“Київський фаховий коледж зв’язку”

Циклова комісія Комп’ютерної інженерії

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №6**

з дисципліни: «Операційні системи»

**Тема: «** **Команди Linux для архівування та стиснення даних. Робота з текстом»**

Виконала студентка

групи РПЗ-13б

Дімітрова С.П.

Перевірив викладач

Сушанова В.С.

Київ 2024

**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 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 |
| **Compression** | Стиснення |
| **Decompression** | Розпакування |
| **Compression ratio** | Ступінь стиснення |
| **Lossy compression** | Стиснення з втратою |
| **Lossless compression** | Стиснення без втрат |
| **Archiving** | Архівація |
| **Extension** | Розширення |
| **Verbose** | Детальний |
| **Compatibility** | Сумісність |

1. На базі розглянутого матеріалу дайте відповіді на наступні питання:
   1. \*Яке призначення команд  *tar*, *xz*, *zip*, *bzip*, *gzip*? Зробіть короткий опис кожної команди та виділіть їх основні параметри. Яким чином їх можна встановити.

* *tar:* is used to create archives (collections of files) that preserve directory structures and file permissions.

*Main parameters:*

* c: create an archive;
* v: verbose output (show details during processing);
* f: specify the archive filename (filename.tar);
* x: extract an archive;
* t: list contents of an archive;
* z: compress the archive with gzip after creation (alternative: -cvzf for creating a verbose gzip archive in one step);
* j: compress the archive with bzip2 after creation (alternative: -cjvf for creating a verbose bzip2 archive in one step);
* J: compress the archive with xz after creation (alternative: -cJvf for creating a verbose xz archive in one step).

*Installation:* tar is typically pre-installed on most Linux distributions. You can verify with *which tar* and it should return the path to the executable. If not available, use your distribution's package manager (e.g., apt, yum) to install it. Command: sudo apt install tar

* *gzip:* compresses files using the DEFLATE algorithm. Offers a balance between speed and compression ratio.

*Main parameters:*

* c: compress data to standard output (use with > to redirect to a file);
* d: decompress a file;
* l: list information about a compressed file (size, compression ratio);
* v: produce verbose output;
* 1 to -9 (or --fast to --best): adjust compression level (higher numbers for better compression but slower processing).

*Installation:* similar to tar, gzip is usually pre-installed. Verify with *which gzip*. If missing, use your package manager to install it.

* *bzip:* Compresses files using the Burrows-Wheeler algorithm. Achieves higher compression than gzip but takes longer.

*Main parameters:*

* Similar flags to gzip with some variations:
  + d: decompress;
  + l: list information;
  + v: Verbose output;
  + 1 to -9: adjust compression level (lower numbers for faster processing);
* s: reduce memory usage during compression (lowers compression ratio).

*Installation:* might not be pre-installed by default. Use your package manager to install bzip2 if needed.

* *xz:* compresses files using the LZMA2 algorithm. Provides the highest compression ratio but has the slowest processing time.

Main parameters:

* + Similar flags to gzip and bzip2:
  + d: decompress;
  + l: list information;
  + v: verbose output;
  + 1 to -9: adjust compression level (lower numbers for faster processing);
* e: use an alternate "extreme" compression mode (very slow).

*Installation:* Might not be pre-installed by default. Use your package manager to install xz if needed.

* *zip:* primarily used for archiving on Windows systems. Can also be used in Linux but might not be pre-installed.

*Functionality:* similar to tar for creating archives, can compress with various algorithms depending on the specific tool. If not pre-installed, use your package manager to install a tool like zip or unzip.

* 1. \*\*Наведіть три приклади реалізації архівування та стискання даних різними командами.
* *tar -czvf archive.tar.gz directory1 :*

**Archiving a directory with gzip compression. This command uses tar to create an archive named "archive.tar" from the contents of "directory1". The -z flag instructs tar to compress the archive using gzip after creation. This is a common approach for creating a compressed archive that is compatible with most systems.**

* *zip -r archive.zip directory1:*

This command utilizes zip to create an archive named "archive.zip" that contains the entire contents of the directory "directory1".

r: Instructs zip to include the entire directory structure recursively, meaning it will archive all subdirectories within "directory1" as well.

