“Київський фаховий коледж зв’язку”

Циклова комісія Комп’ютерної та програмної інженерії

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №3**

з дисципліни: «Операційні системи»

**Тема: “Знайомство з базовими командами CLI-режиму в Linux”**

Виконали студенти

групи Бюджетники:

Топчій А.С.,

Топехін Б.А. та Чичкань Д.С.

Перевірив викладач

Сушанова В.С.

Київ 2022

**Тема: “Знайомство з базовими командами CLI-режиму в Linux”**

**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 Топчій А.***

1. Прочитайте короткі теоретичні відомості до лабораторної роботи та зробіть невеликий словник базових англійських термінів з питань призначення команд та їх параметрів.
2. Вивчіть матеріали онлайн-курсу академії Cisco “NDG Linux Essentials”:

* Chapter 5 - Command Line Skills
* Chapter 6 - Getting Help

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

* Chapter 05 Exam
* Chapter 06 Exam

1. Дайте визначення наступним поняттям:

* Командний інтерпретатор

Командний інтерпретатор - це програма, яка приймає вхідні команди від користувача та виконує їх. Командні інтерпретатори зазвичай використовують інтерфейс командного рядка, який дозволяє користувачам вводити команди за допомогою клавіатури. Командні інтерпретатори також можуть використовувати графічний інтерфейс користувача (GUI), який дозволяє користувачам вводити команди за допомогою миші та клавіатури.

* Оболонка

Оболонка - це тип командного інтерпретатора, який також забезпечує інтерфейс між користувачем та операційною системою. Оболонки зазвичай мають додаткові функції, такі як управління змінними, виконання сценаріїв та надання допомоги. Оболонки є основним способом взаємодії користувачів з операційними системами на базі Unix.

* Команда

Команда - це послідовність символів, яка інструктує командний інтерпретатор виконати певну дію. Команди можуть бути простими, такими як "ls", яке виводить список файлів і каталогів у поточному каталозі, або складними, такими як "apt install vim", яке встановлює текстовий редактор Vim. Команди можуть бути вбудовані в оболонки або виконуватися зовнішніми програмами.

1. Дайте відповіді на наступні питання:

* Яку базову інформацію надає рядок запрошення prompt?

Рядок запрошення prompt - це символьна строка, яка відображається на екрані перед тим, як користувач вводить команду. Він зазвичай містить такі відомості:

* Ім'я користувача
* Поточний каталог
* Символ запрошення
* Для чого команді потрібні параметри та аргументи?

Параметри та аргументи - це додаткові дані, які можуть бути передані команді для зміни її поведінки. Параметри зазвичай є необов'язковими, а аргументи - обов'язковими.

Параметри можуть використовуватися для:

* Зміни поведінки команди
* Передачі змінних команді
* Зміни вихідних даних команди

Аргументи зазвичай вказують на файли або каталоги, які потрібно обробити команді.

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

Команда ls виводить список файлів і каталогів у поточному каталозі. Вона може мати такі параметри та аргументи:

* -a - виводить приховані файли та каталоги
* -l - виводить детальну інформацію про файли і каталоги
* -R - виводить список файлів і каталогів рекурсивно, тобто для всіх підкаталогів

3 приклади використання команди ls:

* ls - виводить список файлів і каталогів у поточному каталозі
* ls -a - виводить список усіх файлів і каталогів, включаючи приховані
* ls -l - виводить список усіх файлів і каталогів з детальною інформацією

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

Історія команд - це список останніх команд, які були виконані. Її можна використовувати наступними способами:

* Перегляд історії:
  + Ctrl+R - перегляд історії команд
  + Ctrl+P - перегляд попередньої команди
  + Ctrl+N - перегляд наступної команди
* Повторне виконання команди:
* стрілка вгору - повторне виконання попередньої команди
* стрілка вниз - повторне виконання наступної команди
* !! - повторне виконання останньої команди

Переваги використання історії команд:

* Повторне виконання команд без необхідності їхнього повторного введення
* Перевірка помилок
* Яке призначення команди echo?

Команда echo виводить на екран вказаний текст. Вона може мати такі аргументи:

- -n - не додає символів переносу рядка в кінці виведеного тексту

- -e - інтерпретує певні символи в тексті як спеціальні символи

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

Змінна в оболонці Bash - це ім'я, яке використовується для зберігання значення. Змінні можуть використовуватися для зберігання тексту, чисел або інших типів даних.

