**"КИЇВСЬКИЙ ФАХОВИЙ КОЛЕДЖ ЗВ’ЯЗКУ"**

**Лабораторна робота 5**

з дисципліни «Операційні системи»

**Тема:**  
“Команди Linux для архівування та стиснення даних. Робота з текстом”

Виконали: студенти **3** курсу, групи **КСМ-13А**

**Засенко Олександр**

(прізвище та ініціали)

**Дзюбенко Дмитро**

(прізвище та ініціали)

**Сторожук Костянтин**

(прізвище та ініціали)

Київ  2023

**ЗМІСТ**

Мета роботи………………………………………..……..……………………….3

Завдання для попередньої підготовки……………………………..….…………3

Хід роботи………………………………………...………………….……………10

Контрольні запитання……………….……………………………………………17

Conclusions……...…………………………………………………………………21

**Мета роботи:**

1. Отримання практичних навиків роботи з командною оболонкою Bash.

2. Знайомство з базовими командами для архівування та стиснення даних.

3. Знайомство з базовими діями при роботі з текстом у терміналі.

**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.

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

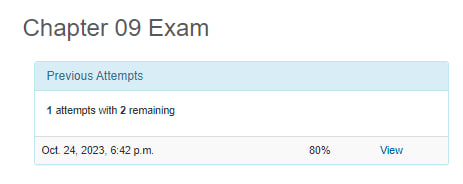
**Завдання для попередньої підготовки**

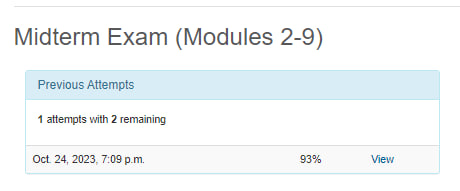
***Готував матеріал студент Zasenko***

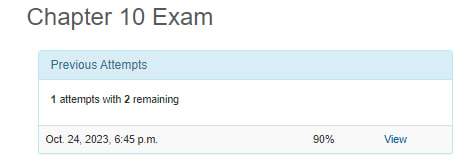
A small glossary of basic English terms related to the purpose of commands and their parameters.

|  |  |
| --- | --- |
| Command | Usage |
| bzip2 afile | This will compress the file and give it the name &quot;afile.bz2&quot;. |
| bzip2 -s afile | While bzip2 implements numbered flags, they mean something somewhat different than they do with gzip. |
| bzip2 -1 file | To decompress a bzip compressed file, you can pass the -d flag: |
| bzip2 -d file.bz2 | This will give back an uncompressed file called &quot;file&quot;. |
| xz file | This will process the file and produce a file called &quot;file.xz&quot;. |
| xz -c test &gt; test.xz | For the numbered flags, xz uses the lower numbers to indicate faster compression. |
| xz -e -9 large\_file | This will take a long time and in the end, may not show very significant gains, but if you need that  functionality, the option is available. |

**Take the NDG Linux Essentials course and get tested on the following topics:**







Based on the material reviewed, we learned:

1. tar:

Purpose: tar (Tape Archive) is used to create archives that can contain multiple files or directories.

Main parameters:

* -c: Creates an archive.
* -x: Unzips the archive.
* -v: Prints detailed output while running.
* -f file: Specifies the file name of the archive.

You can usually install tar using the installation packages of your Linux distribution (for example, apt-get install tar for Debian/Ubuntu).

1. xz:

Purpose: xz is a file compression utility that uses the LZMA2 compression algorithm.

Main parameters:

* -z: Compress the file (create a file with the .xz extension).
* -d: Decompress the file (unzip the .xz file).Use the package manager of your operating system to install xz.

1. zip:

Purpose: zip is a utility for creating and unpacking archives in ZIP format.

Main parameters:

* -r: Recursive archive creation for directories.
* -u: Update the archive by adding new files.
* -d: Delete files from the archive.

Use the package manager of your operating system to install the zip.

1. bzip2:

Purpose: bzip2 is a utility for compressing files using the BZip2 algorithm.

Main parameters:

* -c: Print the compressed content to standard output.
* -d: Decompress the BZip2 file.

Use your operating system's package manager to install bzip2.

1. gzip:

Purpose: gzip is a file compression utility using the GZip algorithm.

Main parameters:

* -d: Unzip the GZip file.
* -c: Print the compressed content to standard output.

Use your operating system's package manager to install gzip.