* *gzip longfile.txt :*

The file is compressed by calling the gzip command with the file name as an argument. After executing this command, the original file disappears, and the compressed version with the .gz file extension remains in its place.

* 1. \*Яке призначення команд  cat, less, more, head and tail? Зробіть короткий опис кожної команди та виділіть їх основні параметри. Яким чином їх можна встановити?
* *cat:* can be used to display file contents and will be used in this example to verify redirected output to the file. The main parameter of cat is -n, which numbers all output lines. The cat command is good for viewing small files, but not ideal for large files;
* *less:* the less command is a paginator that allows scrolling through large text files. It displays the content of a file one screen at a time and provides navigation options;
* *more:* similar to less, more is also a paginator but with fewer features. It displays the content of a file one screen at a time and pauses after each screenful. The main parameter for more is -n, which displays line numbers;
* *head:* the head command is used to display the first few lines of a file. By default, it shows the first 10 lines, but this can be adjusted using the -n parameter followed by the desired number of lines;
* *tail:* conversely, the tail command displays the last few lines of a file. By default, it shows the last 10 lines, but this can be adjusted using the -n parameter followed by the desired number of lines.
  1. \*\*Поясніть принципи роботи командної оболонки з каналами, потоками та фільтрами.
* In Unix-like systems such as Linux, there are two main types of streams: the standard output stream (stdout) and the standard error stream (stderr). Commands output results to stdout and error messages to stderr. There is also a standard input stream (stdin) that reads input data. The shell allows you to redirect stdout and stderr to a file using the > and >> symbols. The > character overwrites the file and the >> character appends to an existing file. You can also redirect stderr with 2>. To redirect both stdout and stderr, use &>.
* Channels allow you to redirect the stdout of one command to the stdin of another command. The pipe, which is indicated by the vertical line symbol "|", can be used to send the result of one command to another. For example, the ls | sort command displays a list of files, which is then sorted by the sort command. Multiple channels can be used in series to link multiple commands together. Each command sees only the data of the previous team.
* Filters are commands that read data from stdin, process this data, and then output the results to stdout. For example, the grep command is a filter that searches for strings that match a given pattern.
  1. \*Яке призначення команди grep?

The grep command is used to search for text in files. It searches for strings that match a given pattern and displays them on the screen. The grep command can be used with several string filtering options:

* The -d option can specify alternative delimiters such as colon or comma
* The -f option can specify which fields to display.
* The -c option is used to select columns of text based on character position.

**Хід роботи:**

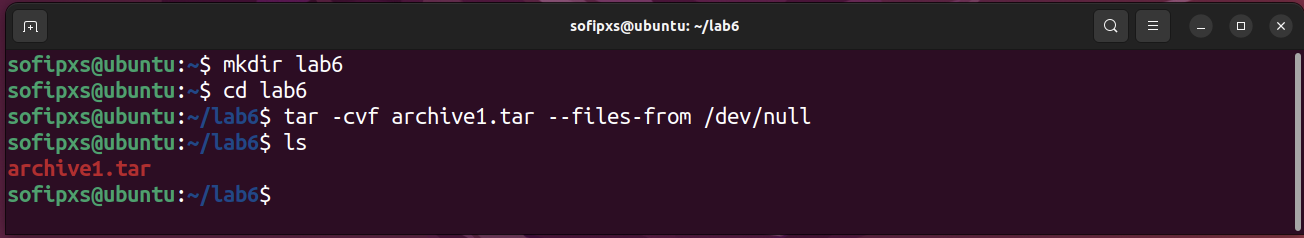
1. Початкова робота в CLI-режимі в Linux ОС сімейства Linux:
   1. Запустіть віртуальну машину VirtualBox, оберіть CentOS та запустіть її. Виконайте вхід в систему під користувачем: CentOS, пароль для входу: reverse ***(якщо виконуєте ЛР у 401 ауд.)*** та запустіть термінал.
   2. Запустіть віртуальну машину Ubuntu\_PC ***(якщо виконуєте завдання ЛР через академію netacad)***
   3. Запустіть свою операційну систему сімейства Linux ***(якщо працюєте на власному ПК та її встановили)*** та запустіть термінал.
2. Опрацюйте всі приклади команд, що представлені у лабораторних роботах курсу ***NDG Linux Essentials - Lab 9: Archiving and Compression*** та ***Lab 10: Working With Text.*** Створіть таблицю для опису цих команд

