля, цикл |
| consistency | узгодженість |
| to ensure | забезпечувати |

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

* Chapter 11 - Basic Scripting
* Chapter 12 - Understanding Computer Hardware

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

* Chapter 11 Exam
* Chapter 12 Exam

1. На базі розглянутого матеріалу дайте відповіді на наступні питання:
   1. \*Охарактеризуйте поняття скриптового сценарію у командній оболонці.

A shell script is a file of executable commands that has been stored in a text file.

* 1. \*Яким чином створюються та редагуються скрипти, що треба зробити щоб запустити скрипт?

To create a script, open an editor (nano, vim), enter the script, save the file, and grant execution rights using the chmod command. To edit, open the script in the editor and make changes. To run the script, write ./ before the file name.

* 1. \*\*Які основні компоненти материнської плати ви знаєте?

CPU socket, chipset, memory slots, expansion slots, I/O ports, bios, power connectors, cable connectors.

* 1. \*\*Коротко охарактеризуйте для яких пристроїв оперують поняттями MBR та GPT?

MBR (Master Boot Record) and GPT (GUID Partition Table) are two different partition table formats used to organize and manage partitions on a hard disk.

* 1. \*\*В чому суть операції монтування, для чого вона потрібна?

Mounting file systems is an important operation for accessing data on external storage devices or internal disk partitions. It allows the operating system to interact with these devices, giving users the ability to read, write, and edit files and folders.

1. Підготувати в електронному вигляді початковий варіант звіту:

* Титульний аркуш, тема та мета роботи
* Словник термінів
* Відповіді на п.4.1 та п.4.5 з завдань для попередньої підготовки