1. Using tar and gzip to create a compressed archive:

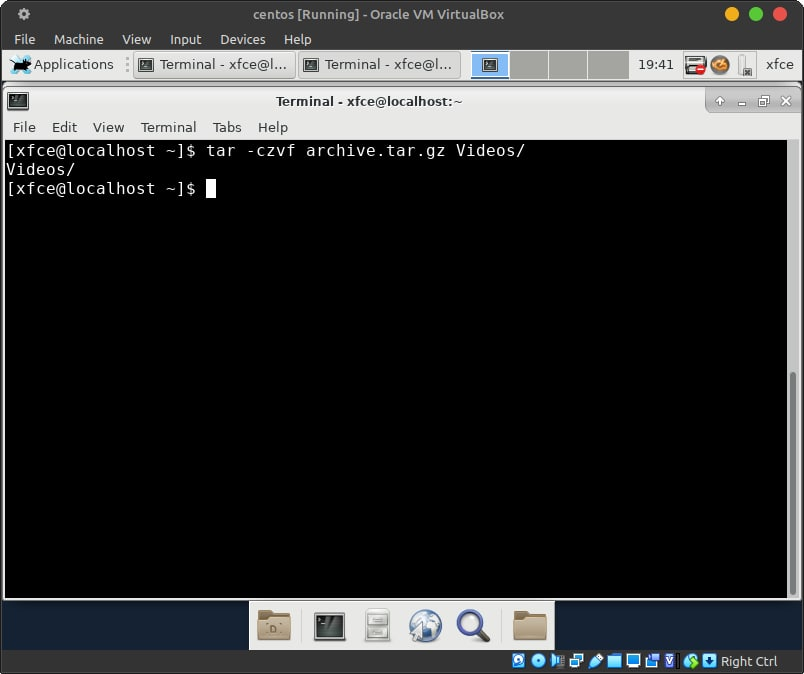


Fig 1. Using tar and gzip to create a compressed archive

In this example, tar creates an archive (archive.tar) from the contents of Videos/, and gzip compresses this archive to create the archive.tar.gz file.

1. Using zip to create a ZIP archive:

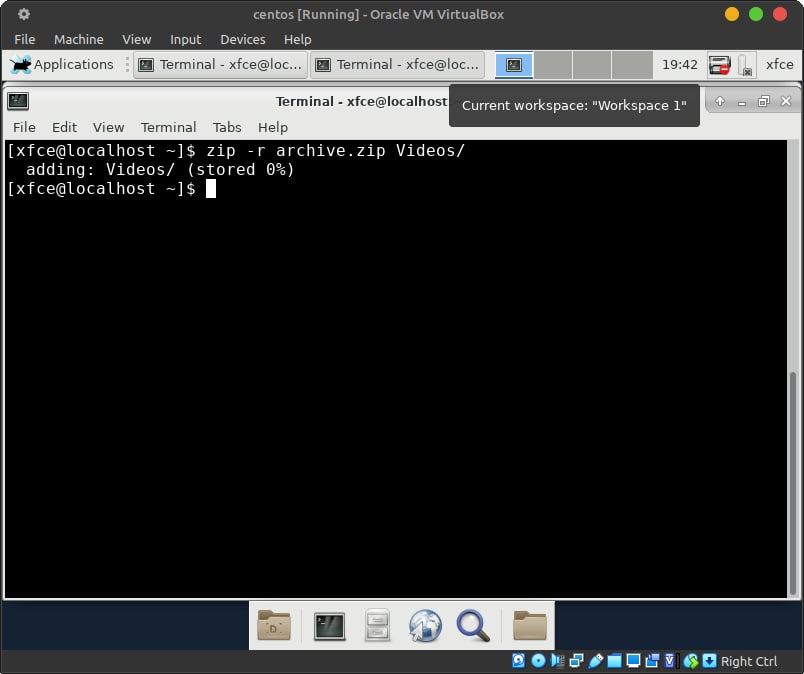


Fig 1. Using zip to create a ZIP archive

With the zip command, you can create a ZIP archive (archive.zip) containing the contents of Videos/. The -r switch causes files and directories to be recursively included in the ZIP archive.

