“Київський фаховий коледж зв’язку”

Циклова комісія Комп’ютерної інженерії

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №6**

з дисципліни: «Операційні системи»

**Тема: «Робота з текстом в CLI-режимі Linux та створення скриптових сценаріїв»**

Виконали:

студенти групи РПЗ-93б

**Команда 1**: Усенко Б.О.,

Мельнічук М.О

Перевірив викладач

Сушанова В.С.

Київ 2022

**Мета роботи:**

1. Отримання практичних навиків роботи з командною оболонкою Bash.
2. Знайомство з базовими діями при роботі з текстом у терміналі.
3. Знайомство з базовими діями при роботі зі скриптовими сценаріями.

**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.

5. Сайт мережевої академії Cisco netacad.com та його онлайн курси по Linux

**Завдання для попередньої підготовки.**

***Готувала матеріал студентка Мельнічук М.О.***

1. Прочитайте короткі теоретичні відомості до лабораторної роботи та зробіть невеличкий словник базових англійських термінів з питань призначення команд та їх параметрів.

|  |  |
| --- | --- |
| **Термін** | **Переклад** |
| head and tail commands  head /etc/sysctl.conf | are used to display only the first few or last few lines of a file |
| pipe | character | вертикальна| лінія  can be used to send the output of one command to another. |
| Input/Output Redirection | Перенаправлення введення/виведення |
| tr command | to capitalize a line of text |
| sort file.txt | to rearrange the lines of files or input in either dictionary or numeric order |
| The asterisk \* character | Символ зірочки \* |
| Filter file sections | Фільтр розділу файлів |
| Regular expression character | Символ регулярного виразу |
| shell script | Скрипт оболонки  a file of executable commands that has been stored in a text file |

1. На базі розглянутого матеріалу дайте відповіді на наступні питання:
   1. Яке призначення команд *cat, less, more, head and tail*?

|  |  |  |
| --- | --- | --- |
| **Команда** | **Опис** | **Приклад** |
| *cat* | Команда cat, скорочення від concatenate, є командою, функції якої включають створення та відображення текстових файлів, а також об’єднання копій текстових файлів. |  |
| *more* | Команда *more*, призначена для посторінкового перегляду великих текстових файлів. |  |
| *less* | Команда *less* дозволяє перемотувати текст не тільки вперед, але й назад, здійснювати пошук в обох напрямках, переходити відразу в кінець або початок файлу. |  |
| *head* | Команда *head* виводить початкові рядки (за замовчуванням – 10) з одного або кількох документів. |  |
| *tail* | Команда *tail* виводить останні рядки (за замовчуванням – 10) з одного або кількох документів. |  |

* 1. Командна оболонка в [UNIX](https://uk.wikipedia.org/wiki/UNIX) є [інтерфейсом командного рядка](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D1%80%D1%8F%D0%B4%D0%BA%D0%B0) в [Unix](https://uk.wikipedia.org/wiki/Unix)-подібних [операційних системах](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), тобто виконує команди, які подає користувач, або які читаються з [файлів](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB). Такі файли з командами оболонки називаються [*сценаріями*](https://uk.wikipedia.org/wiki/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D1%96%D0%B9) (скриптами, програмами) *оболонки*. Ці сценарії не [компілюються](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80), а [інтерпретуються](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80) оболонкою. Це означає, що оболонка прочитує сценарій *від початку до кінця*, рядок за рядком, шукаючи зазначені там команди й виконуючи їх; на відміну від цього підходу, [компілятор](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) перетворює цілу програму до вигляду, придатного до виконання машиною — потім файл з таким кодом можна використати в сценарії оболонки.

Одним з завдань оболонки є забезпечення користувацького середовища, яке можна налаштувати за допомогою конфігураційних файлів.

* 1. Призначення команди *grep*:

Команду *grep* можна використовувати для фільтрації рядків у файлі або виводу іншої команди, яка відповідає заданому шаблону. Цей шаблон може бути таким же простим, як точний текст, який ви хочете знайти, або може бути набагато розширеним за допомогою використання регулярних виразів.

