“Київський фаховий коледж зв’язку”

Циклова комісія комп’ютерної інженерії

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №3**

з дисципліни “Операційні системи”:

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

Виконали студенти

групи РПЗ-13А та РПЗ-13Б

Eleven Two Zeroes:

Vlad Sapozhnyk

Max Karpenko

Dmytro Onufriiev

Київ 2024

Робота студентів групи РПЗ-13А та РПЗ-13Б Eleven Two Zeroes: Vlad Sapozhnyk, Max Karpenko and Dmytro Onufriiev.

**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 – VirtualBox (Oracle).
4. Операційна система GNU/Linux – CentOS.

**Завдання для попередньої підготовки: *Created by Dmytro Onufriiev***  
**Хід роботи. *Created by Vlad Sapozhnyk***

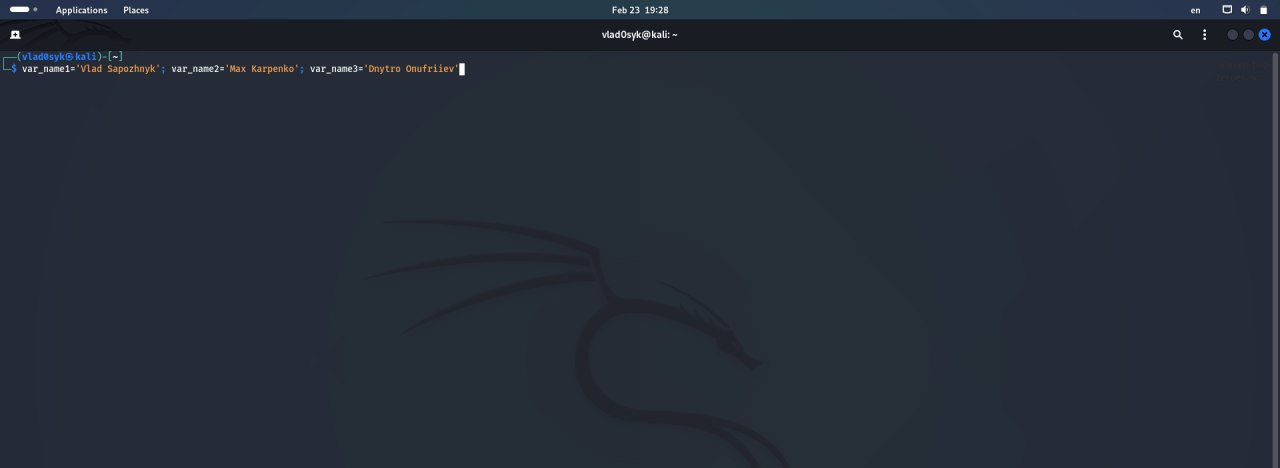
1. Опрацюйте всі приклади команд, що представлені у лабораторній роботі курсу NDG Linux Essentials - Lab 5: Command Line Skills та Lab 6: Getting Help. Створіть таблицю для опису цих команд:

| Назва команди | Її призначення та незалежність |
| --- | --- |
| ls | Displays information about directories and files. By default, it displays information for the current directory without arguments. |
| ls -l | Using the -l option in the ls command allows you to display information about files located in the current working directory in a long format that provides more extensive additional information. |
| ls -l /home | Using the /home argument in combination with the -l option in the ls command command allows you to display detailed information about the files in the /home directory. |
| whoami | This command displays the same information that you see in the first part of the prompt.  The output of the whoami command displays the user name of the current user. Although in this case your username is displayed in the prompt, this command could be used to obtain this information in a situation when the prompt did not contain this information. |
| uname | This command displays information about the current system. To be able to see the name of the kernel you are using, type this command into the terminal. |
| uname -n | uname --nodename | The uname -n and uname --nodename commands display the hostname of the system. This is usually the name of a computer or server, which is determined when the operating system is configured. |
| pwd | The pwd command is used to display your current "location" or current "working" directory. |
| history | One way to use command history is to run the history command to view a numbered list of history. The number to the left of the command can be used to re-run the command. The history command also has a number of parameters and arguments that can be used to specify which commands are saved or displayed. |
| date | The date command prints the time and date on the system. |
| clear | The clear command clears the screen. |
| history 5 | To view a limited number of commands, the history command can take a number as a parameter to display exactly that many recent entries. Type the following command to display the last five commands from your history. |
| !9 | To execute a command again, type the exclamation point and the history list number. For example, to execute the 9th command in your history list, you would execute the following. |
| echo | The echo command can be used to print text and the value of a variable, and to show how the shell environment expands metacharacters. |
| echo $HISTSIZE | In the example above, the command will show the value of the HISTSIZE variable. |
| echo $PATH | This command displays the value of the PATH variable. |
| which date | Use the which command to determine if there is an executable file, in this case named date, that is located within a directory listed in the PATH value. |
| type <command> | The type command can be used to determine information about command type. |
| type -a <command> | Using the -a option of the type command displays all locations that contain the command. |
| alias | To determine what aliases are set on the current shell, use the alias command. |
| echo Today is `date` | This command writes the sentence "Today is" and adds the date that the date command writes. |
| echo Today is $(date) | This command will write the same as the previous command. |
| echo This is the command ‘`date`’ | This command will write the sentence "This is the command `date`". |
| echo This is the command \`date\` | This command will write the same as the previous command. |
| echo This is the command “`date`” | This command writes the sentence "This is the command " and adds the date that the date command writes. |
| echo D\* | This command will write all directories which are starting with the symbol ‘D’. |
| echo “D\*” | This command will write “D\*”. |
| echo Hello; echo Linux; echo Student | These commands will write words “Hello”, “Linux” and “Student” in a column. |
| false; echo Not; echo Conditional | This commands will write words “Not” and “Conditional ” in a column. |
| echo Start && echo Going && echo Gone | These commands will write words “Start”, “Going” and “Gone” in a column. |
| echo Success && false && echo Bye | These commands will write the word “Success”. |
| false || echo Fail Or  true || echo Nothing to see here | The first command will print the sentence "Fail Or", and the other will print nothing. |
| man date | With the help of the man command, we can learn more about the command which will be after command man. In our case, the terminal will display detailed information about the use of the date command. |
| man -k password | In some cases you may not remember the exact name of the command. In these cases you can use the -k option to the man command and provide a keyword argument. For example, execute the following command to display a summary of all man pages that have the keyword "password" in the description. |
| apropos password | The apropos command is another way of viewing man page summaries with a keyword. |
| man -f passwd | There are often multiple man pages with the same name. For example, the following command shows three pages for passwd. |
| man 5 passwd | To display a man page for a different section, provide the section number as the first argument to the man command. |

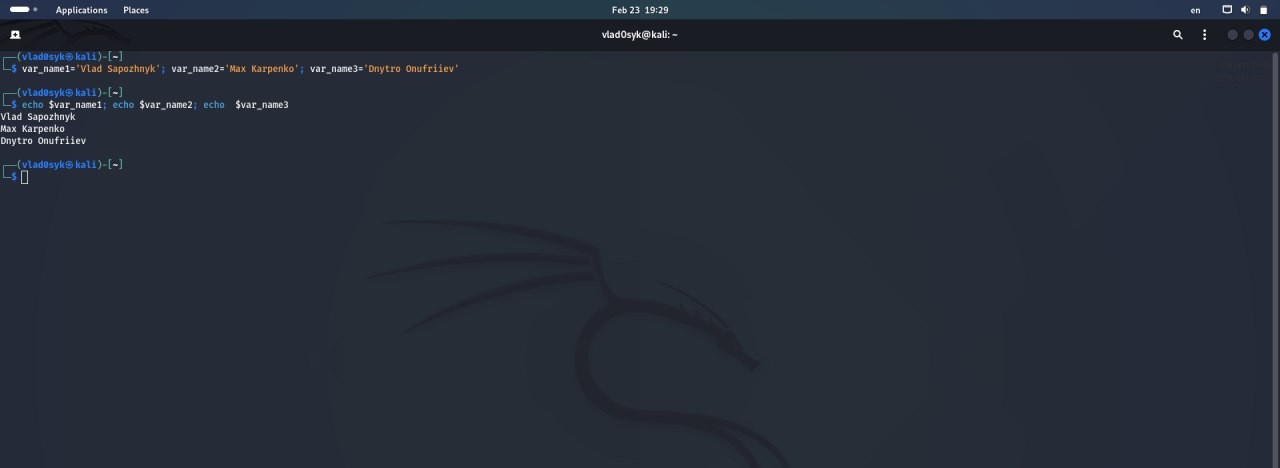
2. Робота в терміналі (закріплення практичних навичок) обов’язково представити свої скріншоти:

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

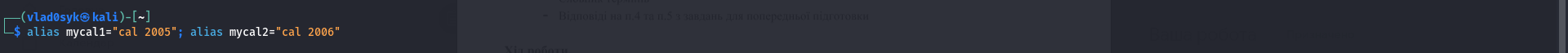


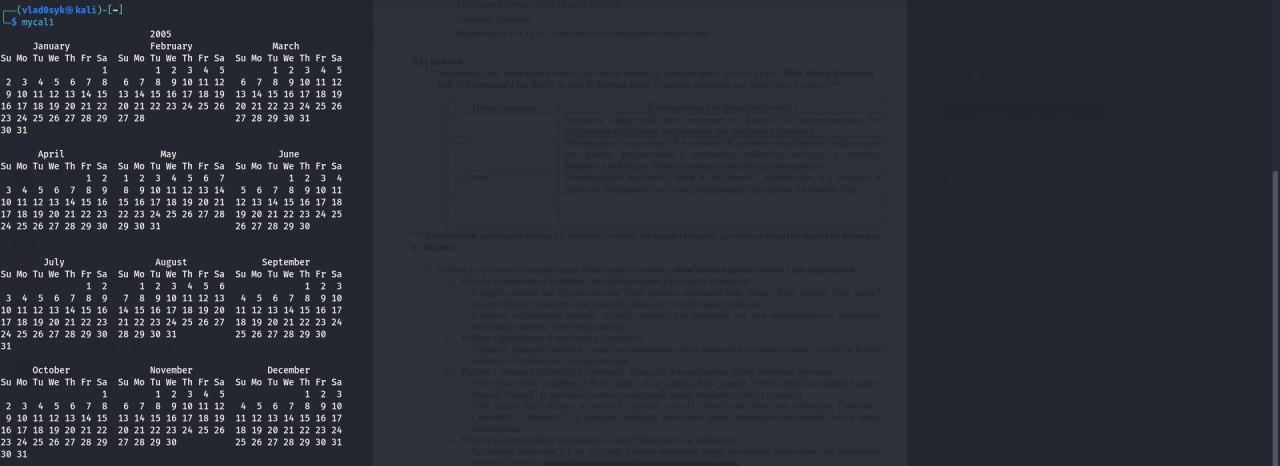
- За допомогою команди echo виведіть імена студентів вашої команди

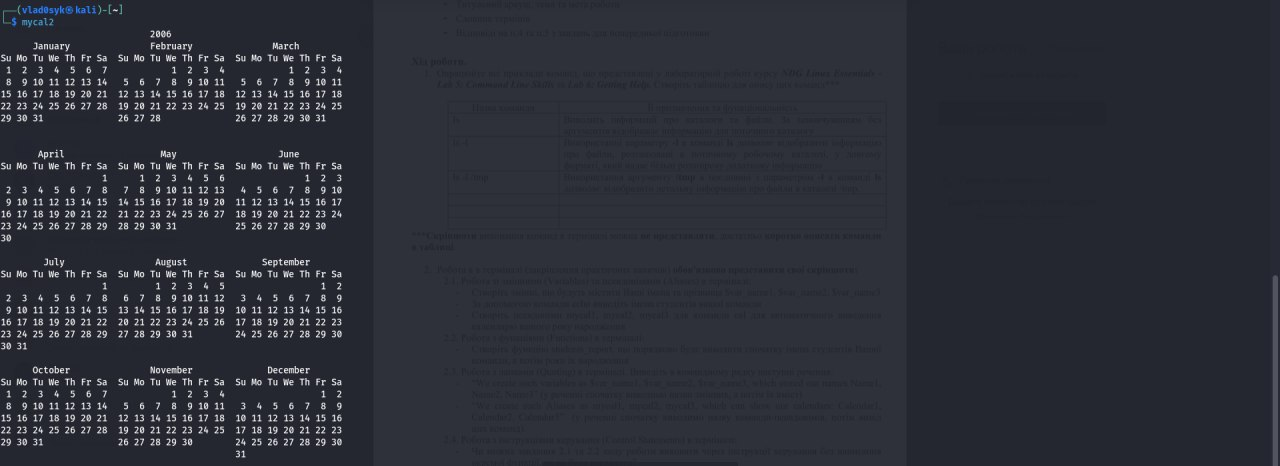


- Створіть псевдоніми mycal1, mycal2, mycal3 для команди cal для автоматичного виведення