1. Using bzip2 to compress a file:

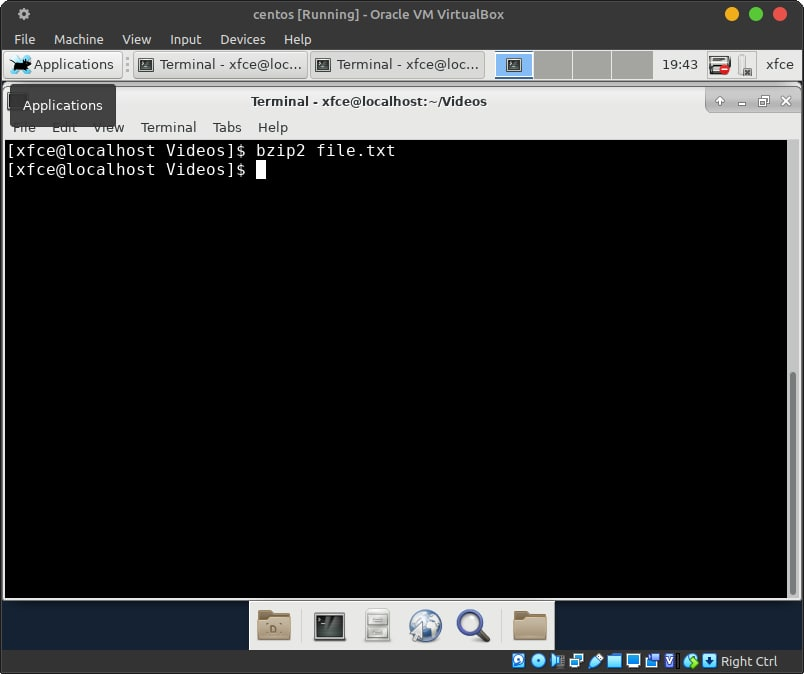


Fig 1. Using bzip2 to compress a file

The bzip2 command compresses the file file.txt, creating a compressed file file.txt

1. cat:

Purpose: cat (from "concatenate") is used to display the contents of files and concatenate them together.

Main parameters:

* Print the contents of one or more files: cat file1.txt file2.txt
* Concatenate files: cat file1.txt file2.txt > combined.txt

cat is usually installed on most Linux distributions by default.

1. less:

Purpose: less is a text file viewer that allows you to scroll up and down and search.

Main parameters:

* Open file for viewing: less file.txt
* Scroll the text down (the "Space" key) and up (the "B" key).
* Exit the program (key "q").

less is usually installed on most Linux distributions by default.

1. more:

Purpose: More is another text file viewer that also allows you to scroll up and down.

Main parameters:

* Open file for viewing: more file.txt
* Scroll down (Space bar) and up (Enter key).
* Exit the program (key "q").

more is also installed on most Linux distributions by default.

1. head:

Purpose: head is used to display the first few lines of a text file.

Main parameters:

* Print the first N lines of the file: head -n N file.txt
* Print the first 10 lines by default: head file.txt

head is also installed on most Linux distributions by default.

1. tail:

Purpose: tail is used to print the last few lines of a text file.

Main parameters:

* Print the last N lines of a file: tail -n N file.txt
* Print the last 10 lines by default: tail file.txt

tail is also installed on most Linux distributions by default.

A command shell (e.g., Bash on Linux) works with channels, streams, and filters to process and exchange data. The basic principles are as follows:

1. Streams:

* STDIN (standard input): This is the input stream through which the command can accept data. Typically, the Enter key or a redirect from a file is used.
* STDOUT: This is the output stream through which the command outputs the results of its work. This is usually output to a terminal or can be redirected to a file.
* STDERR (standard error): This is also an output stream, but for outputting errors and error messages. This is usually output to a terminal, but can be redirected for logging.

1. Pipes:

* Channels allow you to combine the work of several commands together by passing the output stream of one command as the input stream of another command.
* The syntax for a channel is a vertical line |.

1. Filters.:

* Filters are commands that process and modify data transmitted through streams. They can perform filtering, replacement, sorting, trimming, and more.
* Filters are often used in conjunction with channels for complex data processing operations.

The grep command is designed to find and display text strings that match a specific pattern or regular expression in the source text. Common uses of grep include:

1. **Searching for text in a file or data stream**: grep allows you to search for a specific text string or pattern in the contents of a text file or data stream, such as when you are looking for a particular word or phrase in a log file or document.
2. **Use regular expressions**: grep supports the use of regular expressions for more complex searches, such as finding all strings that match a specific pattern, such as all numbers or email addresses in a text file.
3. **Print matching strings**: grep prints all strings that match a pattern or regular expression to standard output. You can also redirect the result to a file or use it in scripts and other commands.
4. **Search multiple files**: grep can be applied to multiple files or include options to recursively search directories.

