“Київський фаховий коледж зв’язку”

Циклова комісія Комп’ютерної інженерії

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №2**

з дисципліни: «Операційні системи»

**Тема: «Знайомство з базовими командами CLI-режиму в Linux»**

Виконавли студенти

групи РПЗ-03

Команда: Губенко Є.О.,

Заїка С.В. та Кресан Р.А.

Перевірив викладач

Сушанова В.С.

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 Заїка С.В.***

1. Дайте визначення наступним поняттям:

- Командний інтерпретатор

Командний інтерпретатор є програмою, яка дозволяє користувачеві взаємодіяти з операційною системою через текстовий інтерфейс. Він дозволяє користувачеві виконувати різні команди, такі як створення, копіювання, переміщення або видалення файлів і каталогів, зміну налаштувань системи та багато іншого

- Оболонка

Оболонка це комп'ютерна програма, що надає доступ до служб операційної системи користувачу (людині або іншій програмі).

- Команда

Команда це назва програми, яку користувач вводить у терміналі для виконання певного завдання. Команди можуть мати додаткові дані та параметри.

1. Дайте відповіді на наступні питання:

- Яку базову інформацію надає рядок запрошення prompt?

*Рядок запрошення prompt надає користувачу текстове повідомлення, яке він повинен прочитати, а потім ввести дані в поле для вводу. Рядок запрошення може містити підказки для користувача, наприклад, тип даних, які він повинен ввести.*

- Для чого команді потрібні параметри та аргументи?

*Параметри та аргументи потрібні для того, щоб дати команді додаткові параметри, які будуть використовуватися для виконання дії. Наприклад, команда «ls» може приймати параметри та аргументи, такі як «-l» і «/ tmp», щоб вивести детальну інформацію про файли в директорії / tmp.*

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

*`ls` - це команда терміналу для відображення списку об’єктів у поточному каталозі. Вона може мати параметри та аргументи, які дозволяють користувачам вивести детальну інформацію про об’єкти у каталозі.*

*Приклади:*

*1. `ls -l` - вивести детальну інформацію про всі об’єкти у поточному каталозі у вигляді списку.*

*2. `ls -a` - вивести всі об’єкти у поточному каталозі, включаючи приховані об’єкти.*

*3. `ls /home/user` - вивести список об’єктів у каталозі /home/user.*

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

*Історія команд дозволяє користувачам повторно використовувати команди, які вони раніше використовували, щоб заощадити час. Це дозволяє користувачам більш ефективно використовувати систему. Крім того, історія команд дозволяє користувачам повторно використовувати команди, які вони забули. Це дозволяє користувачам більш ефективно використовувати систему і заощадити час.*

- Яке призначення команди echo?

*Echo команда використовується для виведення текстових рядків або змінних на екран. Вона може бути використана для виведення повідомлень при запуску скрипта, або для виведення результатів виконання команд.*

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

*Змінні в оболонці Bash - це спеціальні поняття, яке дозволяє вам зберігати дані для доступу до них в подальшому. Змінна може бути задана або її можна присвоїти значення. Оболонка Bash підтримує такі типи змінних як логічні змінні, строкові змінні, цілочисельні змінні, масиви та змінні оточення.*

- Яке призначення команд env, export та unset?

*Команда env використовується для виведення змінних середовища та їх значень. Команда export використовується для додавання змінних середовища до списку доступних для програми. Команда unset використовується для вилучення змінних середовища зі списку доступних для програми.*

- Які команди для отримання довідки по командам в терміналі ви знаєте?

*1. man - використовується для отримання довідки по будь-якій команді у терміналі.*

*2. help - використовується для отримання довідки по будь-якій команді у терміналі.*

*3. info - використовується для отримання довідки по будь-якій команді у терміналі.*

1. *apropos - використовується для пошуку довідки по командам, що відповідають заданому пошуковому запиту.*

**Хід роботи**

***Готував матеріал студент Заїка С.В.***

1. Опрацюйте всі приклади команд, що представлені у лабораторній роботі курсу NDG Linux Essentials - Lab 5: Command Line Skills та Lab 6: Getting Help.