календарю вашого року народження

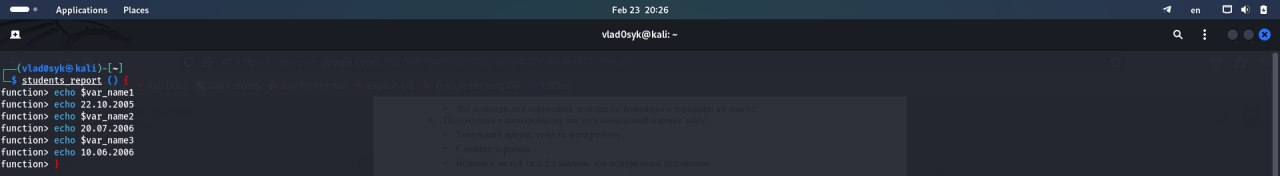


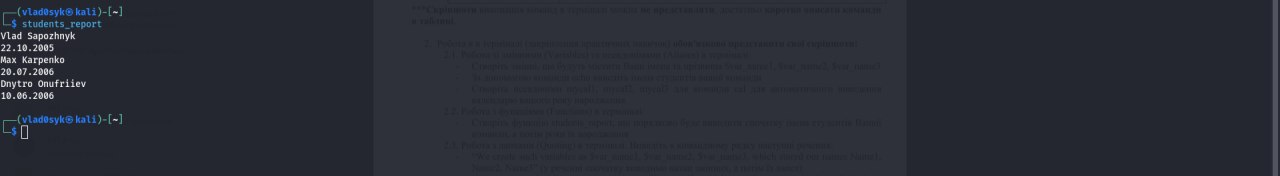




2.2. Робота з функціями (Functions) в терміналі:

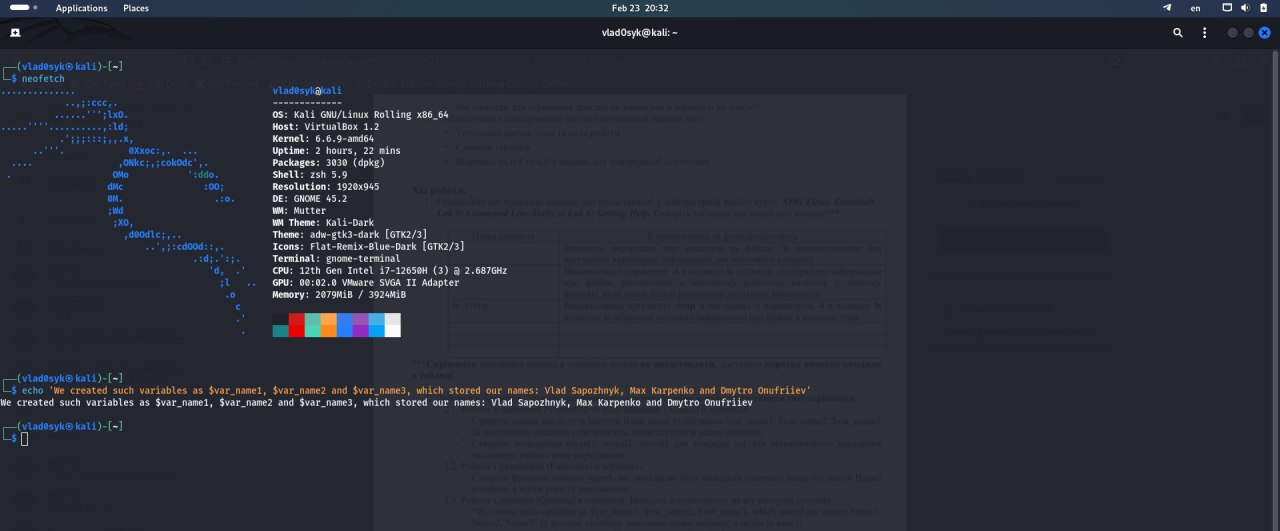
- Створіть функцію students\_report, що порядково буде виводити спочатку імена студентів Вашої команди, а потім роки їх народження





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).