|  |  |
| --- | --- |
| Назва команди | Її призначення та функціональність |
| mkdir mybackups | Create a new mybackups directory in the user's home directory |
| tar -cvf mybackups/udev.tar /etc/udev | The tar command is used to combine several files into a single file. In this case, the contents of the /etc/udev directory will be saved to the udev.tar archive in the mybackups directory. The -c option tells 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. |
| tar –tvf mybackups/udev.tar | Display the contents of a tar file by using the available options (t = list contents, v = verbose, f = filename): |
| tar –zcvf mybackups/udev.tar.gz /etc/udev | This command creates a compressed tar file using the -z option. The -z option makes use of the gzip utility to perform compression. |
| ls –lh mybackups | This command lists the contents of the directory mybackups in a human-readable format, showing details such as file sizes in a format easy to understand. |
| tar –xvf udev.tar.gz | This command extracts the contents of the archive. By default, the data will be restored to the current directory. |
| tar -rvf udev.tar /etc/hosts | This command adds the /etc/hosts file to the existing udev.tar archive. The -r option is used to add files to the archive, -v provides extended output to display the process, and -f udev.tar specifies the name of the archive file. |
| tar –tvf udev.tar | This command displays the contents of the udev.tar archive without unpacking it. Use the -t option to view the contents of the archive. |
| gzip words | Compresses the file words using gzip compression, creating a new file named words.gz while removing the original uncompressed file words. |
| gunzip words.gz | Execute the following commands to uncompress the words.gz file |
| bzip2 words | The compressed file is created with a .bz2 extension. The extension is removed when uncompressed. |
| bunzip2 words.bz2 | Execute the following commands to uncompress the words.bz2 file |
| xz words | The compressed file is created with a .xz extension. The extension is removed when uncompressed. |
| unxz words.xz | Execute the following commands to uncompress the words.xz file. |
| zip words.zip words | Use the zip command to compress the words file.The first argument (words.zip) of the zip command is the file name that we wish to create. The remaining arguments (words) are the files we want placed in the compressed file |
| zip -r udev.zip /etc/udev | Compress the /etc/udev directory and its contents with zip compression. The recursive -r option must be specified in order to perform recursion into subdirectories. |
| unzip -l udev.zip | To view the contents of a zip archive, use with the -l option with the unzip command. |
| rm -r etc  unzip udev.zip | To extract the zip archive, use the unzip command without any options. In this example we first need to delete the files that were created in the earlier tar example: |
| ***Lab 10: Working With Text.*** | |
| echo "Hello World"  echo "Hello World" > mymessage  cat mymessage | Use the redirection symbol > along with the echo command to redirect the output from the normal output of stdout (to the terminal) to a file. The first command echoes the message (stdout) to the terminal. The second command redirects the output; instead of sending the output to the terminal, it is sent to a file called mymessage. The last command displays the contents of the mymessage file. |
| echo "Greetings" > mymessage  cat mymessage | When you use the > symbol to redirect stdout, the contents of the file are first destroyed |
| cat mymessage  echo "How are you?" >> mymessage  cat mymessage | You can avoid clobbering a file by using >> instead of >. By using >>you append to a file. |
| find ~ -name "\*bash\*" | The find command will begin the search in the directory specified and recursively search all of the subdirectories. To search for files beginning in your home directory containing the name bash. Remember that ~ is used to represent your home directory. |
| find /etc -name hosts 2> err.txt  cat err.txt | This command redirects stderr (error messages) to a file. Descriptor for stderr is the number 2, so it is used along with the > symbol to redirect the stderr output to a file called err.txt. Note that 1> is the same as >. |
| find /etc -name hosts > std.out 2> std.err  cat std.err  cat std.out | You can also redirect stdout and stderr into two separate files. A space is permitted but not required after the > redirection symbol. |
| find /etc -name hosts > find.out 2>&1  cat find.out | To redirect both standard output (stdout) and standard error (stderr) to one file, first redirect stdout to a file and then redirect stderr to that same file by using the notation 2>&1. The 2>&1 part of the command means "send the stderr (channel 2) to the same place where stdout (channel 1) is going". |
| tr a-z A-Z  this is interesting  how do I stop this?  ^D | Standard input (stdin) can also be redirected. Normally stdin comes from the keyboard, but sometimes you want it to come from a file instead. For example, the tr command translates characters, but it only accepts data from stdin, never from a file name given as an argument. This is great when you want to do something like capitalize data that is inputted from the keyboard (Note: Press Control+d, to signal the tr command to stop processing standard input). |
