***The material was prepared by student Kanavets Kateryna (kanavetsk)***

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\*\*\*

|  |  |
| --- | --- |
| 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 | Повертає інформацію про поточного користувача, який виконує команду. Функціональність "whoami" полягає в тому, що вона повертає ім'я користувача, яке відображається в системі операцій. |
| uname | Повертає інформацію про операційну систему, на якій ви працюєте. Функціональність "uname" полягає в тому, що вона повертає різноманітну інформацію про систему, включаючи назву операційної системи, версію ядра та архітектуру процесора. |
| uname -n | Виводить в термінал лише ім'я вузла (host name) комп'ютера, на якому виконується команда. |
| uname --nodename | Команда "uname --nodename" є альтернативним варіантом команди "uname -n" і виводить в термінал ім'я вузла комп'ютера, на якому виконується команда. Команда "uname --nodename" може бути корисною для отримання імені вузла комп'ютера в скриптах або програмах, що виконуються в терміналі. |
| pwd | Виводить в термінал шлях (абсолютний шлях) до поточного робочого каталогу. |
| history | Команда виводить список останніх команд, які використовувались у поточній сесії терміналу або командного рядка. |
| echo | Команда використовується для виведення в термінал або командний рядок заданого текстового рядка або значення змінної. |
| echo $PATH | Команда "echo $PATH" виводить список шляхів, які містять виконувані файли, що можуть бути виконані з будь-якого місця в командному рядку. |
| which date | Команда "which" використовується для знаходження шляху до виконуваного файлу в операційній системі. Команда "which" приймає аргумент - ім'я виконуваного файлу та повертає повний шлях до цього файлу. |
| type command | Команда "type" використовується для визначення типу команди в залежності від того, як вона інтерпретується в системі. Функціональність команди "type" полягає в тому, що вона може вказати, як саме операційна система інтерпретує даний вхідний рядок (команду). |
| which ls | Команда "which" використовується для визначення шляху до виконуваного файлу для вказаної команди в операційній системі. Наприклад, виконання команди "which ls" може повернути "/bin/ls", що означає, що виконуваний файл для команди "ls" знаходиться в директорії "/bin". |
| type vi | Команда "type" використовується для визначення типу команди в оперативній системі. Виконання команди "type vi" покаже, що "vi" є вбудованою командою, яка є текстовим редактором. |
| echo Today is `date` | Команда "echo Today is date" виведе поточну дату та час в зручному для читання форматі. Команда "date" використовується для відображення поточної дати та часу в операційній системі, а команда "echo" використовується для виведення текстового повідомлення в терміналі. |
| echo Today is $(date) | Команда "echo Today is $(date)" виконає ту ж саму операцію, що і "echo Today is date". Використання "$(date)" замість "date" дозволяє виконувати команду "date" в дужках без використання символу апострофа. |
| echo This is the command | Команда "echo This is the command 'date'" виведе текстовий рядок, який містить команду "date" в апострофах. Команда "echo" використовується для виведення текстового повідомлення в терміналі. |
| echo This is the command \`date\` | Команда "echo This is the command `date`" також виведе текстовий рядок, який містить команду "date" в апострофах. |
| echo This is the command "`date`" | Така команда "echo This is the command "date"" також виведе текстовий рядок, який містить команду "date" в апострофах. Але в даному випадку для вставки команди використовуються подвійні лапки. |
| echo D\* | Команда "echo D\*" виведе на екран усі імена файлів, що починаються на літеру "D" в поточному каталозі (тобто в каталозі, в якому виконується команда). |
| echo "D\*" | Команда "echo" у терміналі Linux або подібній системі виводить передані їй аргументи на екран. У цьому випадку, коли ви використовуєте команду "echo" з аргументом "D\*", вона виведе на екран всі імена файлів в поточній директорії, що починаються на літеру "D". |
| man date | При виконанні команди "man date" в терміналі відкривається сторінка керівництва для команди "date", яка надає інформацію про те, як використовувати команду та її різні параметри та аргументи. |
| man -k password | Команда "man -k password" шукає в базі даних керівництва (man-сторінках) всі підказки (descriptions) що містять слово "password". Ця команда допомагає користувачеві знайти всі доступні інструменти, які пов'язані з темою "password". |
| apropos password | Ця команда допомагає користувачеві знайти всі доступні керівництва, які пов'язані з темою "password". Після виконання команди, виводиться список всіх знайдених керівництв, які містять слово "password". |
| man -f passwd | Команда "man -f passwd" призначена для пошуку короткої інформації про вказаний файл або команду в базі даних kерівництва (man-сторінках). |
| man 5 passwd | Щоб відобразити сторінку довідки для іншого розділу, укажіть номер розділу як перший аргумент команди man. |
| whatis passwd | Замість використання man -f для відображення всіх розділів сторінки довідки для імені, ви також можете використати команду whatis. |
| info date | Команда "info date" відкриває довідковий файл (info-сторінку) про команду "date" в системі Linux. |
| date --help | Команда "date --help" виконує запуск команди "date" з опцією "--help", що дозволяє вивести коротку довідку про команду "date" зі списком доступних опцій та їхніх призначень. |
| ls /usr/share/doc | Команда "ls /usr/share/doc" виконує виведення списку файлів та каталогів, які знаходяться у директорії "/usr/share/doc" в системі Linux. |
| locate crontab | Команда "locate crontab" виконує пошук файлів з назвою "crontab" у всіх доступних директоріях у системі, використовуючи базу даних імен файлів, яка зберігається на системі. |
| locate -b "\crontab" | Команда "locate -b '\crontab'" виконує той самий пошук файлів з назвою "crontab" у всіх доступних директоріях у системі, що й команда "locate crontab". Однак, вона додає опцію "-b", що дозволяє виконати точний пошук файлів з назвою "crontab", включаючи лише повні шляхи до файлів, які містять слово "crontab" як окреме слово. |
| whereis passwd | Команда "whereis passwd" використовується для пошуку бінарного файлу, файлів довідки та файлів джерела для програми "passwd" у стандартних місцях у системі Linux. |