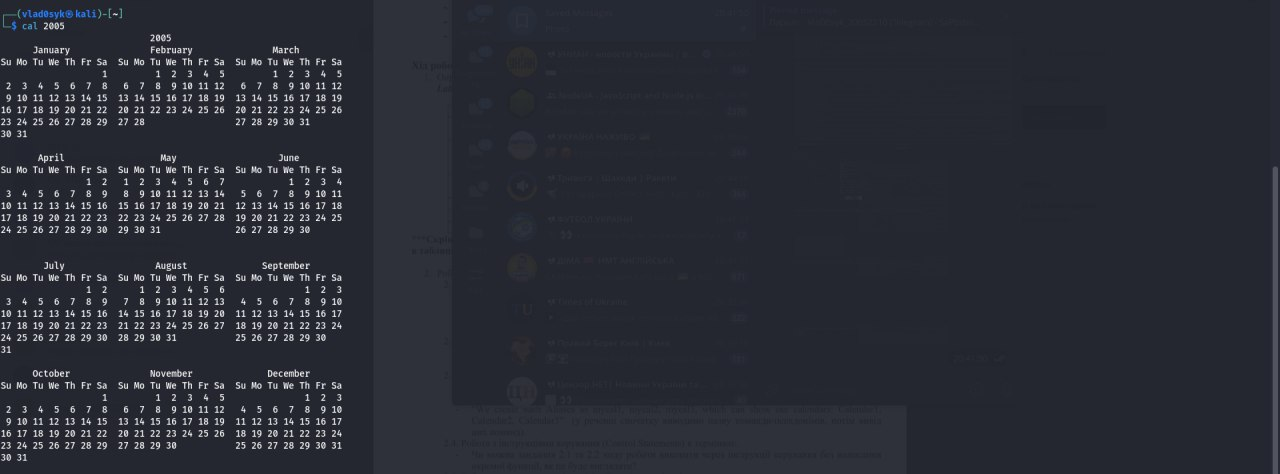


2.4. Робота з інструкціями керування (Control Statements) в терміналі:.

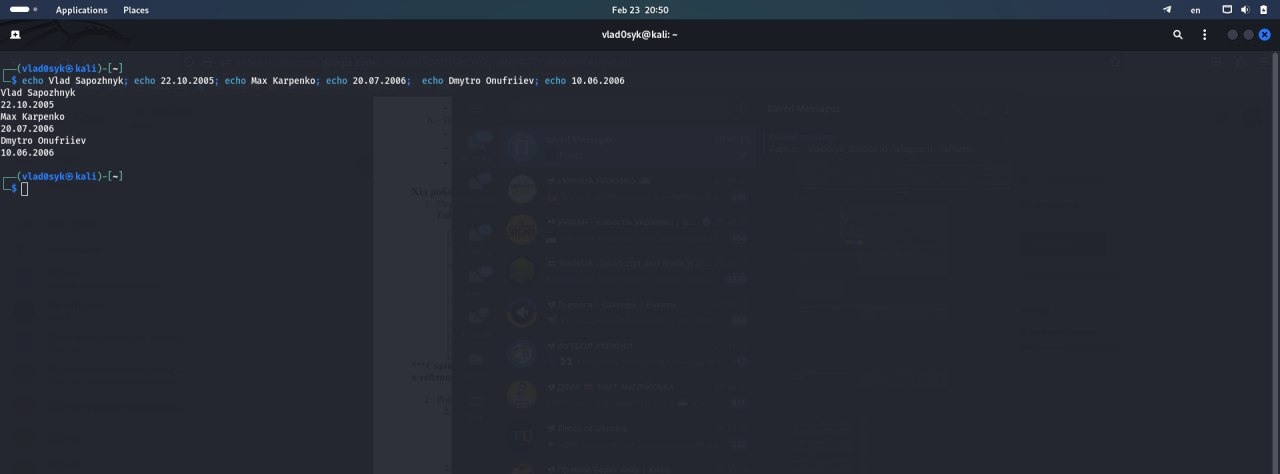
- Чи можна завдання 2.1 та 2.2 ходу роботи виконати через інструкції керування без написання окремої функції, як це буде виглядати?