| tr A-Z a-z > myfile  Wow, I SEE NOW  This WORKS! | The tr command accepts keyboard input (stdin), translates the characters and then redirects the output to stdout. To create a file of all lower-case characters, execute this command. |
| cat myfile | To verify you created the file, execute this command. |
| cat myfile  tr a-z A-Z < myfile | Execute the following commands to use the tr command by redirecting stdin from a file. |
| ls -l /etc | more | Execute this command to take the output of the ls command and send it into the more command, which displays one page of data at a time. |
| cut -d: -f1 /etc/passwd | In the following example, you will use a command called cut to extract all of the usernames from a database called /etc/passwd (a file that contains user account information). First, try running the cut command by itself. |
| cut -d: -f1 /etc/passwd | sort | The output in the previous example was unordered and scrolled off the screen. In the next step you are going to take the output of the cut command and send it into the sort command to provide some order to the output. |
| cut -d: -f1 /etc/passwd | sort | more | Now the output is sorted, but it still scrolls off the screen. Send the output of the sort command to the more command to solve this problem: |
| cat /etc/passwd | The /etc/passwd is likely too large to be displayed on the screen without scrolling the screen. To see a demonstration of this, use the catcommand to display the entire contents of the /etc/passwdfile. |
| more /etc/passwd | Use the more command to display the entire contents of the /etc/passwd file. |
| h | While you are in the more command, you can view the help screen by pressing the *h* key. |
| <SPACE> | Press the Spacebar to view the rest of the document. |
| less /etc/passwd  /bin  nnnNNNq | Use the less command to display the entire contents of the /etc/passwd file. Then search for the word bin, use *n* to move forward, and *N* to move backwards. Finally, quit the less pager by typing the letter *q.* |
| head /etc/passwd | You can use the head command to display the top part of a file. By default, the head command will display the first ten lines of the file. |
| tail /etc/passwd | Use the tail command to display the last ten lines of the /etc/passwd file. |
| head -2 /etc/passwd | Use the head command to display the first two lines of the /etc/passwd file. |
| ls /etc | tail -5 | Execute the following command line to pipe the output of the lscommand to the tail command, displaying the last five file names in the /etc directory. |
| head -n -20 /etc/passwd | Another way to specify how many lines to output with the head command is to use the option *-n -#,* where # is the number of lines counted from the bottom of the output to exclude. Notice the minus symbol - in front of the #. For example, if the /etc/passwd contains 27 lines, the following command will display lines 1-7, excluding the last twenty lines. |
| cd /etc  grep sshd passwd | The use of grep in its simplest form is to search for a given string of characters, such as sshd in the /etc/passwd file. The grep command will print the entire line containing the match. |
| grep root passwd | Regular expressions are "greedy" in the sense that they will match every single instance of the specified pattern. Note the red highlights indicate what exactly was matched. |
| grep '^root' passwd | To limit the output, you can use regular expressions to specify a more precise pattern. For example, the caret ^ character can be used to match a pattern at the beginning of a line; so, when you execute the following command line, only lines that begin with root should be matched and displayed. |
| grep 'sync' passwd | Match the pattern sync anywhere on a line. |
| grep 'sync$' passwd | Use the $ symbol to match the pattern sync at the end of a line. |
| grep '.y' passwd | Use the period character . to match any single character. For example, execute the following command to match any character followed by a 'y'. |
| grep 'sshd|root|operator' passwd | The pipe character, |, or "alternation operator", acts as an "or" operator. For example, execute the following to attempt to match either sshd, root or operator. |
| grep -E 'sshd|root|operator' passwd | Use the -E switch to allow grep to operate in extended mode in order to recognize the alternation operator. |
| egrep 'no(b|n)' passwd | Use another extended regular expression, this time with egrep with alternation in a group to match a pattern. The strings nob and non will match. The parentheses,  ( ), were used to limit the "scope" of the | character. Without them, a pattern such as nob|n would have meant "match either nob or n”. |
| head passwd | grep '[0-9]' | The [ ] characters can also be used to match a single character. If you want to match a numeric character, you can specify [0-9]. The head command was used to limit the output of the grep command. |
| grep -E '[0-9]{3}' passwd | Suppose you want to search for a pattern containing a sequence of three digits. You can use { } characters with a number to express that you want to repeat a pattern a specific number of times; for example: {3}. The use of the numeric qualifier requires the extended mode of grep. |