**Хід роботи**

***Готував матеріал студент Dziubenko***

1. Initial work in CLI mode in the Linux family of operating systems:

1.1. Start the VirtualBox virtual machine, select CentOS, and start it. Log in to the system under the user: CentOS, login password: reverse (if you are performing the LP in 401 classroom) and launch the

terminal.

1.2. Start the Ubuntu\_PC virtual machine (if you are performing the PL task through the netacad academy)

1.3. Start your operating system of the Linux family (if you are working on your own PC and have installed it) and start the terminal.

installed) and launch the terminal.

1. Work through all the sample commands presented in the labs of the NDG Linux Essentials course -

Lab 9: Archiving and Compression and Lab 10: Working With Text. Create a table to describe these commands\*\*\*.

|  |  |
| --- | --- |
| Command | What does the command do? |
| mkdir mybackups | Creating a new directory "mybackups" in the user's home directory |
| tar -cvf mybackups/udev.tar | The command "tar" is used to combine multiple files into one file. In this case, |
| /etc/udev | the contents of the "/etc/udev" directory will be saved in an archive called "udev.tar" in the "mybackups" directory. The "-c" option instructs the "tar" command to create a tar file. The "-v" option stands for "verbose," which tells the "tar" command to show what it is doing. The "-f" option is used to specify the name of the tar file. |
| ls mybackups | List the contents of the "mybackups" directory. |
| cp /usr/share/dict/words | Copy the file "words" from "/usr/share/dict" to the current directory. |
| gzip words | Compress the file "words" using gzip, creating "words.gz." |
| gunzip words.gz | Decompress the file "words.gz" to restore "words." |
| bzip2 words | Compress the file "words" using bzip2, creating "words.bz2." |
| bunzip2 words.bz2 | Decompress the file "words.bz2" to restore "words." |
| xz words | Compress the file "words" using xz, creating "words.xz." |
| unxz words.xz | Decompress the file "words.xz" to restore "words." |
| zip words.zip words | Create a ZIP archive named "words.zip" containing the "words" file. |
| unzip -l udev.zip | List the contents of the ZIP archive "udev.zip" without extracting. |
| rm -r etc | Recursively remove the "etc" directory and its contents. |
| echo “Hello World” | Display the text "Hello World." |
| cat mymessage | Display the content of the file "mymessage." |
| find ~ -name “\*bash\*” | Search for files with "bash" in their names within the home directory (~). |
| tr a-z A-Z | Translate lowercase letters to uppercase. |
| cut -d: -f1 /etc/passwd | sort | Extract and sort usernames from the "/etc/passwd" file using ":" as the delimiter. |
| more /etc/passwd | Display the contents of the "/etc/passwd" file one page at a time. |
| Less /etc/passwd | Display the contents of the "/etc/passwd" file in a pager for reading. |
| head /etc/passwd | Display the first few lines of the "/etc/passwd" file. |
| tail /etc/passwd | Display the last few lines of the "/etc/passwd" file. |
| grep sshd passwd | Search for the string "sshd" in the "passwd" file. |
| egrep | This is a command that would typically be followed by a regular expression for pattern matching. It's not specified here. |

1. Familiarize yourself with the tar command and use it to perform the following actions in the terminal:

* create a file with the .tar extension;

**tar -cvf file.tar file1 file2**

* create a file with the .tar extension, consisting of several files and directories at the same time;

**tar -cvf file.ter file1 directory1**

* view the contents of the file;

**tar -tvf file.tar**

* extract the contents of the tar file;

**tar -xvf file.tar**

* create a bzip compressed tar archive file;

**tar -cvjf file.tar.bz2 file1 file2**

* extract contents of tar bzip file;

**tar -xvjf file.tar.bz2**

* create an archive tar file compressed with gzip;

**tar -cvzf file.tar.gz file1 file2**

* extract tar gzip file contents.