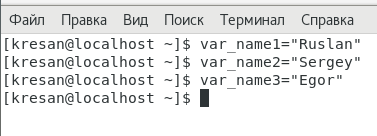
|  |  |
| --- | --- |
| Назва команди | Її призначення та функціональність |
| ls | Виводить інформації про каталоги та файли. За замовчуванням без аргументів відображає інформацію для поточного каталогу |
| ls -l | Використанні параметру -l в команді ls дозволяє відобразити інформацію про файли, розташовані в поточному робочому каталозі, у довгому форматі, який надає більш розширену додаткову інформацію |
| ls -l /tmp | Використання аргументу /tmp в поєднанні з параметром -l в команді ls дозволяє відобразити детальну інформацію про файли в каталозі /tmp. |
| whoami | Вона допомагає визначити права доступу користувача до файлів і директорій, а також вирішувати проблеми з доступом до ресурсів і забезпечувати безпеку в операційній системі. |
| uname | Вона відображає назву операційної системи, її версію та тип процесора, на якому вона працює. Ця команда корисна для діагностики проблем з операційною системою та встановлення сумісності додатків з різними версіями ОС. |
| uname -n | Виводить назву вузла або комп'ютера в мережі. Вона дозволяє користувачам визначати ідентифікатори комп'ютерів у мережі та допомагає в налагодженні мережевих з'єднань і з'єднань з віддаленими комп'ютерами. |
| uname --nodename | Виводить назву вузла або комп'ютера в мережі. Ця команда є альтернативою скороченій команді "uname -n". |
| pwd | Виводить поточний робочий каталог або шлях до поточного робочого каталогу. Вона допомагає користувачам визначати поточне місцезнаходження в файловій системі та навігацію між різними каталогами. ця команда корисна для виконання рутинних задач у терміналі та автоматизації роботи з файлами і каталогами. |
| echo | Виводить текстовий рядок на екран або в інший потік виведення. |
| history | Виводить історію введених команд в поточній сесії терміналу. Вона дозволяє переглядати попередні команди та їх параметри, що були введені у попередні сесії, та повторювати їх знову без потреби введення знову. |
| history 5 | Виводить список останніх п'яти команд, які були введені в поточній сесії терміналу. Вона дозволяє переглядати тільки останні команди з історії введення, замість повного списку всіх введених команд. |
| !9 | Виконує дев'яту команду з історії введення команд. Вона дозволяє повторно виконати попередні команди, не потрібно їх знову вводити. |
| echo $HISTSIZE‌ | Виводить поточне значення змінної середовища HISTSIZE. Змінна HISTSIZE визначає максимальну кількість команд, які будуть зберігатися в історії введення команд. |
| echo $PATH | Виводить список шляхів до директорій, які містять виконувані файли. Змінна PATH містить розділені двокрапками шляхи до каталогів, де операційна система шукає виконувані файли. |
| which date | Виводить шлях до виконуваного файлу для команди date. Команда корисна для визначення місцезнаходження виконуваного файлу для конкретної команди в системі. |
| type command | Використовується для визначення типу команди (вбудована, зовнішня або аліас), що вказана в аргументах командного рядка. Вона дозволяє з'ясувати, як саме виконується команда в терміналі та допомагає вирішувати проблеми, пов'язані з невідомими командами або конфліктами імен. |
| which ls | Виводить шлях до виконуваного файлу для команди ls. Команда корисна для визначення місцезнаходження виконуваного файлу для конкретної команди в системі. |
| alias | Використовується для створення скороченого варіанту введення команди, яка складається з більш складної послідовності команд або параметрів. Після використання команди "alias" користувач може створити скорочення для будь-якої команди або послідовності команд і викликати їх за допомогою нового скорочення. |
| echo Today is `date` | Виводить поточну дату та час на екран. |