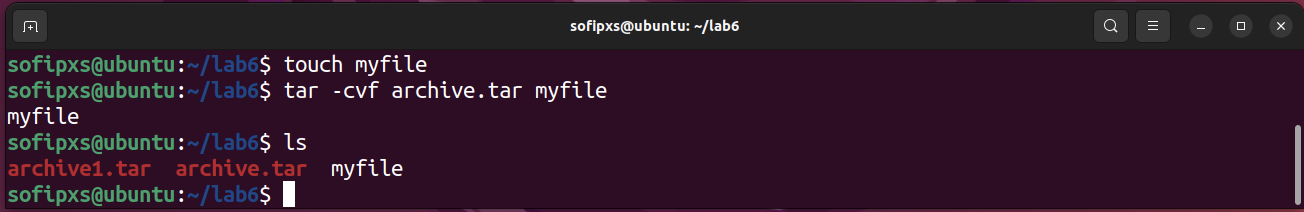
1. Ознайомтесь з командою tar та за її допомогою виконати у терміналі наступні дії:

* створити файл з розширенням .tar;

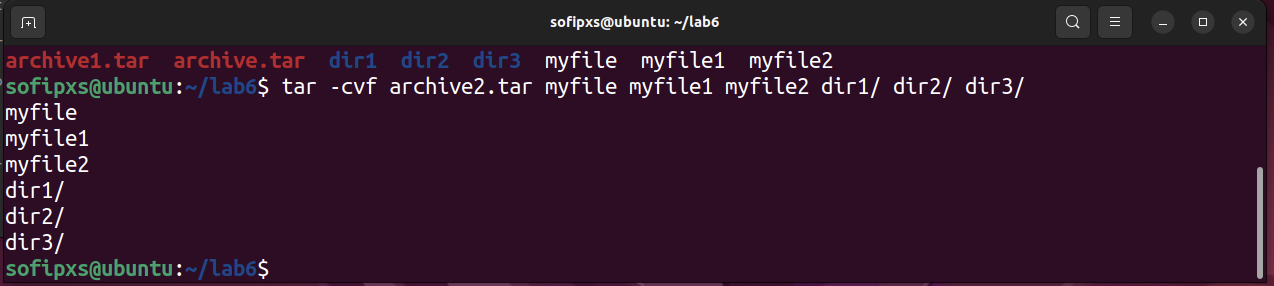
To create an empty file with the .tar extension, run the command:



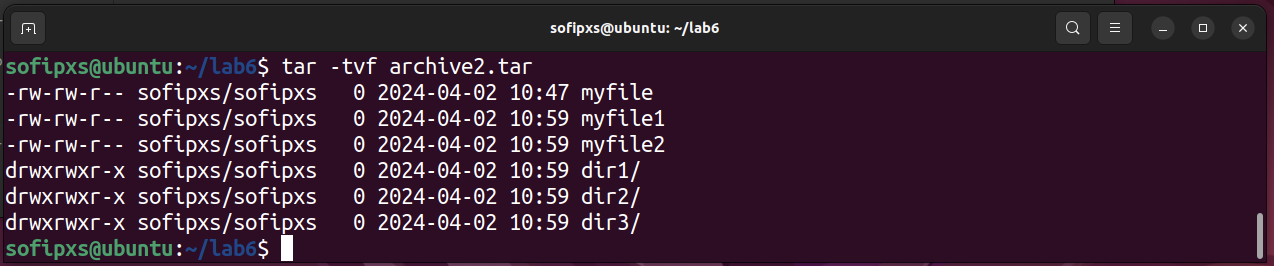
To create a file with content, run the command:



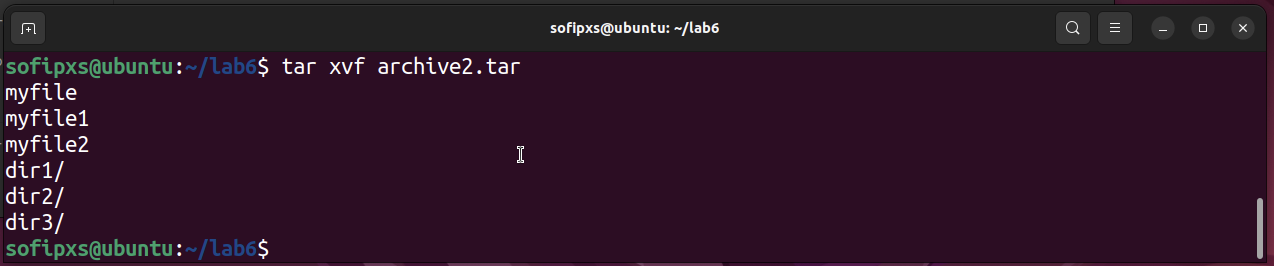
* створити файл з розширенням .tar, що складається з декількох файлів і каталогів  одночасно;



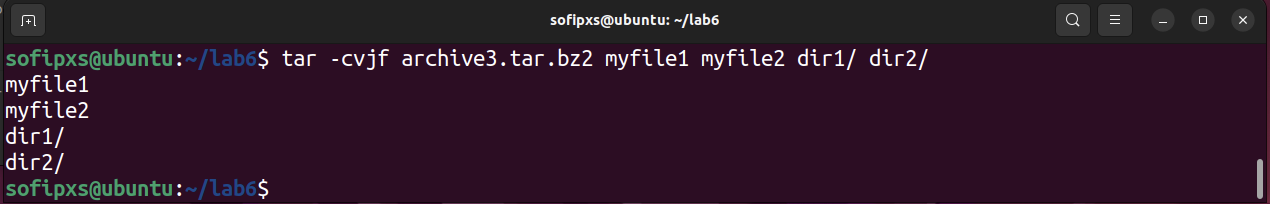
* перегляду вмісту файлу;