Так, можна в пункті 2.1 не створювати змінні а просто через команду echo і ‘;’ вивести імена та прізвища студентів. Також можна було не створювати псевдоніми і також через команду cal вивести календар будь-якого року. Це буде виглядати таким чином:





В пункті 2.2 можна було не створювати функцію, а окремо (так як і в пункті 2.1) вивести всю інформацію через команду echo та ‘;’. Ось так це буде виглядати:



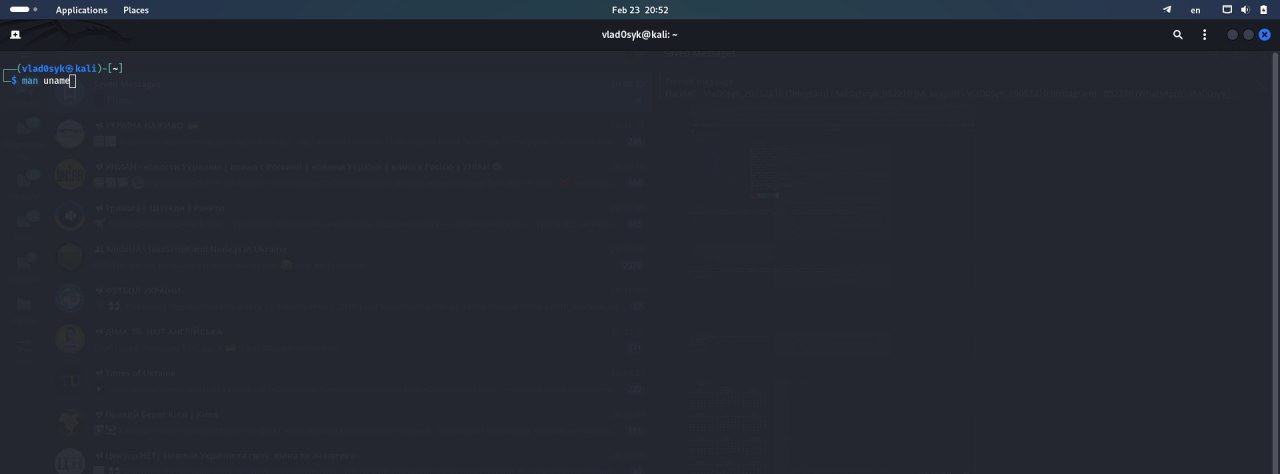
2.5. Робота з командами довідки (Man Pages) в терміналі:.

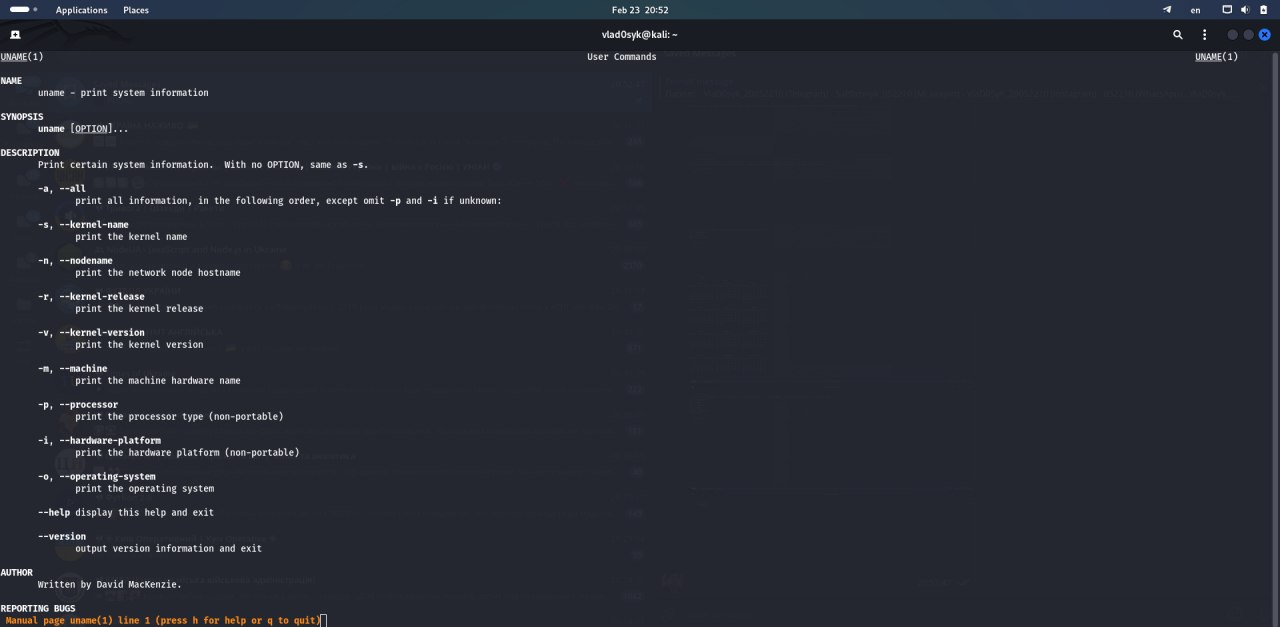
- На прикладі команди uname продемонструйте як отримати довідку. На основі отриманої