| echo D\* | Виводить на екран список всіх файлів та папок в поточній директорії, імена яких починаються з літери "D". Символ "\*" використовується для зіставлення з будь-яким рядком після літери "D", тому результатом можуть бути будь-які імена файлів та папок, що починаються з "D". |
| false | Повертає завжди значення 1, що є ознакою того, що команда не виконалася успішно або що сталася помилка. |
| echo Start && echo Going && echo Gone | Виводить три повідомлення "Start", "Going" та "Gone" послідовно у вигляді виведення в стандартний вивід. |
| echo Success && false && echo Bye | Виводить повідомлення "Success" в стандартний вивід, потім виконує команду "false", яка завжди повертає значення "false", тобто не виконується успішно, тому наступна команда "echo Bye" не виконується. |
| false || echo Fail Or  true || echo Nothing to see here | Команда `false || echo Fail Or` має на меті вивести рядок "Fail Or" тільки в разі, якщо попередня команда `false`завершиться з помилкою.  Команда `true || echo Nothing to see here` має на меті вивести рядок "Nothing to see here" тільки в разі, якщо попередня команда `true` завершиться успішно. |
| date | Виводить поточну дату та час в форматі, який може бути налаштований за допомогою параметрів командного рядка. |
| man date | використовується для відображення сторінки довідки (man-сторінки) для команди `date` в Unix-подібних операційних системах. Ця сторінка містить детальну інформацію про синтаксис та опції команди `date`, а також приклади її використання. |
| man -k password | призначена для пошуку в посібнику (manual) Linux за ключовим словом "password". Функціональність полягає в тому, що вона дозволяє знайти всі документи (включаючи керівництва користувача), які містять слово "password" в їхньому описі. |
| man -f passwd | виводить короткий опис сторінки довідки, пов'язаної з командою `passwd` . Опція `-f`` вказує на те, що виводити тільки короткий опис замість повної сторінки довідки. Команда `passwd` використовується для зміни пароля користувача в Linux-системах. |
| man 5 passwd | призначена для відображення інформації з розділу 5 (файлові формати та стандарти) керівництва користувача Linux про файл `/etc/passwd` Цей файл містить інформацію про користувачів системи, включаючи їх імена, домашні каталоги та інші деталі їх облікових записів. |
| info date | Відображає інформацію про команду "date" в форматі Info. Інформація включає опис опцій, синтаксис та приклади використання команди "date" в Unix-подібних операційних системах. |
| date --help | виводить коротку довідку про опції та формати, що підтримуються командою `date` Її призначення - надати користувачеві короткий опис доступних опцій та форматів команди `date` |
| ls /usr/share/doc | виводить список файлів та папок, що містять документацію для встановлених пакетів на системі в директорії "/usr/share/doc". Її призначення полягає у перегляді інформації про програми, які встановлені на комп'ютері. |
| locate crontab | Команда "locate" шукає шляхи до файлів та директорій з використанням бази даних, яка зберігає інформацію про файлову систему. У даному випадку, запит "locate crontab" шукає шлях до будь-якого файлу або директорії зі словом "crontab" у назві або шляху. |
| locate -b "\crontab" | шукає швидкий зв'язок бази даних із зазначеним виразом `”crontab” і повертає шлях до будь-якого файлу, який містить цей вираз в його імені. Параметр `-b` забезпечує точний збіг лише з базовим ім'ям файлу (без шляху). |
| whereis passwd | призначена для пошуку виконуваних файлів, бінарних файлів та довідкових файлів пов'язаних з заданим іменем файлу. В даному випадку, `whereis passwd` шукає місцеположення виконуваного файлу, даних та довідкового файлу команди `passwd` |