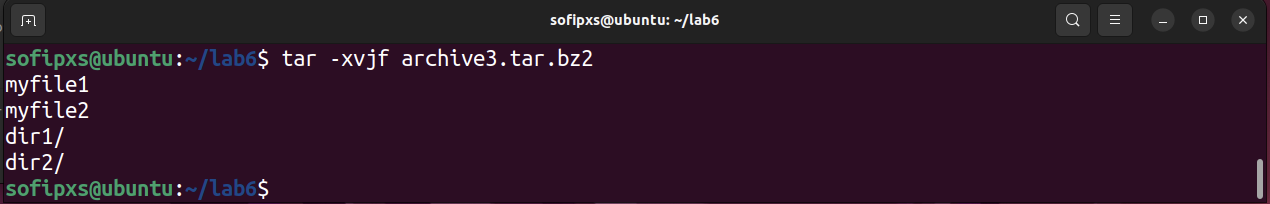
* витягти вміст файлу tar



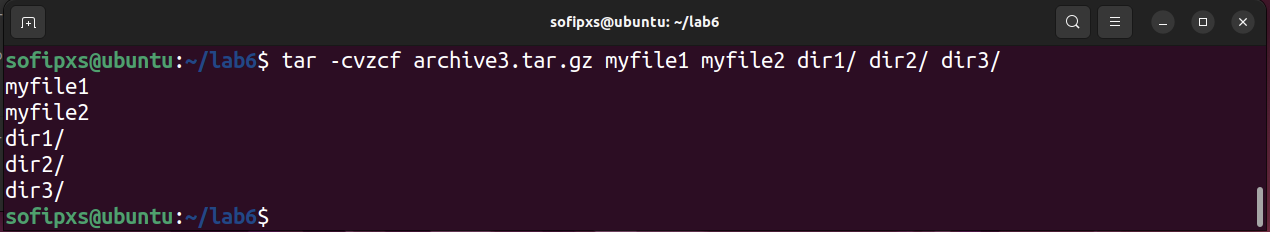
* створити архівний файл tar, стиснений за допомогою bzip;

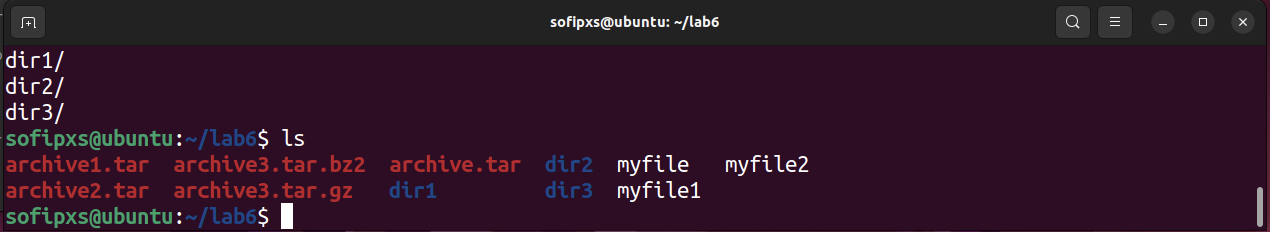


* витягти вміст файлу tar bzip;



* створити архівний tar файл, стисненого за допомогою gzip;





* витягти вміст файлу tar gzip.



1. \*Як буде відбуватись перенаправлення потоків виведення в bash для наступних дій з командами (позначено як cmd) та файлами (позначено як file):

|  |  |
| --- | --- |
| **Команда** | **Що виконує команда?** |
| cmd 1> file | This command redirects the standard output (stdout) of cmd to a file. If file does not exist, it will be created; otherwise, it will be overwritten. |
| cmd > file | This is the same as cmd 1> file. It redirects the standard output (stdout) of cmd to a file. |
| cmd 2> file | redirects the standard error stream (stderr) from cmd to file, overwriting any existing content. |
| cmd >> file | adds standard output from cmd to file, preserving any existing content. |
| cmd &> file | Redirects both stdout and stderr of cmd to file. |
| cmd > file 2>&1 | Redirects stdout of cmd to file and stderr to the same place as stdout. |
| cmd >> file 2>&1 | Similar to the previous one, but appends stdout of cmd to file. |
| cmd 2>&1 > /dev/null | Redirects stderr of cmd to the same place as stdout and discards stdout. |
| cmd 2> /dev/null | redirects stderr from cmd to /dev/null, effectively rejecting errors. |
| cmd1 | cmd2 | Creates a pipe. The standard output (stdout) of cmd1 is sent as input to cmd2. The output of cmd2 is then displayed on the terminal. |
| cmd1 2>&1 | cmd2 | Sends both standard output (stdout) and standard error (stderr) from cmd1 to the standard input of cmd2. The output of cmd2 is displayed on the terminal. |