***The material was prepared by student Kryvenko Andrew (AndrewKryvenko)***

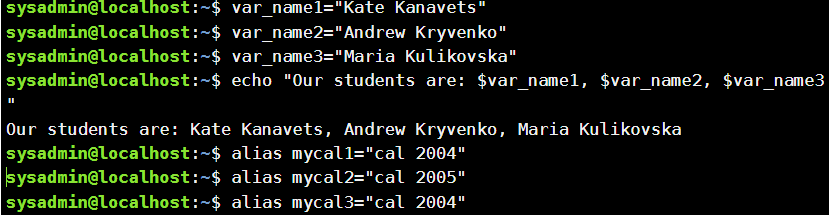
2. Робота в в терміналі (закріплення практичних навичок) обов’язково представити свої скріншоти:

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

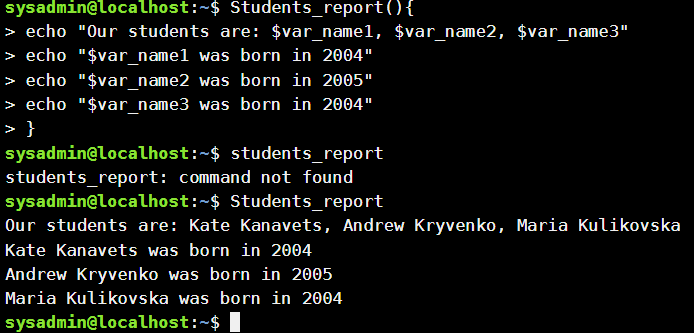
- Створіть псевдоніми mycal1, mycal2, mycal3 для команди cal для автоматичного виведення

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:

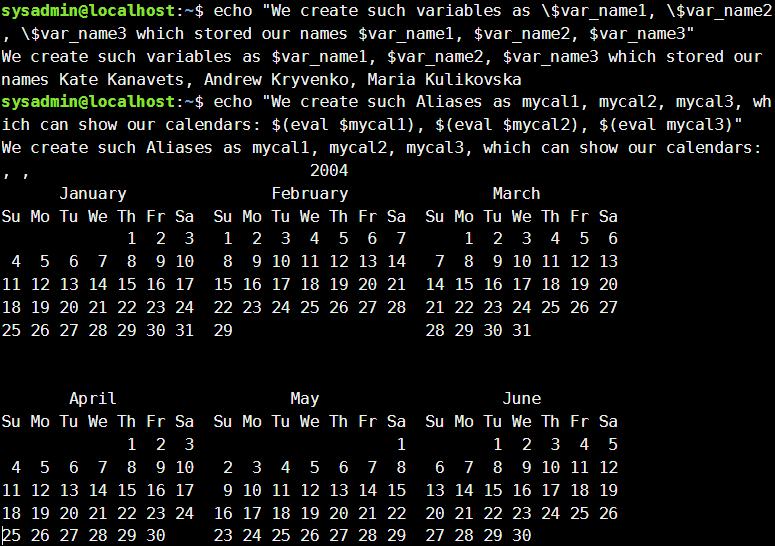
- “We create such variables as $var\_name1, $var\_name2, $var\_name3, which stored our names Name1,

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

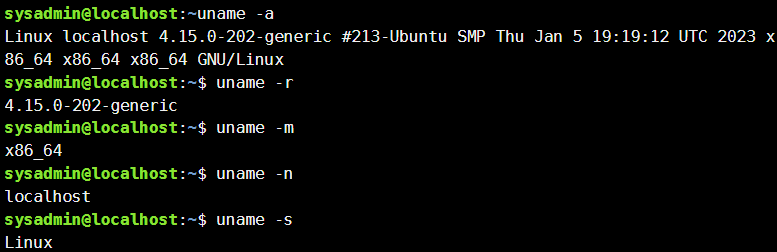
окремої функції, як це буде виглядати?

Ми можемо виконати завдання 2.1 та 2.2 за допомогою інструкцій керування в терміналі, але це буде складніше та менш зрозуміло, ніж використання функцій. Наприклад, ми можемо використовувати цикли та умовні оператори для виведення змінних та календарів у відповідному порядку, але це потребує більшої кількості коду.

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) 

***The material was prepared by student Kanavets Kateryna (kanavetsk)***

Контрольні запитання

1. Які типи команд існують в оболонці Bash?

В оболонці Bash існує декілька типів команд:

1)Вбудовані команди: це команди, які реалізовані безпосередньо в оболонці Bash. Вони виконуються швидше, ніж зовнішні команди, оскільки не потребують запуску нового процесу. До прикладу вбудовані команди: cd, echo, pwd, export, alias.

2)Зовнішні команди: це команди, які виконуються в окремому процесі, створеному оболонкою. Вони зазвичай розміщуються в директоріях, що містяться в змінній PATH, яка містить список шляхів до виконуваних файлів на вашій системі. До прикладу зовнішні команди: ls, grep, awk, sed, sort.

3)Функції: це блоки коду, які можуть бути викликані з командного рядка або з іншої функції в оболонці. Функції корисні для організації повторюваного коду та забезпечення його повторного використання.

4)Аліаси: це скорочення для команд або послідовності команд, які можуть бути викликані з командного рядка. Аліаси корисні для скорочення довгих чи складних команд та для збереження персоналізованих команд.

5)Спеціальні символи: це символи, які використовуються для керування поведінкою оболонки та виконання різних операцій в командному рядку. До прикладу спеціальні символи: >, <, |, &, ;.

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

Змінні оточення (environment variables) - це змінні, які містять інформацію про оточення, в якому запущений процес в операційній системі. Кожен процес має свій власний список змінних оточення, які можуть бути налаштовані для забезпечення різних функціональних можливостей та параметрів конфігурації.

Існує декілька типів змінних оточення:

Системні змінні оточення: це змінні, які встановлюються системою та відображають системні параметри. До прикладу, PATH зберігає список шляхів до виконуваних файлів в операційній системі.

Користувацькі змінні оточення: це змінні, які можуть бути встановлені користувачем або використовуватися в різних скриптах або програмах. До прикладу, MY\_VAR може бути використано для зберігання певних значень у програмі.