1. Вивчіть матеріали онлайн-курсів академії Cisco:

* NDG Linux Essentials (Chapter 10-12 all Topics)

1. На базі розглянутого матеріалу у онлайн курсах дайте відповіді на наступні питання:

4.1 Охарактеризуйте поняття скриптового сценарію.

**Сценарій** (скрипт) — це програма, яка автоматизує деяке завдання, яке без **сценарію** користувач робив би вручну, використовуючи інтерфейс програми.

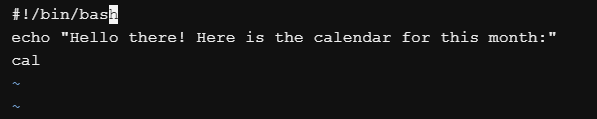
*Скрисповий сценарій у командній оболонці* — це файл виконуваних команд, який зберігається в текстовому файлі. Під час запуску файлу виконується кожна команда.  Ви можете створювати сценарії для автоматизації повторюваних частин вашої роботи, що звільняє ваш час і забезпечує послідовність кожного разу, коли ви використовуєте сценарій. Наприклад, якщо ви запускаєте одні й ті самі п’ять команд щодня, ви можете перетворити їх на сценарій оболонки, який зводить вашу роботу до однієї команди.

4.2 Яким чином створюються та редагуються скрипти, що треба зробити щоб запустити скрипт?

В курсі Linux Essentials згадуються 2 редактори: редактор GNU **nano** — це дуже простий редактор, який добре підходить для редагування невеликих текстових файлів та візуальний редактор **vi**.

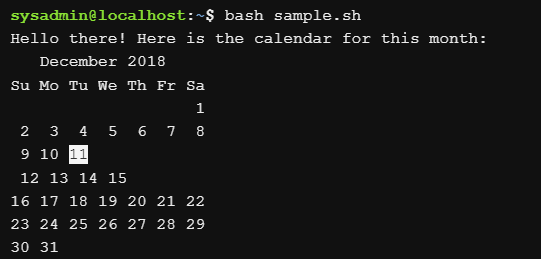
Щоб створити простий сценарій оболонки, вам просто потрібно створити текстовий файл і додати команди. Створіть файл з назвою sample.sh і додайте такі рядки(також можна виправити ті рядки, які потребують редагування)(**Рисунок 1**):





**Рисунок 1** – Створення файлу та додавання команд

Один із способів запустити цю програму – ввести **bash** перед іменем файлу. Наприклад(**Рисунок 2**):



**Рисунок 2** – Запуск скрипту

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

* Chapter 10 Exam
* Chapter 11 Exam
* Chapter 12 Exam

**Хід роботи.**

***Готувала матеріал студентка Усенко Б.О.***

* 1. Початкова робота в CLI-режимі в Linux ОС сімейства Linux:
  2. Запустіть віртуальну машину Ubuntu\_PC ***(якщо виконуєте завдання ЛР через академію netacad)***
  3. Опрацюйте всі приклади команд, що представлені у лабораторних роботах курсу ***NDG Linux Essentials:***
* ***Lab 10: Working With Text***
* ***Lab 11: Basic Scripting***
* ***Lab 12: Understanding Computer Hardware***
  1. Створіть таблицю команд вивчених у п.2 ходу роботи у наступному вигляді:

|  |  |
| --- | --- |
| Назва команди | Її призначення та функціональність |
| cat | can be used to display file contents |
| find | good command to demonstrate how stderr works. This very flexible command allows searching with a host of options such as filename, size, date, type and permission.  The find command will begin the search in the directory specified and recursively search all of the subdirectories |
| tr | command translates characters, but it only accepts data from stdin, never from a file name given as an argument. The tr command accepts keyboard input (stdin), translates the characters and then redirects the output to stdout |
| more | The more command is intended for page-by-page viewing of large text files |
| lscpu | determine the type of CPU |
| free -m  free -g | discover how much RAM and swap space is being used  The output shows the amount of memory in megabytes when the -m option is used and in gigabytes when the -g option is used: |
| lspci | to see what devices are connected to the PCI bus  Use the lspci command with the -k option to show devices along with the kernel driver and modules used |
| lsusb | list the USB connected devices |
| lsmod | to view the currently loaded modules |

* 1. Створіть скриптові сценарії з виводом текстових повідомлень для користувача:
* сценарій має виводити привітання до поточного користувача вказуючи поточну дату та інформацію про поточну систему;
* сценарій має виводити інформацію про апаратну конфігурацію поточної системи (використовуйте команди розглянуті в Lab 12).