1. \*\*Розгляньте наведені нижче приклади та поясніть, що виконують дані команди та який тип перенаправлення потоків вони використовують:

|  |  |  |
| --- | --- | --- |
| **Команда** | **Що виконує команда?** | **Який потік перенаправлення?** |
| $echo "It is a new story." > story | This command writes the string "It is a new story." to a file named "story". | STDOUT: The output of echo (the text) is sent to the file "story" instead of being displayed on the terminal. If the file exists, it will be overwritten. |
| $ date > date.txt | This command writes the current date and time to a file named "date.txt". | STDOUT: The output of date (the date/time) is sent to the file "date.txt". If the file exists, it will be overwritten. |
| $ cat file1 file2 file3 > bigfile | Combines the contents of files "file1", "file2" and "file3" into a new file "bigfile". | STDOUT: The combined output of cat (the contents of the three files) is sent to the file "bigfile". |
| $ls -l >> directory | Adds a detailed list of files (with the "-l" flag) in the current directory to the "directory" file. | STDOUT: The output of ls -l (the file listing) is appended to the end of the file "directory" (assuming it already exists). |
| $ sort < file1\_unsorted > file2\_sorted | Sorts the contents of the file "file1\_unsorted" and writes the sorted result to the file "file2\_sorted". | STDIN: Receives data from the file "file1\_unsorted".  STDOUT: Redirected to the file "file2\_sorted". |
| $ find -name '\*.txt' > file.txt 2> /dev/null | Finds all files with the extension ".txt" in the current directory and writes them to the file "file.txt". Errors are dumped to "/dev/null". | STDOUT: Redirected to the file "file.txt".  STDERR: Redirected to "/dev/null" (discard). |
| $ cat file1\_unsorted | sort > file2\_sorted | Reads the contents of file1\_unsorted, sorts it, and writes the result to file2\_sorted. | The cat command outputs the contents of the file file1\_unsorted, which is passed as input to the sort command through the channel (|). The sorted output is redirected to the file file2\_sorted. If the file exists, it will be overwritten. The stdout redirection is used here. |
| $ cat myfile | grep student | wc -l | This command counts the number of lines in the "myfile" file that contain the word "student". | The output of the cat command (stdout) is redirected as input to the grep command through the (|) channel. The output of the grep command (stdout) is then redirected as input to the wc -l command through another channel (|). In general, this command uses stdout to pass data between commands. |

**Відповіді на контрольні запитання:**

1. Надайте порівняльну характеристику процесам стискання та архівування.
2. Які програми, окрім наведених в роботі, можуть використовуватись для стискання та архівування файлів та каталогів в ОС Linux? Наведіть приклади та їх короткий опис.
3. \*Порівняйте алгоритми стискання, що використовуються в командах (програмах), використовуваних в Linux. Які з алгоритмів можна вважати найшвидшим та найефективнішим?
4. \*Опишіть програмні засоби для стискання та архівування, що можуть бути використані у вашому мобільному телефоні.
5. \*Опишіть та порівняйте програмні засоби для стискання та (де)архівування даних у ОС сімейства Windows.
6. \*\*Поясніть яким чином стиснення та архівування даних може бути використано для резервування даних. В яких ще задачах системного адміністрування воно може бути використано.
7. \*\*Яке призначення директорії файлу /dev/null?

**Висновки:**

During the lab, the students explored the main aspects of data compression and archiving in the Linux operating system, gaining a more detailed understanding of various compression methods such as gzip, bzip2 and xz. Practical skills in working with the tar command for creating, unpacking and compressing archives were acquired.