Щоб переглянути змінні оточення в терміналі, можна використати наступну команду:

printenv

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

$PS1 - це змінна оточення, яка відповідає за виведення первинного приглашення командного рядка (prompt) в терміналі. Вона містить спеціальні символи, які можна використовувати для виведення інформації про поточну робочу директорію, користувача, хост, поточну дату та іншу інформацію.

Значення змінної $PS1 містить форматування символів, що визначають вигляд та зміст первинного приглашення командного рядка. Зазвичай, значення цієї змінної складається з послідовності спеціальних символів, таких як \u, \h, \w та інших, які інтерпретуються терміналом та замінюються на потрібну інформацію при виведенні первинного приглашення командного рядка.

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 в Bash можна, встановивши нове значення цієї змінної. Щодо питання про те, що станеться з рядком запрошення після зміни змінної $PS1, слід зазначити, що він буде відображати нове значення змінної $PS1, якщо воно було змінено на щось інше.

Щоб змінити значення змінної $PS1 за замовчуванням, потрібно внести зміни до файлу конфігурації оболонки Bash. Зазвичай, це файл ~/.bashrc. Для зміни значення змінної $PS1 в цьому файлі потрібно додати рядок, що містить виконання команди зміни значення змінної $PS1

5. Для чого використовують лапки в оболонці Bash?

Лапки (одинарні або подвійні) в оболонці Bash використовуються для збереження рядків тексту як один елемент. Зокрема, використання лапок дозволяє:

Задати значення змінних, якщо в них містяться пробіли або спеціальні символи, такі як "\*", "?", "[", "{", "}", "$" і т.д.

Задати аргументи для команд, якщо вони містять пробіли або спеціальні символи.

Використовувати спеціальні символи, які мають особливе значення в Bash, як звичайні символи.

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

Інструкції керування використовуються для управління поведінкою оболонки та виконання команд в залежності від певних умов. Основні види інструкцій керування в оболонці Bash наступні:

Умовні інструкції:

if-then-else: використовується для перевірки умови, якщо умова істинна, то виконується перший блок команд, якщо умова неправдива - другий блок команд.

case: використовується для заміни послідовної послідовності інструкцій if-else. Він перевіряє значення змінної і виконує відповідну гілку команд.

Циклічні інструкції:

for:використовується для виконання певного набору команд для кожного елементу списку.

while: використовується для виконання команд до тих пір, поки умова залишається істинною.

until: використовується для виконання команд, поки умова не стає істинною.

Інші інструкції:

break: використовується для виходу з циклу.

continue: використовується для переходу до наступної ітерації циклу.

exit: використовується для завершення роботи програми.

function: використовується для визначення функцій.

source: використовується для читання та виконання команд з файлу.

Кожен тип інструкції керування має свій синтаксис та особливості використання.

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

Символ $ або # в кінці рядку запрошення (PS1) в bash показує, яким користувачем виконується командна оболонка. Зазвичай символ $ вказує, що оболонка виконується з правами звичайного користувача, а символ # - з правами суперкористувача (root).

Отже, якщо в кінці рядку запрошення стоїть символ $, це означає, що оболонка виконується з правами звичайного користувача. Якщо ж в кінці рядку запрошення стоїть символ #, це означає, що оболонка виконується з правами суперкористувача (root).

Зміна символу в кінці рядку запрошення з # на $ може вказати на зміну прав користувача, з яким виконується оболонка. Наприклад, при зміні користувача зі звичайного користувача на суперкористувача (root), символ # може змінитися на символ $.

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

Команди whereis та locate використовуються в Unix-подібних операційних системах для пошуку файлів у системі.

whereis шукає виконувані файли, об'єктні файли та файли довідки для заданої команди, вказаної як аргумент. Вона повертає список шляхів до цих файлів.

locate шукає файл у базі даних, яка містить імена всіх файлів у системі. Вона повертає список файлів, що відповідають заданому шаблону.

Основна відмінність між whereis та locate полягає в тому, що whereis шукає файл за його типом, тоді як locate шукає файл за іменем. Також, locate працює швидше, оскільки вона шукає в базі даних, тоді як whereis шукає по всій файловій системі. Однак, оновлення бази даних для locate може займати деякий час, тоді як whereis завжди поверне актуальний результат.