***Готував матеріал студент*** ***Кресан Р.А.***

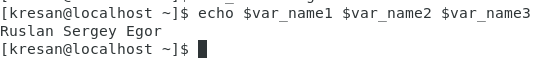
2. Робота в терміналі (закріплення практичних навичок) обов’язково представити свої скріншоти:

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

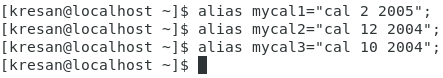
- Створіть змінні, що будуть містити Ваші імена та прізвища $var\_name1, $var\_name2, $var\_name3



- За допомогою команди echo виведіть імена студентів вашої команди

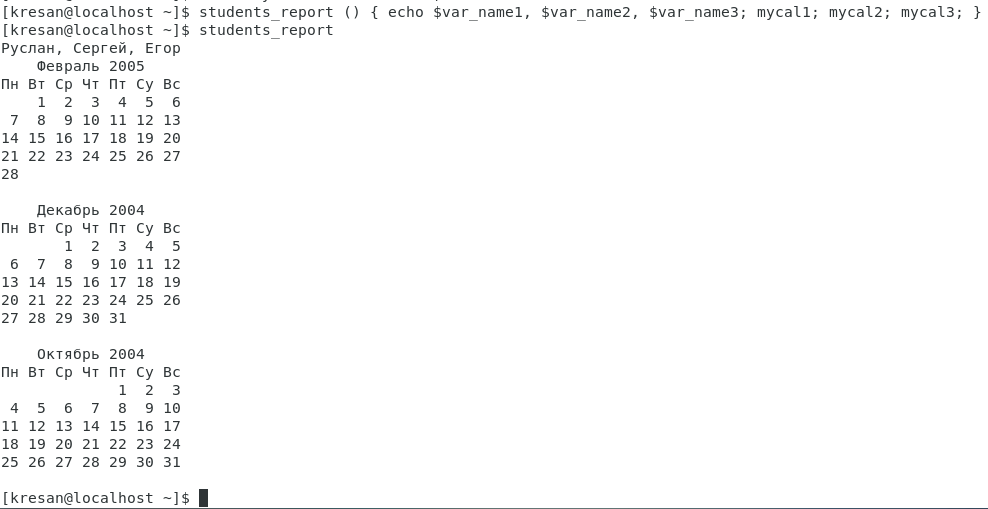


- Створіть псевдоніми mycal1, mycal2, mycal3 для команди cal для автоматичного виведення календарю вашого року народження



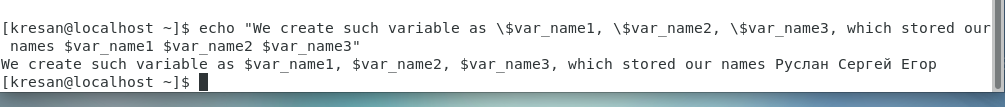
2.2. Робота з функціями (Functions) в терміналі:

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

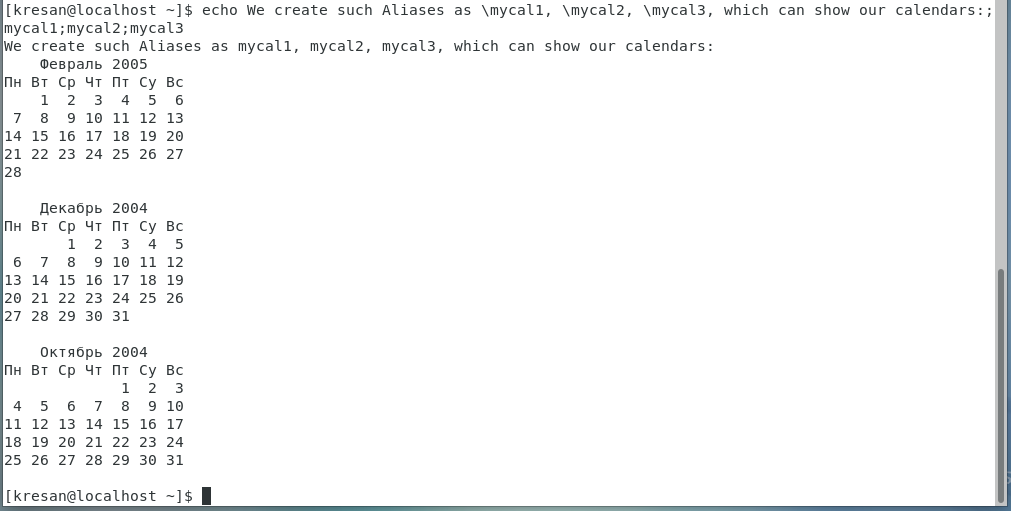


2.3. Робота з лапками (Quoting) в терміналі. Виведіть в командному рядку наступні речення:

- “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)

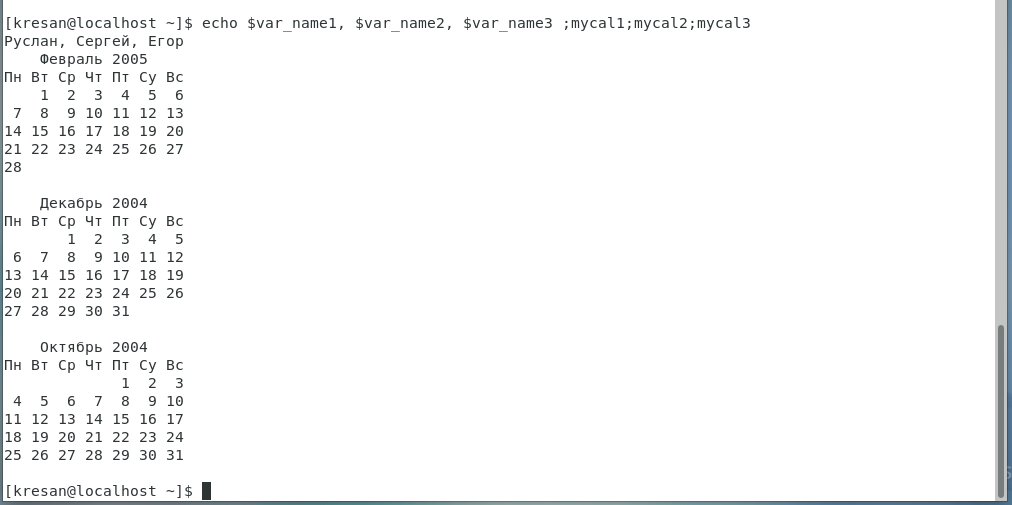


- “We create such Aliases as mycal1, mycal2, mycal3, which can show our calendars: Calendar1, Calendar2, Calendar3” (у реченні спочатку виводимо назву команди-псевдонімів, потім вивід цих команд).



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).

|  |
| --- |
|  |
| Використанні команди:   * -s, (-kernel-name) – друкує ім'я ядра. * -n, (-nodename) - друкує ім'я вузла системи (ім'я хоста). Це ім'я, яке система використовує під час спілкування по мережі. * -v, (-kernel-version) - друкує версію ядра. * -p, (-processor) - друкує архітектуру процесора. * -o, (-operating-system) - роздрукує назву операційної системи. * -a, (-all) - при використанні опції -a uname поводиться так само, як якщо б були задані опції -snrvmo. |

**Відповіді на контрольні запитання**

1. Які типи команд існують в оболонці Bash?

*Оболонка Bash підтримує різноманітні типи команд, серед яких є наступні:*

*Вбудовані команди: це команди, які вбудовані в саму оболонку Bash, такі як cd, echo, pwd, exit тощо.*

*Зовнішні команди: це команди, які не є вбудованими в оболонку Bash, а знаходяться в окремих програмах, які можна запустити з командного рядка. Прикладами можуть бути ls, cat, grep тощо.*

*Аліаси: це короткі імена, які можна створити для довгих команд, щоб зручно використовувати їх у майбутньому.*

*Функції: це фрагменти коду, які можна написати самостійно та використовувати замість повних команд. Функції зазвичай використовуються для виконання складних завдань, які потребують кількох кроків.*

*Псевдоніми: це інша форма аліасів, які дозволяють створювати короткі імена для команд, які можуть бути введені з будь-якої точки оболонки. Вони діють, незалежно від того, де знаходиться користувач в оболонці.*

*Ці типи команд можна використовувати окремо або в комбінації для створення складних команд та скриптів, які виконують різні завдання в оболонці Bash.*

2. Що таке змінні оточення? Які вони бувають. Як їх можна переглянути в терміналі?