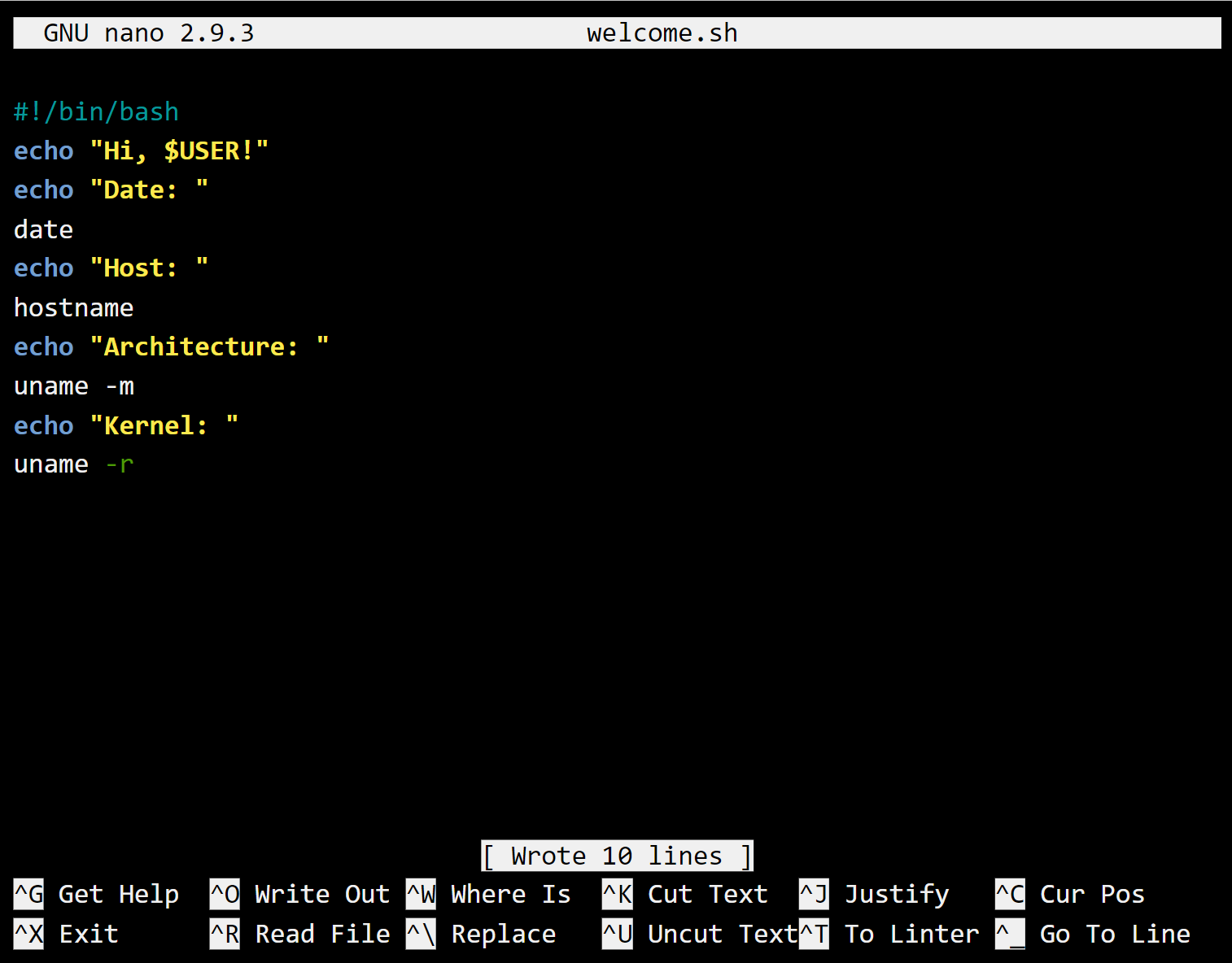
**Хід роботи:**

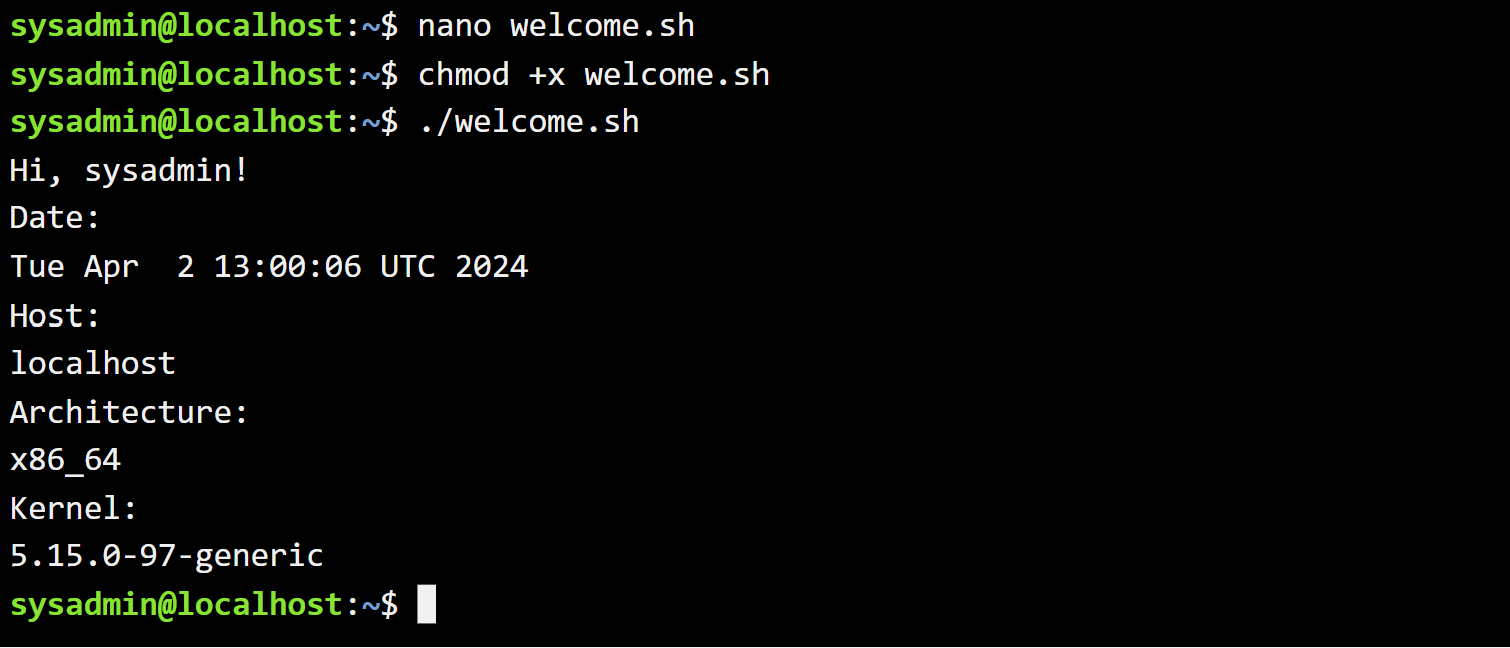
* 1. Початкова робота в CLI-режимі в Linux ОС сімейства Linux:
  2. Запустіть віртуальну машину VirtualBox, оберіть CentOS та запустіть її. Виконайте вхід в систему під користувачем: CentOS, пароль для входу: reverse ***(якщо виконуєте ЛР у 401 ауд.)*** та запустіть термінал.
  3. Запустіть віртуальну машину Ubuntu\_PC ***(якщо виконуєте завдання ЛР через академію netacad)***
  4. Запустіть свою операційну систему сімейства Linux ***(якщо працюєте на власному ПК та її встановили)*** та запустіть термінал.
  5. Опрацюйте всі приклади команд, що представлені у лабораторних роботах курсу ***NDG Linux Essentials - Lab 11: Basic Scripting*** та ***Lab 12: Understanding Computer Hardware.*** Створіть таблицю для опису цих команд