Оболонка Bash підтримує такі типи змінних:

- Локальні змінні: існують тільки в межах поточного сеансу командного рядка

- Екологічні змінні: доступні з будь-якого сеансу командного рядка

- Змінні середовища: доступні з будь-якої програми, яка запускається в цьому середовищі

* Яке призначення команд env, export та unset?

- env - виводить список всіх змінних середовища

- export - робить змінну доступною з будь-якого сеансу командного рядка

- unset - видаляє змінну

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

- man - виводить повноформатну документацію по команді

- whatis - виводить коротку інформацію по команді

- info - виводить довідник по команді в інтерактивному режимі

1. Підготувати в електронному вигляді початковий варіант звіту:

* Титульний аркуш, тема та мета роботи
* Словник термінів
* Відповіді на п.4 та п.5 з завдань для попередньої підготовки
  1. Опрацюйте всі приклади команд, що представлені у лабораторній роботі курсу ***NDG Linux Essentials - Lab 5: Command Line Skills*** та ***Lab 6: Getting Help.*** Створіть таблицю для опису цих команд\*\*\*

***Готував матеріал студент Топчій А.***

| 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** дозволяєвідобразити детальну інформацію про файли в каталозі /tmp. |
| cd | Змінює поточний каталог. |
| pwd | Виводить поточний каталог. |
| mkdir | Створює новий каталог. |
| rmdir | Видаляє порожній каталог. |
| cp | Копіює файл або каталог. |
| mv | Переміщує файл або каталог. |
| rm | Видаляє файл або каталог. |

* 1. Робота в 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)

## Відповіді на контрольні запитання - Готував матеріал студент Топехін Б.

1. Які типи команд існують в оболонці Bash?

- Вбудовані команди,Зовнішні команди,Псевдокоманди,Команди пайпу(pipe),Команди зворотнього апострофа (backticks)

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

- Змінні оточення - це змінні, які зберігають інформацію про оточення іконфігурацію операційної системи, доступну для процесів та програм. Вонивпливають на роботу програм і служать для зберігання інформації пропараметри середовища.

- Для перегляду значень змінних оточення в терміналі використовуйтекоманду “echo”.

- Для перегляду змінних оточення в терміналі використовуйте команду “echo$ЗМІННА”, де “ЗМІННА” - це ім'я змінної оточення, яку ви хочетепереглянути. Наприклад, “echo $PATH” виведе значення змінної “PATH”.

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

- Змінна “$PS1” в оболонці Bash визначає формат та вигляд приголомшливого рядка (примірника) перед командним рядком в терміналі. Щоб переглянути їївміст в терміналі, використовуйте команду “echo $PS1”.

4. Як можна змінити значення змінної $PS1? Що при цьому відбудеться в рядкузапрошенні в bash (рядок

запрошення перед початком кожної команди). Як змінити значення цієї змінної не напоточний сеанс, а за замовчуванням?

- Змінна $PS1 використовується в оболонці Bash для налаштування рядказапрошення (prompt), який відображається перед вводом кожної команди. Виможете змінити значення цієї змінної, щоб налаштувати власний вигляд рядка запрошення. Щоб змінити значення $PS1, введіть наступну команду в терміналіBash: PS1="нове\_значення"

- Де "нове\_значення" - це рядок, який ви хочете встановити як новий рядокзапрошення. Наприклад, щоб встановити рядок запрошення, який включаєпоточний каталог і символ "$" як знак команди, ви можете використовуватитаку команду: PS1="\w $ "

- Після введення цієї команди ваш рядок запрошення буде виглядати як щосьна зразок: /home/user/Documents $

- Для того, щоб зберегти це значення $PS1 на постійну, ви можете додатикоманду в свій файл налаштувань оболонки, такий як "/.bash\_profile".