*Змінні оточення (environment variables) - це змінні, які встановлюються в операційній системі та доступні для всіх процесів, що працюють в системі. Змінні оточення містять інформацію про систему, налаштування оболонки, шляхи до виконуваних файлів та інше.*

*Є кілька змінних оточення, які є стандартними для більшості Unix-подібних систем, такі як:*

*HOME: шлях до домашньої директорії користувача*

*PATH: шляхи до виконуваних файлів*

*USER: ім'я користувача, під яким запущено процес*

*SHELL: шлях до оболонки, яку використовує користувач*

*TERM: тип терміналу, який використовується для виводу*

*Змінні оточення можна переглянути в терміналі за допомогою команди "printenv" або "env". Наприклад, команда "printenv PATH" виведе шляхи до виконуваних файлів, які доступні в поточному середовищі. Також можна використовувати команду "echo" для виведення значень окремих змінних, наприклад, "echo $HOME".*

3. Опишіть змінну $PS1. Як в терміналі переглянути її вміст?

*Змінна $PS1 є однією з змінних оточення в оболонці Bash, яка відповідає за вміст приглашення командного рядка (prompt). Ця змінна використовується для налаштування вигляду командного рядка в терміналі та може містити різноманітні спецсимволи та керуючі послідовності, які відображають інформацію про поточний стан оболонки, наприклад, поточний шлях, ім'я користувача, хост-систему тощо.*

*Приклад значення змінної $PS1:  
export PS1='\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '*

*В цьому прикладі вміст $PS1 складається з наступних компонентів:*

*\u - ім'я користувача*

*\h - ім'я хост-системи*

*\w - поточний шлях*

*$ - символ приглашення командного рядка ($ для звичайного користувача, # для користувача з правами root)*

*керуючі послідовності з кольорами (в даному випадку, зелений колір для ім'я користувача та хоста, та синій колір для поточного шляху)*

*Щоб переглянути вміст змінної $PS1 в терміналі, можна використовувати команду "echo $PS1". Це дозволить відобразити поточне значення змінної $PS1 в терміналі.*

4. Як можна змінити значення змінної $PS1? Що при цьому відбудеться в рядку запрошенні в bash (рядок запрошення перед початком кожної команди). Як змінити значення цієї змінної не на поточний сеанс, а за замовчуванням?

*Змінити значення змінної $PS1 можна за допомогою команди export. Наприклад, якщо потрібно змінити значення змінної $PS1 на нове значення, то можна ввести наступну команду в терміналі Bash:*

*export PS1="нове значення"*

*При зміні значення змінної $PS1 відбудеться зміна відображення рядка запрошення перед початком кожної команди в терміналі Bash. Нове значення може містити будь-які спецсимволи та керуючі послідовності, що відповідають за вигляд приглашення командного рядка.*

*Щоб змінити значення змінної $PS1 за замовчуванням, можна додати команду export PS1="значення" до файлу конфігурації оболонки (наприклад, .bashrc). Це дозволить зберегти нове значення змінної $PS1 настійно, тобто при кожному запуску оболонки в майбутньому воно буде використовуватися за замовчуванням.*

5. Для чого використовують лапки в оболонці Bash?