|  |  |
| --- | --- |
| Назва команди | Її призначення та функціональність |
| vi myfile | To create a new file, execute this command. |
| :wq | Type :wq to write the file to disk and quit. |
| vi myfile | Invoke the vi editor to modify the file you created. |
| 3G (hold down the number "3" key and press the letter "g")  k  8l (that's the number eight followed by the letter "l") | Move the cursor to the beginning of the word "very" by pressing the keys. |
| dw | Delete the word “very” by issuing the command dw (**d**elete **w**ord). |
| u | Undo the last operation. |
| 2dw | Delete two words. |
| xxxx | Delete four characters, one at a time. |
| 4u | Undo the last 4 operations and recover the deleted characters. |
| 14x | Delete 14 characters. |
| 5X | Delete the five characters to the left of the cursor (type 5 then **Shift+x**). |
| dd | Delete the current line. |
| p | Paste the deleted lines below the current line. |
| 2u | Undo the last two operations. |
| 2dd | Delete two lines, the current and the next. |
| 4w  D | Move to the fourth word then delete from the current position to the end of the line **Shift+D.** |
| J | Join two lines, the current and the next by typing a capital J (**Shift+J**). |
| yw | Copy (or “yank”) the current word. |
| P | Paste (or “put”) the copied word before the current cursor by typing **Shift+p.** |
| 1G  3J | Move to the first line, then join three lines. |
| :%s/text //g | Search for and delete the word text (add a space after the word text). |
| i | Navigate to beginning of file, then press i to enter insert mode to add text. |
| ESC | Exit insert mode and return to command mode by pressing the **Escape** key. |
| l  ~ | Move forward one space by pressing the lower case l to place the cursor on the W and toggle it to lower case by pressing the tilde (~). |
| :w | Type :w and the **Enter** key. |
| j | Move down to second line. |
| 10l | 10 followed by the lowercase ‘L’. |
| a | Enter insert mode. |
| text | Text followed by a space. |
| o | Open a blank line below the current line by typing a lowercase letter o. |
| vi myfile | Open myfile. |
| 3G  2dd | Navigate to the third line, and delete the third and fourth lines. |
| :q! | Quit the vi editor without saving your changes |
| /line | Search forward for the word line. |
| n | Search for the next instance of the word line by pressing the letter n. |
| ?line | Search backward for the word line. |
| cw  entry | You will replace the word line with the word entry. When you press cw you will be in insert mode and you will be able to type over the word line. |
| I | Add text at the beginning of a line. Enter insert mode again and add a line by pressing upper case i. |
| 2G  A  [Space]Indeed! | Add text at the end of a line (uppercase A). First move to the second line and add the phrase Indeed!. |
| :x | Save your changes and exit vi. |
| echo "Hello there! Here is the calendar for this month:"  cal | Create a file called sample.sh and add the following lines. |
| #!/bin/bash | To make it clear that this is a BASH shell script, you need to include a special line at the top of the file called a "shbang" (or "shebang"). This line starts with #! and then contains the path to the BASH shell executable. Add the following line at the top of the sample.sh file. |
| bash sample.sh | One way that you can run this program is by typing bash before the filename. |
| ls -l sample.sh  chmod a+x sample.sh  ls -l sample.sh  ./sample.sh | You can avoid having to type bash in front of the filename by making the file "executable" for all users. |
| vi sample.sh |  |
| echo "Today is" `date +%A` |  |
| cat sample.sh  ./sample.sh | Execute the sample.sh file. |
| sample.sh | Execute the following to see how the shell would fail to find the file if you don't use the ./. |
| echo $PATH | Execute the following to see the $PATH variable for the sysadmin account. |
| mkdir bin  mv sample.sh bin  sample.sh | Note that /home/sysadmin/bin is one of the directories in the $PATH variable. This is a great place to put your shell scripts. |
| vi drive.sh | Create the drive.sh file. |
| #!/bin/bash | Begin by placing the following in drive.sh. |
| cat drive.sh  chmod a+x drive.sh  ./drive.sh | Then make the file executable and run it. |
| if [ $age -lt 16 ] | Modify the if line of drive.sh. |
| cat drive.sh  ./drive.sh | Then run the program again. |
| vi check.sh | Create a file called check.s. |
| #!/bin/bash  echo "Enter a username to check: "  read name  if grep $name /etc/passwd > /dev/null  then  echo "$name is on this system"  else  echo "$name does not exist"  fi | #!/bin/bash: This line specifies which command line shell should be used to execute the script. In this case, the Bash shell is used.  echo "Enter a username to check: ": This command displays a line prompting you to enter a username.  read name: This command waits for the user to enter a value and writes it to the name variable.  if grep $name /etc/passwd > /dev/null: This command uses grep to search for the entered username in the /etc/passwd file, which stores information about system users. The > /dev/null option sends all of the grep output to black holes, i.e. discards it because we don't need to display the result on the screen.  then: This command marks the beginning of a block of code that is executed if the previous command was successful (i.e., returned output 0).  echo "$name is on this system": This command displays a message indicating that the entered username was found on the system.  else: If the grep command fails (i.e., does not find the entered username), then the block of commands after else is executed.  echo "$name does not exist": This command displays a message stating that the entered user does not exist on the system.  fi: This line marks the end of the if-else block, which means the end of the logical structure of the conditional statement. |
| cat check.sh  chmod a+x check.sh  ./check.sh | cat check.sh: This command displays the contents of a file named check.sh on the screen. The check.sh file is a Bash script that has been created or already exists on the system.  chmod a+x check.sh: This command gives the check.sh file the execution privileges for all (a - all). The +x option sets the file's execution privileges.  ./check.sh: This command executes the check.sh file. The ./ symbol before the filename means that the check.sh file is in the current directory. Also, since we have granted the file execution rights, we can directly call it as an executable. |
| for name in /etc/passwd /etc/hosts /etc/group  do  wc $name  done | This allows you to perform a set of operations on each value. |
| ls  for num in `seq 1 12`  do  touch test$num  done  ls | Run the following on the command line to create 12 files named test1, test2, test3, etc. (up to test12). |
| lscpu | In order to determine the type of CPU execute the lscpu command. |
| head -n 20 /proc/cpuinfo | Use the head command with the -n option to list the first 20 lines of the cpuinfo file. |
| free -m  free -g | To discover how much RAM and swap space is being used, use the free command. |
| lspci | To see what devices are connected to the PCI bus, use the lspci command. |
| lspci -k | Use the lspci command with the -k option to show devices along with the kernel driver and modules used |
| lsusb | Attempt to list the USB connected devices. |
| lsmod | Use the lsmod command to view the currently loaded modules. |
| fdisk -l | Execute the fdisk command to list the disk devices. |