**tar -xvzf file.tar.gz**

1. How will the output streams be redirected to bash for the following actions with commands (denoted as cmd) and files (denoted as file):

|  |  |
| --- | --- |
| Command | What does the command do? |
| cmd 1&gt; file | Outputting stdout of cmd command to file. |
| cmd &gt; file | The same as the previous action, outputting stdout to file |
| cmd 2&gt; file | Stderr output of the cmd command to the file. |
| cmd &gt;&gt; file | Adding the stdout of the cmd command to the contents of the file (instead of overwriting). |
| cmd &amp;&gt; file | Output both stdout and stderr of the cmd command to the file file. |
| cmd &gt; file 2&gt;&amp;1 | and outputting the stdout of the cmd command to the file file, as well as stderr (2) to the same file as stdout (1). |
| cmd &gt;&gt; file 2&gt;&amp;1 | Adding both stdout and stderr of the cmd command to the contents of the file. |
| cmd 2&gt;&amp;1 &gt; /dev/null | Outputting the stdout of the cmd command to /dev/null (complete discard), while stderr remains. |
| cmd 2&gt; /dev/null | Outputting the stderr of the cmd command to /dev/null (completely discarding stderr). |
| cmd1 | cmd2 | Redirecting the stdout of the cmd1 command to the input of the cmd2 command (pipeline). |
| cmd1 2&gt;&amp;1 | cmd2 | Redirecting the stdout and stderr of cmd1 to the input of cmd2 (pipeline), that is, both output streams of cmd1 are used by cmd2 |

1. Consider the following examples and explain what these commands do and what type of flow redirection they use:

|  |  |  |
| --- | --- | --- |
| Command (command container) | What the command do? | What is the redirect flow? |
| $echo &quot;It is a new story.&quot; &gt; story | The echo command prints the string "It is a new story." to the standard output (stdout).   * The output of the echo command is redirected to the file story. | It uses the > operator, which redirects stdout to a file. |
| $ date &gt; date.txt | The date command displays the current date and time to stdout.   * The output of the date command is redirected to the file date.txt. | It uses the > operator, which redirects stdout to a file. |
| $ cat file1 file2 file3 &gt; bigfile | The cat command concatenates the contents of files file1, file2, and file3 and outputs them to stdout.   * The output of the cat command is redirected to the file bigfile. | It uses the > operator, which redirects stdout to a file. |
| $ls -l &gt;&gt; directory | The ls -l command lists files and directories in "long listing" format and outputs to stdout.   * The output of the ls -l command is appended to the end of the file directory. | It uses the >> operator, which appends stdout to a file. |
| $ sort &lt; file1\_unsorted &gt; file2\_sorted | The sort command sorts the content of the file file1\_unsorted.   * The content of file1\_unsorted is redirected to the stdin of the sort command. * The sorted result is saved to the file file2\_sorted. | It uses the < operator, which redirects stdin from a file. |
| $ find -name &#39;\*.txt&#39; &gt; file.txt 2&gt; /dev/null | The find command searches for files with the extension .txt in the current directory and its subdirectories.   * The search result (stdout) is redirected to the file file.txt. * The stderr is redirected to /dev/null, effectively discarding error messages. | It uses the > and 2> operators, which redirect stdout and stderr, respectively. |
| $ cat file1\_unsorted | sort &gt; file2\_sorted | The cat command outputs the content of file1\_unsorted to stdout.   * The output of cat is piped (|) to the stdin of the sort command, which sorts the data. | The sorted result is saved to the file file2\_sorted. |
| $ cat myfile | grep student | wc -l | This is a sequence of commands: cat, grep, and wc -l.   * cat outputs the content of myfile to stdout. * The output of cat is piped to the stdin of the grep command, which searches for lines containing the word "student." | The result of grep is then piped to the stdin of the wc -l command, which counts the number of lines. |

**Контрольні запитання**

***Готував матеріал студент Storozhuk***

1.

|  |  |  |
| --- | --- | --- |
| Characteristics | Data compression | Data archiving |
| Objective | Reduce file size | Reduce file size and combine them into a single archive |
| Result | Compressed files | Archives |
| Types of compression | Lossless, lossless | Lossless, lossless |
| Advantages | Reduces the size of files while maintaining their original format | Reduces file size, facilitates file management, and provides backup |
| Disadvantages | May be slower than archiving | May be less effective for smaller files |

2. In addition to the programs described in this article, you can also use the following programs to compress and archive files and directories in Linux:

* 7-Zip is a free, open-source data compression and archiving program that supports many different compression algorithms, including LZMA, LZMA2, PPMd, BZIP2, DEFLATE, ZIP, and TAR.
* PeaZip is another free, open-source data compression and archiving program that supports a wide range of formats, including ZIP, 7Z, RAR, TAR, CAB, ISO, and IMG.
* WinRAR is a commercial data compression and archiving program that supports ZIP, RAR, CAB, ARJ, ISO and VHD formats.
* WinZip is another commercial data compression and archiving program that supports ZIP, ZIPX, RAR, CAB, ARJ, ISO and VHD formats.