додаткової інформації наведіть 5 різних варіантів виводу результату інформації по даній

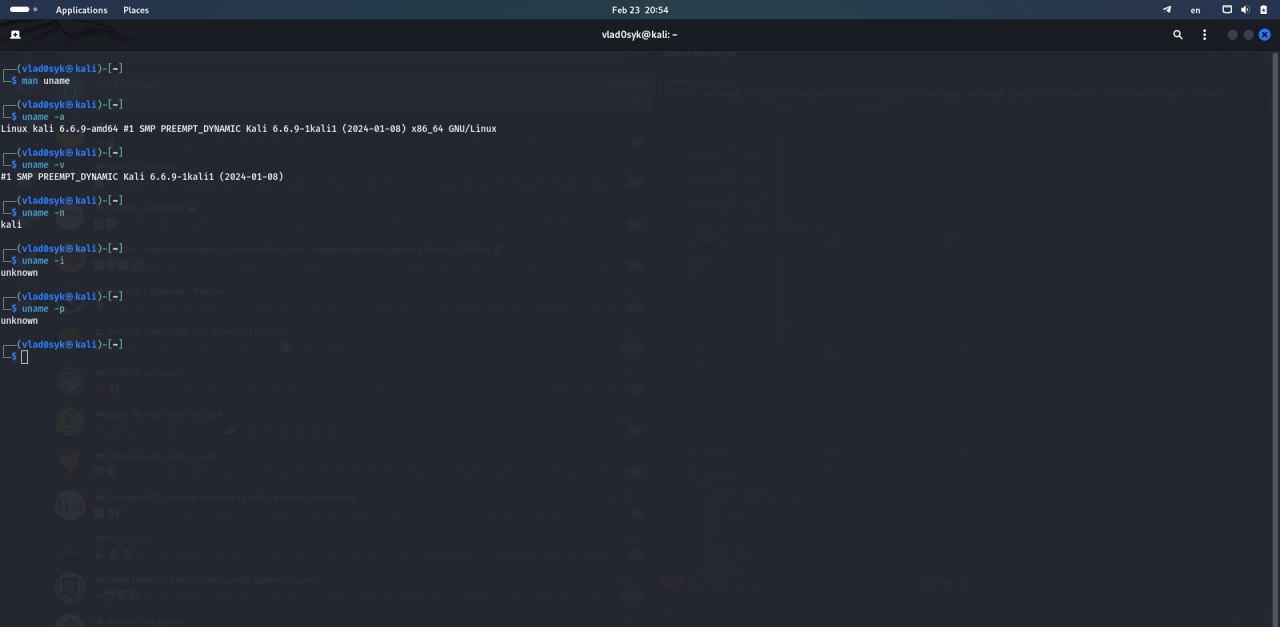
команді з використанням 5 різних параметрів (Options)

Щоб отримати довідку на прикладі команди uname треба написати ось таку команду:





Ось 5 варіантів виводу:



**Контрольнi запитання: *Created by Max Karpenko***

1. Які типи команд існують в оболонці Bash?