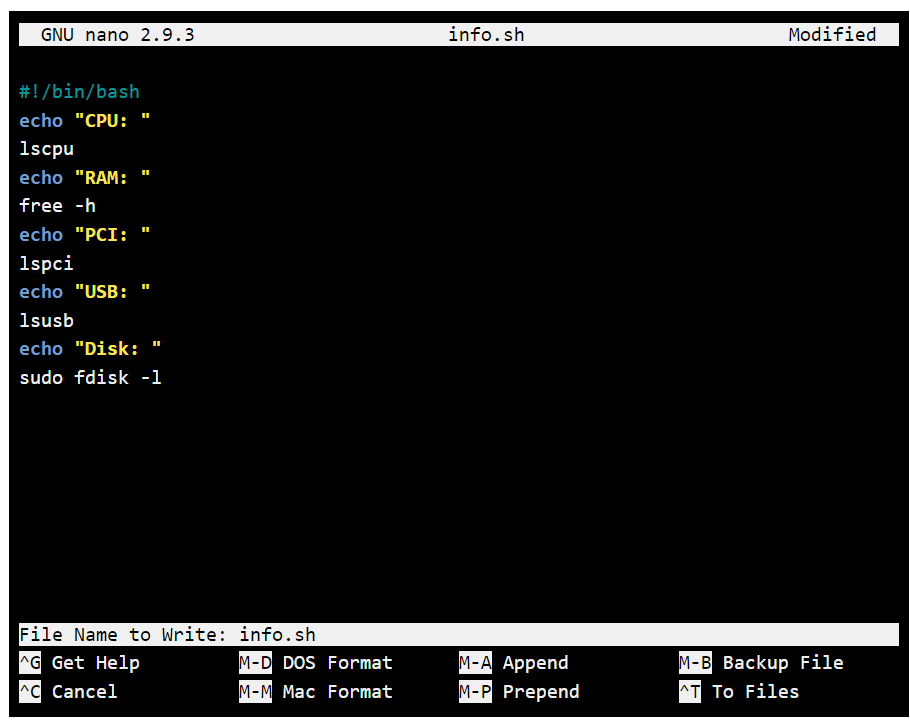
**Примітка:** **Скріншоти** виконання команд в терміналі можна **не представляти**, достатньо **коротко описати команди в таблиці**.

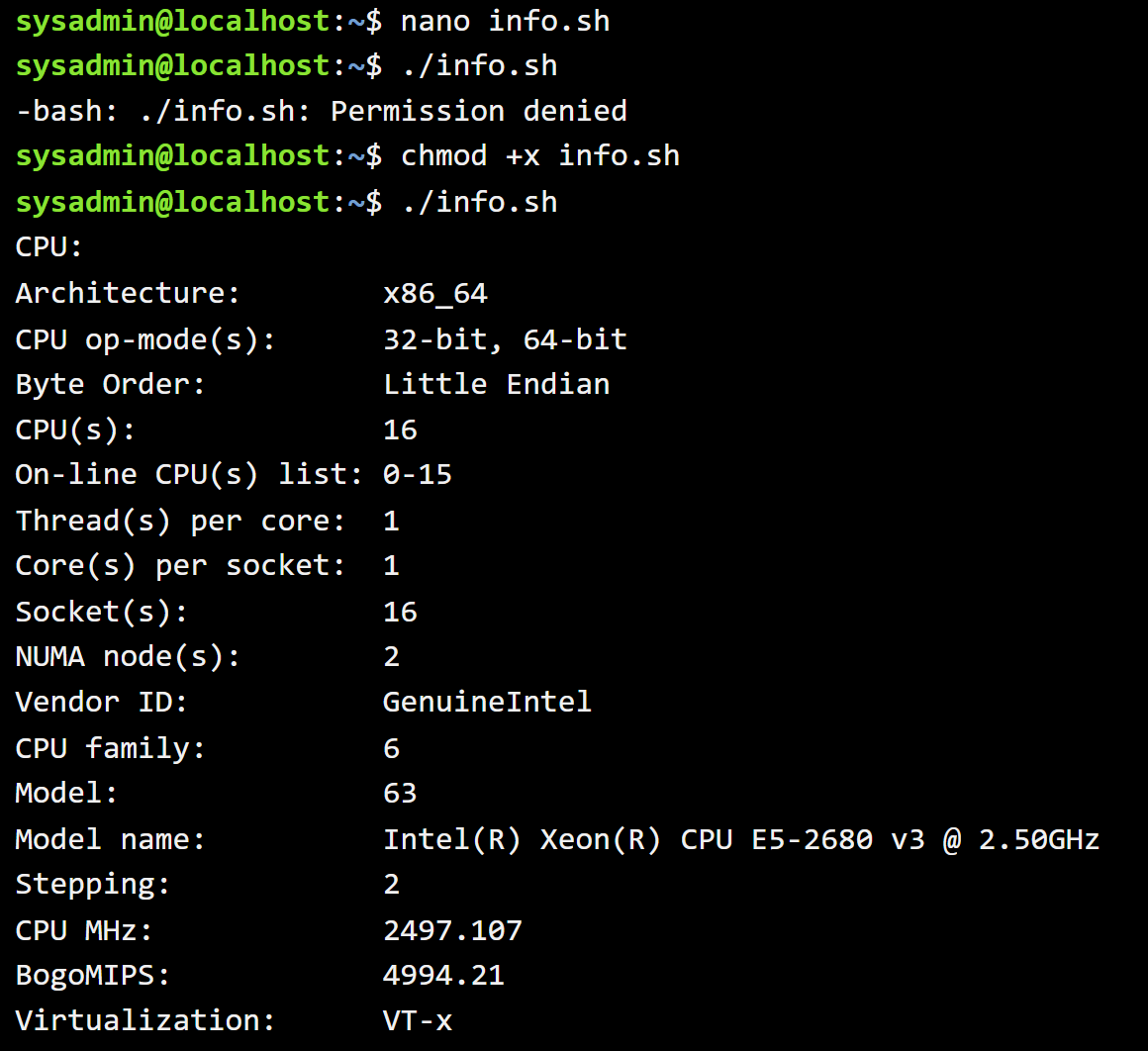
* 1. Створіть скриптові сценарії з виводом текстових повідомлень для користувача (продемонструйте скріншоти):
* сценарій має виводити привітання до поточного користувача вказуючи поточну дату та інформацію про поточну систему;