Додайте рядок зі зміною $PS1 в цей файл, і вона буде встановлена автоматичнопри запуску нового сеансу оболонки. Наприклад: echo 'PS1="\w $ "' >> ~/.bashrc

- Після цього відкрийте новий термінал або виконайте команду source

~/.bashrc, щоб застосувати зміни..Змінити значення $PS1 за замовчуваннямможна, внісши зміни в конфігураційний файл оболонки. Для цього:

- Відкрийте конфігураційний файл оболонки, наприклад, ~/.bashrc длякористувача або /etc/bash.bashrc для всіх користувачів.

- Додайте рядок, що встановлює нове значення $PS1. Наприклад:

- Для користувача: PS1="\w $ "

- Для всіх користувачів (потрібні права адміністратора): sudo echo 'PS1="\w

$ "' >> /etc/bash.bashrc

- Збережіть файл.

- Відкрийте новий термінал або виконайте команду source ~/.bashrc абоsource /etc/bash.bashrc (залежно від файлу, який ви редагували), щобзастосувати зміни. Змінене значення $PS1 буде використовуватися зазамовчуванням у нових сеансах оболонки.

5. Для чого використовують лапки в оболонці Bash?

Лапки в оболонці Bash використовуються для захисту вмісту від інтерпретації та длявизначення рядків. Вони дозволяють:

- 1. Захищати вміст від інтерпретації змінних: Використання лапок дозволяєвставляти значення змінних в рядки без їхньої автоматичної інтерпретації.Наприклад,

"«Значення $змінної" • виведе рядок зі словом \* "Значення" \* і самоюзмінною $змінною", а не її значенням.

- 2. Визначати рядки: Лапки дозволяють визначити значення рядків,включаючи спеціальні символи та змінні. Наприклад, " "Привіт, $імʼя!""визначає рядок, в якому $імʼя" замінюється на його значення.

- 3. Створювати командні рядки: У випадку виконання команд в Bash,подвійні лапки

використовуються для визначення командного рядка, який буде виконуваний.

Наприклад, \*"ls -1" • створить командний рядок для виклику команди 'Is зпараметром '-1'

- Загалом, використання лапок допомагає контролювати інтерпретаціюзмінних, спеціальних символів та створювати рядки з бажаним вмістом воболонці Bash.

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

- Інструкції керування в програмуванні використовуються для контролю надвиконанням програми та управління потоком виконання. Основні видиінструкцій

керування включають:

1. Умовні інструкції (if-else): Використовуються для виконання певного блокукоду, якщо задана умова істинна, та інш§го блоку коду, якщо умова хибна.

2. Цикли (for, while, do-while): Використовуються для повторення блоку кодудекілька разів. "for\* використовується для ітерації через

послідовність. while\* i do-while\* використовуються для повторення, поки заданаумова виконується.

3. Переривання (break, continue): Використовуються в циклах для виходу ізциклу (break) або переходу до наступної ітерації (continue).

4. Перемикачі (switch/case): Використовуються для вибору одного збагатьох варіантів дій, в залежності від значення виразу.

5. Функції та підпрограми: Використовуються для організації коду в окреміфункції або процедури, які можуть бути викликані з різних частин програми.

6. Винятки (try/except): Використовуються для обробки винятковихситуацій (виключень), які можуть виникнути в процесі виконанняпрограми.

7. Заборона памʼяті (goto): Ця інструкція рідко використовується і нерекомендується в сучасному програмуванні, оскільки може призвести донепередбачуваного та

важкочитаного коду.

7. В чому різниця якщо в кінці рядку запрошення bash стоїть символ $ чи #?

- різниця між символами '$ і # в промпті полягає в рівні привілеїв користувача. $ вказує на звичайного користувача, а # на суперкористувача або адміністраторасистеми.

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

- whereis призначений для пошуку виконувих файлів та документації програм, івін швидше, але менш універсальний в порівнянні з locate.

- locate більш загальний та потужний і може знаходити будь-які файли тадиректорії у системі, але він вимагає оновленої бази даних і може бутиповільним при першому запуску.

- Якщо вам потрібно швидко знайти бінарний файл або документацію програми,whereis може бути зручнішим. Однак, якщо вам потрібно шукати файлизагалом по всій системі, locate є більш ефективним інструментом.

**Висновки -** ***Готував матеріал студент Топчій А.***

Під Час лабораторної роботи ми ознайомились з базовими командами CLI-режиму в Linux також ми ознайомились з базовими текстовими командами в термінальному режимі роботи на різних OC.