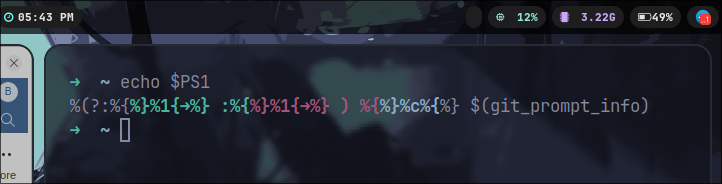
* **Вбудовані команди оболонки (Builtin commands)**: Це команди, які виконуються безпосередньо оболонкою Bash і не вимагають виклику зовнішніх виконуваних файлів. Вони зазвичай забезпечують функціональність, необхідну для управління робочим середовищем оболонки, наприклад, cd для зміни директорії, export для встановлення значень змінних середовища тощо.
* **Зовнішні команди**: Це програми, які зберігаються де-інде на файловій системі і викликаються оболонкою Bash для виконання. Ці команди не вбудовані в оболонку і можуть бути частиною операційної системи або встановлені користувачем. Приклади зовнішніх команд включають більшість утиліт Unix/Linux, таких як ls, grep, awk тощо.
* **Аліаси (Aliases)**: Аліаси дозволяють визначати короткі псевдоніми для довгих команд або послідовностей команд, що полегшує їх введення та запам'ятовування. Наприклад, користувач може створити аліас ll як короткий спосіб виклику ls -l.
* **Функції оболонки**: Функції в Bash дозволяють групувати кілька команд в один блок, який можна викликати за допомогою одного імені функції. Це спосіб створення складних скриптів та процедур в оболонці, що можна використовувати для автоматизації завдань.
* **Ключові слова (Keywords)**: Це слова, які мають спеціальне значення в Bash. Вони використовуються для управління логікою та потоком виконання скриптів. Приклади ключових слів включають if, else, fi, for, while, do, done тощо.

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

Змінні оточення - це змінні, які визначають налаштування та параметри робочого середовища операційної системи та програм. Для перегляду змінних оточення в терміналі можна використати команду printenv або env.

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

$PS1 - це змінна оточення, яка визначає вигляд командного рядка (prompt) в Bash. Для перегляду її вмісту в терміналі використовуйте команду echo $PS1.



1. 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?

Для зміни $PS1 використайте команду export PS1='нове\_значення'. Це змінить вигляд рядка запрошення в Bash на поточний сеанс. Щоб змінити значення не на поточний сеанс, а за замовчуванням, додайте таку ж команду до файлу **~/.bashrc** або **~/.profile** та перезавантажте термінал або виконайте source **~/.bashrc**.

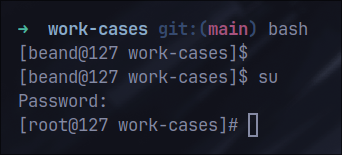
1. Для чого використовують лапки в оболонці Bash?

* Одинарні лапки (' ') зберігають буквальне значення всіх символів в середині.
* Подвійні лапки (" ") дозволяють інтерпретувати спеціальні символи (наприклад, $, `) в середині.
* Зворотні лапки (` `) використовують для виконання команд і підстановки результату в місце їх використання.

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

* Умовні оператори (**if, else, elif, case**)
* Цикли (**for, while, until**)
* Керування вибором (**select**)
* Розгалуження (**break, continue**)
* Повернення значення (**exit, return**)

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



1. Символ **$** в кінці рядка запрошення означає, що ви залогінені як звичайний користувач.
2. Символ **#** означає, що ви залогінені як користувач root (суперкористувач).
3. Яке призначення команд whereis та locate? Яка між ними відмінність?

**whereis** шукає бінарні файли, вихідні коди та сторінки довідника для програми. locate шукає файли, використовуючи базу даних, створену командою **updatedb**.

Основна відмінність: **whereis** зосереджений на програмному забезпеченні, тоді як **locate** шукає всі файли в системі.

**Висновок:** We got acquainted with the basic commands of the CLI mode in Linux. We also practiced them using the Kali distribution. We understood the types of variables in the terminal and how to declare them and display their values. We considered the creation of functions and aliases. We understood the difference between using three types of quotes.