**Відповіді на контрольні запитання:**

***Готували матеріал студентки Мельнічук М.О, Усенко Б.О.***

1. Яким чином в командному інтерпретаторі можна перенаправляти потоки? Продемонструйте приклади, коли перенаправляється ввід / вивід / повідомлення про помилки.

1 команда > файл Направляє стандартний потік виводу в новий файл

2 команда 1> файл Направляє стандартний потік виводу у вказаний файл

3 команда >> файл Направляє стандартний потік виводу у вказаний файл (режим приєднання)

4 команда > файл 2>&1 Направляє стандартні потоки виводу і помилок у вказаний файл

5 команда 2> файл Направляє стандартний потік помилок у вказаний файл

6 команда 2>> файл Направляє стандартний потік помилок у вказаний файл (режим приєднання)

7 команда >> файл 2>&1 Направляє стандартні потоки виводу і помилок у вказаний файл (режим приєднання)

8 команда < файл1 > файл2 Отримує вхідні дані з першого файлу і направляє вихідні дані у другий файл

9 команда < файл в якості стандартного вхідного потоку отримує дані з вказаного файлу

10 команда << розділювач Отримує дані зі стандартного потоку вводу до тих пір, поки не зустрінеться розділювач

11 команда <&m В якості стандартного вхідного потоку отримує дані з файлу з дескриптором m

12 команда >&m Направляє стандартний потік виводу в файл з дескриптором m п Оператор n>&m дозволяє перенаправляти файл з дескриптором n туди, куди спрямований файл з дескриптором m. пПодобных операторів в командному рядку може бути декілька, в цьому випадку вони обчислюються зліва направо.

*У Linux існує три стандартних потоки введення / виводу даних.*

*Перший* - це стандартний потік введення (standard input). В системі це - потік №0 (так як в комп'ютерах рахунок зазвичай починається з нуля). Номери потоків ще називають дескрипторами. Цей потік являє собою якусь інформацію, передану в термінал, зокрема - інструкції, передані в оболонку для виконання. Зазвичай дані в цей потік потрапляють в ході введення їх користувачем з клавіатури.

*Другий потік* - це стандартний потік виводу (standard output), йому присвоєно номер 1. Це потік даних, які оболонка виводить після виконання якихось дій. Зазвичай ці дані потрапляють в той же вікно терміналу, де була введена команда, яка викликала їх появу.

І, нарешті, *третій потік* - це стандартний потік помилок (standard error), він має дескриптор 2. Цей потік схожий на стандартний потік виведення, так як зазвичай те, що в нього потрапляє, виявляється на екрані терміналу. Однак, він, за своєю суттю, відрізняється від стандартного виводу, як результат, цими потоками, при бажанні, можна управляти окремо. Це корисно, наприклад, у такій ситуації. Є команда, яка обробляє великий обсяг даних, виконуючи складну і піддану помилок операцію. Потрібно, щоб корисні дані, які генерує ця команда, не змішувалися з повідомленнями про помилки. Реалізується це завдяки роздільному перенаправлення потоків виведення і помилок.

1. *Командні фільтри* — це спосіб перегляду певних записів у формі, звіті, запиті або таблиці даних.

Застосування фільтра дає змогу обмежити кількість даних для відображення без змінення оформлення основного об'єкта.

1. */dev/null* - це файл віртуального пристрою, який при записі до нього все відправляє в порожнечу, а при читанні з нього - зчитує нуль. Найчастіше перенаправлення в / dev / null використовується для придушення стандартного виводу (вихідного потоку) (таке придушення найчастіше використовується в командних сценаріях (shell scripts) для придушення небажаного виведення на консоль).

**Висновок**: В ході виконання лабораторної роботи нами було отримано практичні навики роботи з командною оболонкою Bash, ми ознайомилися з базовими діями при роботі з текстом у терміналі та при роботі зі скриптовими сценаріями.