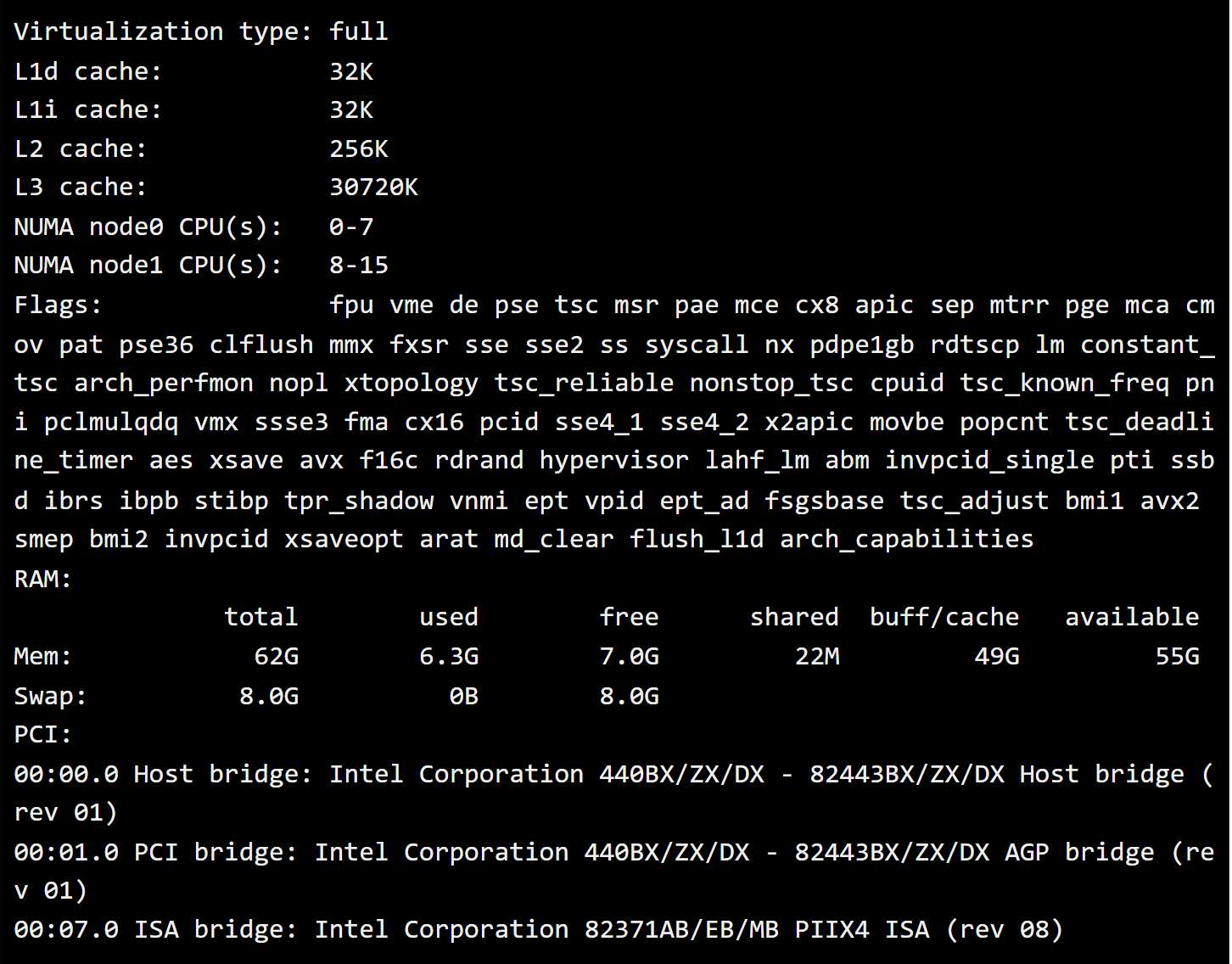


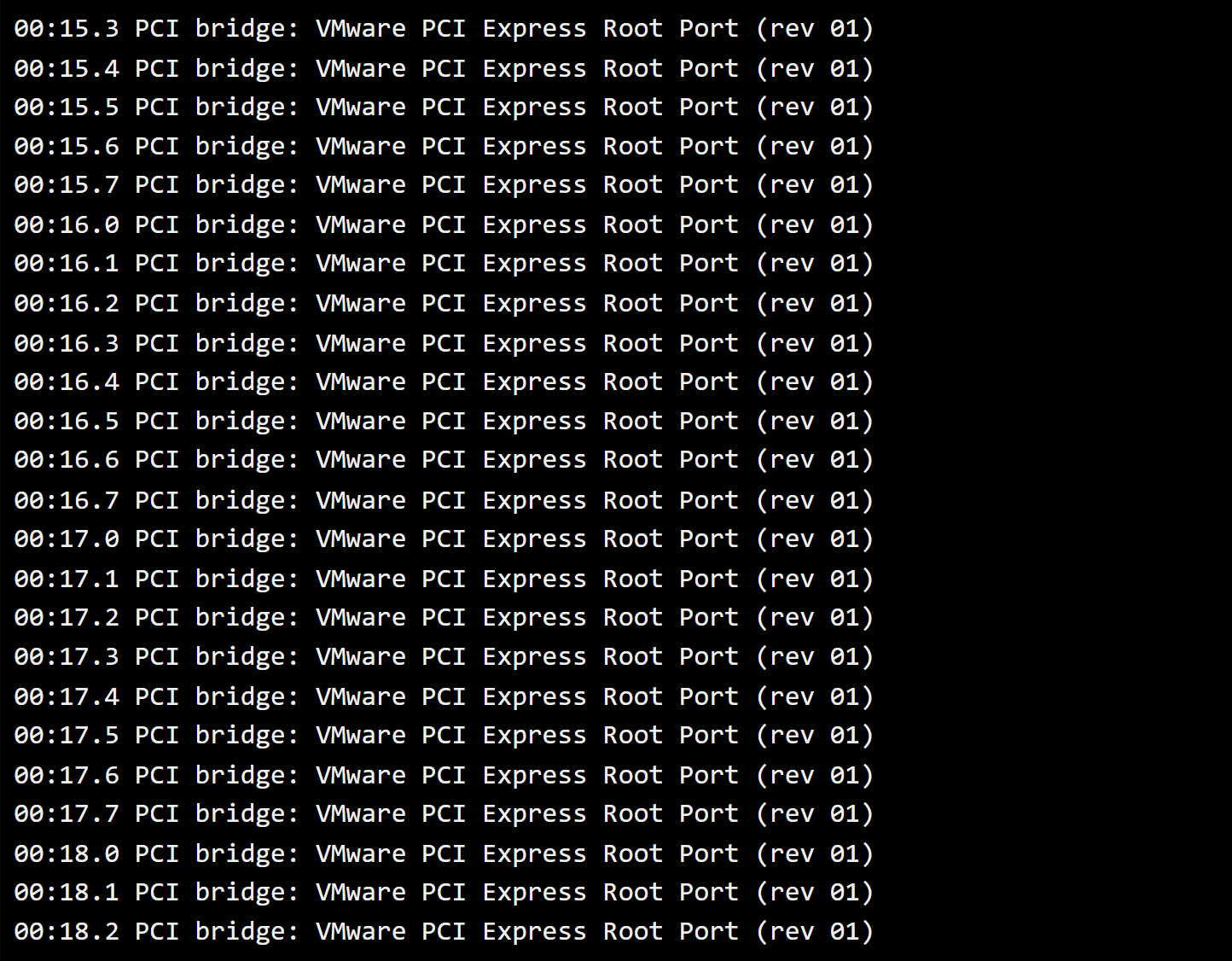


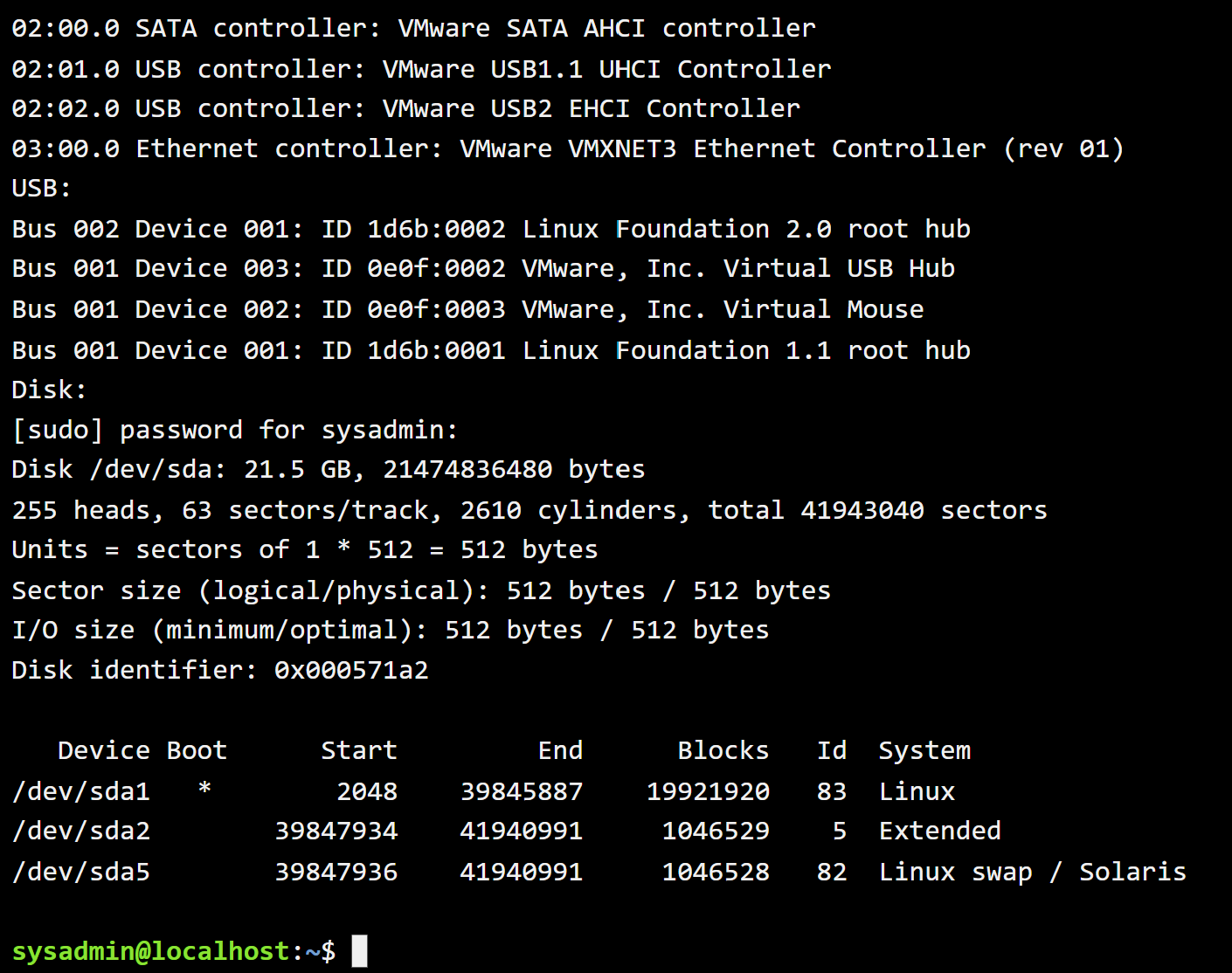
* \*сценарій має виводити інформацію про апаратну конфігурацію поточної системи (використовуйте команди розглянуті в Lab 12: Understanding Computer Hardware);