Brief description of the programs:

* 7-Zip is a powerful program with a wide range of features, including support for many different compression algorithms, creating multi-volume archives and creating self-extracting archives.
* PeaZip is an easy-to-use program with a user-friendly interface. It supports a wide range of formats and can be used both to compress files and to archive them.
* WinRAR and WinZip are popular programs that support all major archive formats. They offer a wide range of additional features, such as backup, password protection, and processing of archives in a package.

3. The following compression algorithms are most commonly used in Linux:

* gzip is a DEFLATE compression algorithm that combines the LZ77 and Huffman coding algorithms. It is one of the most common compression algorithms on Linux because it provides a good balance between speed and efficiency.
* bzip2 is a compression algorithm that uses block sorting and Huffman encoding. It provides a higher compression ratio than gzip, but is slower.
* xz is a compression algorithm that uses an improved version of the bzip2 algorithm. It provides an even higher compression ratio than bzip2, but is even slower.
* zstd is a new compression algorithm that provides a very high compression ratio. It is the fastest compression algorithm of all, but it is still under development and is not supported by all programs.

The choice of compression algorithm depends on the specific needs of the user:

* For small files (up to 1 MB), gzip is the best choice.
* For medium-sized files (1 MB to 100 MB), you can use bzip2 or xz.
* For large files (over 100 MB), it is best to use xz or zstd.

4. Mobile devices such as smartphones have a variety of file compression and archiving applications that can be used to optimize space and data processing. Here are some popular compression and archiving software that can be found on many mobile platforms:

* **RAR for Android:**

RAR for Android allows users to compress files in RAR format and archive them directly from their mobile device. It also supports extracting files from RAR archives and other popular formats.

* **ZArchiver:**

ZArchiver is a powerful tool for working with archives on mobile devices. It allows you to create archives, unpack files from archives, and compress files to optimize space.

* **RARLAB's RAR:**

RARLAB's RAR is another popular compression and archiving program available on mobile devices. It supports the standard RAR format and also allows you to compress files to reduce the size.

* **AndroZip:**

AndroZip is a universal tool for working with archives on Android devices. It allows you to compress files, archive folders, unzip archives, and manage files.

5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Characteristics | Built-in features of the operating system | WinRAR | WinZip | 7-Zip |
| Availability | Built into the operating system | Paid | Paid | Free, open source |
| Archive formats | ZIP, CAB, RAR, ARJ, ISO | ZIP, RAR, ZIPX, 7Z, ACE, GZ, BZ2, TAR, ISO | ZIP, ZIPX, RAR, 7Z, CAB, ARJ, ISO, WIM | ZIP, 7Z, GZIP, BZIP2, TAR, CAB, RAR, ISO, WIM, XZ |
| Additional features | None | Encrypt, backup, create self-extracting archives | Encrypt, backup, create self-extracting archives | Encryption, backup, create self-extracting archives, create multi-volume archives, command line support |

6. Data compression and archiving can be used to back up data in such cases:

* Reducing the size of an archive. When data is compressed, its size is reduced. This can make backups more compact and easier to store.
* Easier transportation of the archive. When data is archived, it is combined into a single file. This can make backups easier to transport to external media, such as USB drives or optical disks.
* Improve archive security. Data in the archive can be encrypted to protect it from unauthorized access.

7. The /dev/null file directory, also known as the "null device," is a special file in Unix-like operating systems. It does not store data, but simply rejects all data that comes to it.

/dev/null is often used for such purposes:

* Hiding command output. If you want to hide the output of a command, you can redirect it to /dev/null. For example, the command echo "Hello, world!" > /dev/null will not print any text.
* Refuse to print errors. If you want to stop a program from printing errors, you can redirect it to /dev/null. For example, the command ./myprogram 2> /dev/null will not print any errors.
* Destroying data. If you want to destroy data, you can redirect it to /dev/null. For example, the command cat file.txt > /dev/null will destroy the file.txt file.

**Conclusions**

***Готував матеріал студент Storozhuk***

In this paper, we have learned about data compression and archiving in Linux. We also consolidated this knowledge in practice. We did not have any problems during this lab.