*Лапки (одинарні і подвійні) в оболонці Bash використовуються для обмеження простору імен та для передачі рядків зі спеціальними символами як єдиного аргументу.*

*Одинарні лапки (') використовуються для обмеження простору імен. Якщо використати одинарні лапки, то змінні в середині рядка не будуть інтерпретовані, тобто значення змінних не буде підставлено в рядок, а вважатиметься простим текстом. Наприклад:*

*echo '$USER'*

*Виведе на екран рядок $USER, а не ім'я користувача, як у випадку з подвійними лапками.*

*Подвійні лапки (") використовуються для передачі рядків зі спеціальними символами як єдиного аргументу. В середині рядка значення змінних будуть інтерпретовані та підставлені в рядок. Наприклад:*

*echo "Привіт, $USER"*

*Виведе на екран рядок Привіт, ім'я\_користувача.*

*Важливо зазначити, що якщо у рядку знаходяться символи, які мають спеціальне значення для оболонки (наприклад, пробіл, крапка з комою тощо), то рядок потрібно брати у подвійні лапки. При цьому важливо мати на увазі, що символ подвійної лапки також має спеціальне значення, тому його потрібно екранувати за допомогою зворотного слешу (\).*

6. Для чого використовують інструкції керування, які їх види Ви знаєте?

*Інструкції керування використовуються в програмуванні для зміни поведінки виконання програми. Вони дозволяють здійснювати управління виконанням програми, включаючи умовні оператори, цикли та перехід до інших частин програми.*

*Основні види інструкцій керування включають:*

*Умовні оператори:*

*if-else: дозволяє виконати один блок коду, якщо умова є істинною, та інший блок коду, якщо умова є хибною.*

*switch: дозволяє перевірити, чи дорівнює значення змінної одному зі списку заданих значень.*

*Цикли:*

*for: дозволяє повторювати виконання блоку коду задану кількість разів.*

*while: дозволяє повторювати виконання блоку коду до тих пір, поки умова є істинною.*

*do-while: дозволяє повторювати виконання блоку коду принаймні один раз, після чого продовжувати повторювання до тих пір, поки умова є істинною.*

*Інструкції переходу:*

*break: дозволяє припинити виконання циклу або перейти до наступного оператора після блоку коду switch.*

*continue: дозволяє перейти до наступної ітерації циклу.*

*goto: дозволяє перейти до мітки в коді, яка знаходиться десь інде в цій же функції.*

*Інструкції керування можуть значно полегшити процес програмування та зробити код більш зрозумілим і структурованим. Однак їх використання повинно бути обмеженим та обґрунтованим, оскільки надмірне використання може зробити код складнішим та менш зрозумілим.*

7. В чому різниця якщо в кінці рядку запрошення bash стоїть символ $ чи #? Наприклад на екрані ми бачимо наступні записи





*Запис [centos@localhost Desktop]$ означає, що користувач з ім'ям "centos" працює в оболонці bash зі звичайним рівнем доступу, оскільки в кінці рядка запрошення стоїть символ "$".*

*Запис [root@localhost Desktop]# означає, що користувач з ім'ям "root" працює в оболонці bash з підвищеними правами, оскільки в кінці рядка запрошення стоїть символ "#".*

*Отже, різниця між символами $ та # в кінці рядка запрошення полягає у рівні доступу користувача в оболонці bash: символ $ відображає звичайний рівень доступу, тоді як символ # відображає підвищений рівень доступу.*

8. Яке призначення команд whereis та locate? Яка між ними відмінність?

*Команди whereis та locate використовуються для пошуку файлів та програм в операційній системі.*

*Команда whereis шукає файли програм в заданих директоріях системи та повертає шлях до виконуваного файлу, файлів документації та коду програми. Наприклад, виконання команди whereis bash поверне шлях до виконуваного файлу bash, файлу документації та до файлу з кодом програми.*

*Команда locate, натомість, шукає файли в базі даних файлів системи, а не в конкретних директоріях, що може значно прискорити пошук. Пошук виконується швидко, але база даних повинна бути оновлювана регулярно, щоб містити актуальну інформацію про файли системи.*

*Отже, відмінність між командами whereis та locate полягає у тому, що whereis шукає файли програм в заданих директоріях, тоді як locate шукає файли в базі даних файлів системи. Друга команда може бути швидшою, але вона потребує оновлення бази даних, що може зайняти певний час.*

**Висновки**

В ході виконання лабораторної роботи ознайомились з базовими командами CLI-режиму в Linux та базовими текстовими командами в термінальному режимі роботи в різних ОС.