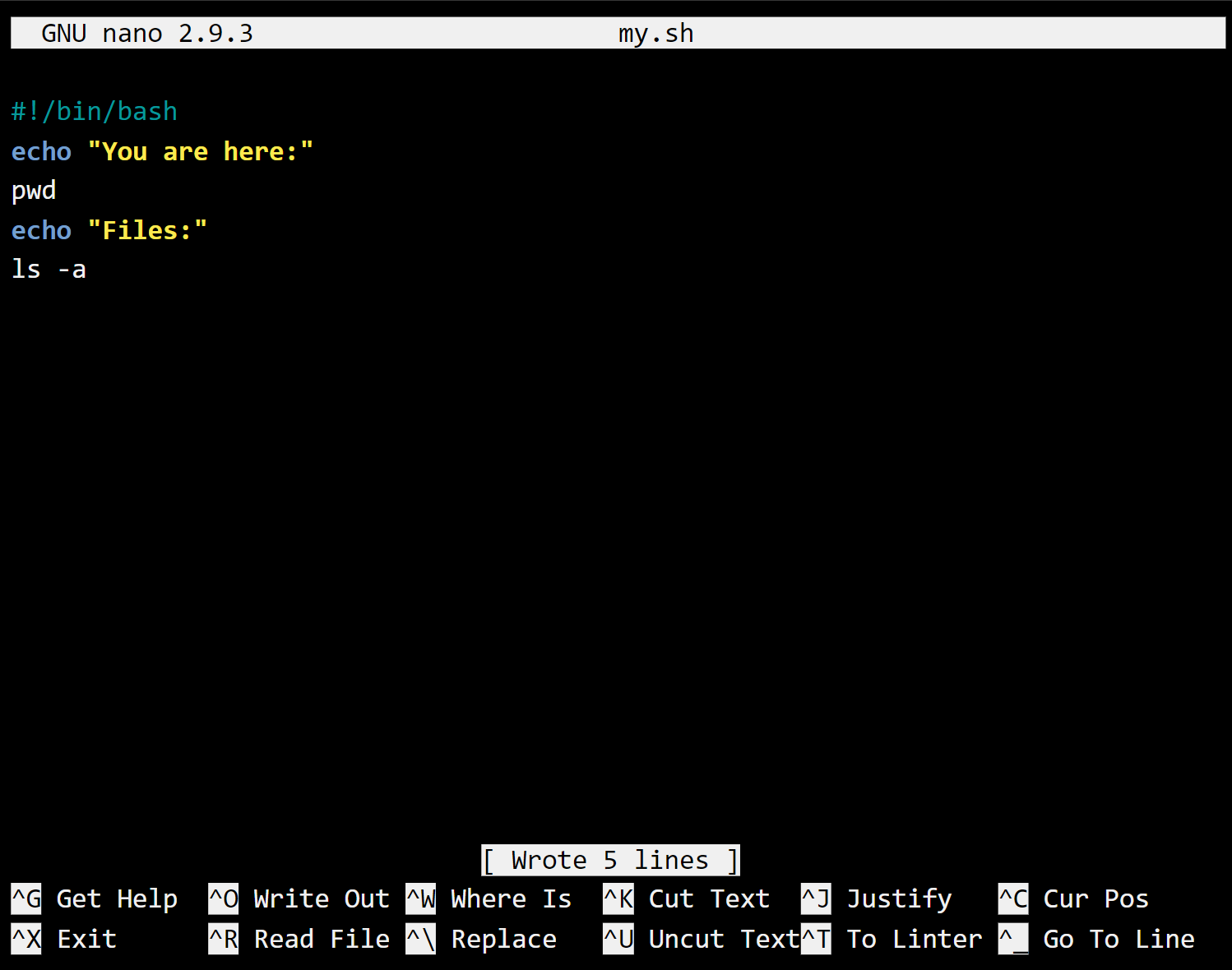


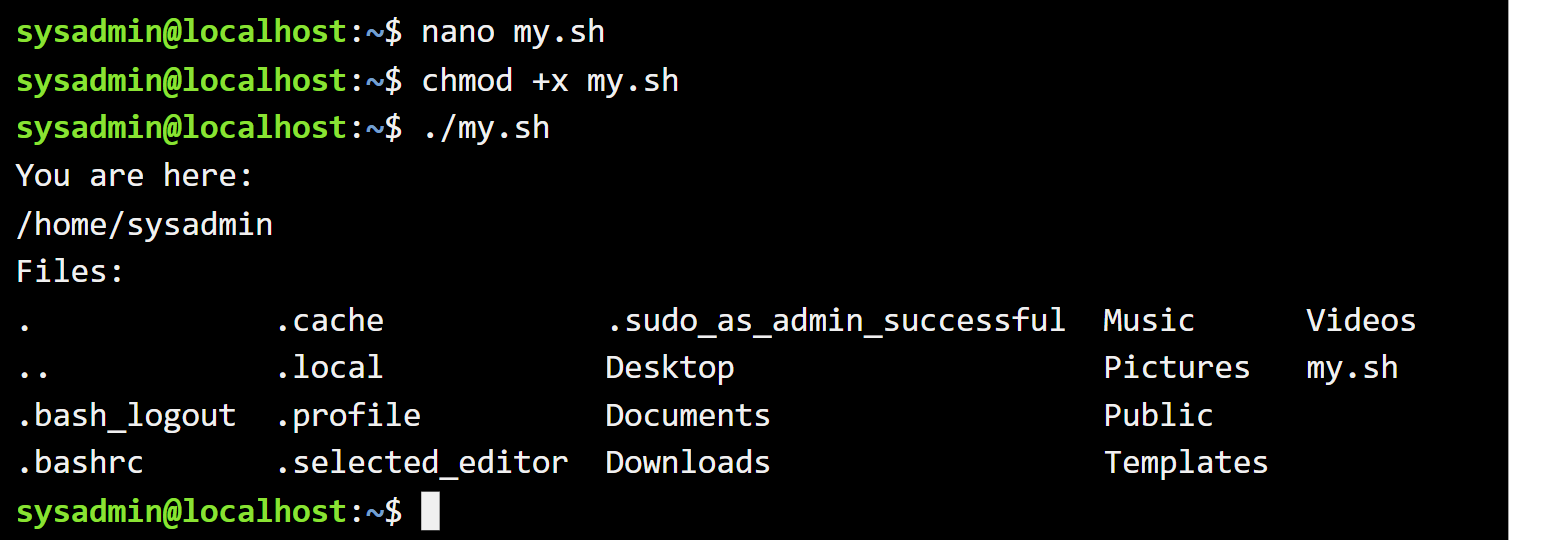






* \*\*наведіть свій приклад скриптового сценарію.





**Контрольні запитання:**

1. В чому відмінність між командами arch та lscpu?

The arch command simply displays information about the processor architecture on which it is running. The lscpu command provides more detailed information about the processor and the system.

1. Якою командою можна отримати інформацію про стан використання RAM поточною системою?

You can use the free command to get information about the current system's RAM usage in Linux.

1. \*Яким чином у скриптах можна опрацьовувати змінні та створювати розгалужені та циклічні сценарії?

To create variables, you use = signs and assign a value. To create loops, the for, while constructs are used. To create branches, use the if-then-else construct.

1. \*Які команди для перегляду стану підключення периферійних пристроїв можна використати в терміналі?

lsusb, lspci, lsblk, df –h, lsof, dmesg.

1. \*\*Які можливості застунку gparted?

Create, delete and resize, format, move, rename, partitions, manage partition labels, view partition properties. It also works with partitions using LVM and RAID, performs operations without rebooting, and supports Live CD/USB.

**Висновки**

В ході виконання лабораторної роботи мною було досліджено, команди Linux для створення скриптових сценаріїв та визначенння апаратної конфігурації системи, більш детально теоретично досліджено питання апаратного забезпечення. Отримано практичні навички роботи з командами створення та застосування скриптових сценаріїв і визначення апаратної конфігурації системи у терміналі.