Ulchich

Відповідно до індивідуального варіанту, охарактеризуйте чотири команди командного рядка Linux і наведіть приклади їх використання. Охарактеризувати словами, що вона робить, навести приклади використання.

`u — userdel`
`l — ls`
`c — cat`
`h — head`

Команда `userdel`

Використання:

`userdel [OPTIONS] LOGIN`

Команда `userdel` змінює системні файли облікових записів, видаляючи усі записи, які стосуються користувача з іменем `LOGIN`. Користувач із заданим іменем повинен існувати.

Ключі:

`-f` — видаляє користувача, навіть якщо він все ще залогінений в системі, разом з його домашньою директорією.

`-r` — видаляє домашню директорію користувача та `spool` пошти.

`-Z` — видаляє користувача SELinux, що відповідає заданому користувачу.

Наприклад, щоб видалити користувача `jack` разом із вмістом його домашньої директорії та пошти, виконується така команда:

```
$ userdel -r jack
```

Щоб видалити цього ж користувача разом з відповідним користувачем SELinux, домашньою директорією та навіть якщо він зараз залогінений в системі:

```
$ userdel -fZ jack
```

Команда `ls`

Використання:

`ls [OPTION]... [FILE]...`

Команда `ls` виводить інформацію про файли (за замовчуванням — про поточну директорію).

Найважливіші ключі:

`-a` — виводити записи, що починаються з `.` (приховані файли).

`-l` — виводити записи у довгій, більш детальній формі.

`--sort=WORD` — сортування, де `WORD = [none|extension|size|time|version]`. Методи сортування:

1. `none` — не сортувати.
2. `extension` — за розширенням.
3. `size` — за розміром файлу.
4. `time` — за часом зміни.
5. `version` — за версією.

Наприклад, вивести зміст поточної директорії:

```
$ ls
```

Вивести вміст підпапки «testdir» домашньої директорії у довгій формі, сортуючи за розміром файлу:

```
$ ls -l --sort=size ~/testdir
```

Команда `cat`

Використання:

`cat [OPTION]... [FILE]...`

Команда `cat` конкатенує (додає один файл в кінець одного) та виводить результат до стандартого виводу.

Найважливіші ключі:

`-v` — виводити недруковані символи ASCII.

`-A` — виводити усі недруковані знаки, еквівалентно `-vET`.

`-E` — виводити знак $ в кінці кожного рядка.

`-T` — виводити знаки табуляції як `^I`.

Команду `cat` можна використовувати, Щоб вивести зміст файлу:

```
$ cat file.txt
```

Щоб об'єднати файли `file1.txt` і `file2.txt` та перенаправити (>, зберегти) їх у `res.txt`:

```
$ cat file1.txt file2.txt > res.txt
```

Команда head

Використання:

`head [OPTION]... [FILE]...`

Команда `cat` виводить перші 10 рядків файлу до стандартого виводу. Якщо файлів декілька, перед кожним файлом вона виводить його назву.

Найважливіші ключі:

`-c=[-]K` — вивести *K* перших байт кожного файлу. Якщо у параметрах задані «-», команда виводить усі байти, крім *K* останніх.

`-n=[-]K` — вивести *K* перших рядків кожного файлу. Якщо у параметрах задані «-», команда виводить усі рядки, крім *K* останніх.

Щоб вивести усі рядки файлів `f1.txt`, `f2.txt`, крім 5 останніх, команду `head` використовують так:

```
$ head -n=-5 file.txt f1.txt f2.txt
```

Значення *K* може мати суфікси, наприклад: `b(=512)`, `kB(=1000)`, `K(=1024)`, `MB(=1000*1000)`, `M(=1024*1024)` і так далі для `G, T, P, E, Z, Y`.

Літера «С» — Розгляньте два типи віртуалізації, що використовуються в сучасних системах: повна та часткова віртуалізація. Поясніть переваги та недоліки кожного з методів

Повна віртуалізація

У повній віртуалізації, віртуальна машина імітує достатньо обладнання, щоб дозволити незміненій гостьовій ОС (один розроблений для того ж набору команд) для запуску в ізоляції.

Повна віртуалізація працює швидше за емуляцію обладнання, але її швидкодія менша порівняно з роботою напряму. Найбільшою перевагою повної віртуалізацію в тому, що в гостьову операційну систему не вносяться зміни.

Єдиним обмеженням є те, що операційна система повинна підтримувати основні апаратні засоби.

Часткова віртуалізація

При такому підході віртуальна машина симулює кілька примірників апаратного оточення (але не всього), зокрема, простору адрес. Такий вид віртуалізації дозволяє спільно використовувати ресурси і ізолювати процеси, але не дозволяє розділяти примірники гостьових операційних систем. Строго кажучи, при такому вигляді віртуалізації користувачем не створюються віртуальні машини, а відбувається ізоляція будь-яких процесів на рівні операційної системи.

Тим не менше, в порівнянні з повною віртуалізацією, її недоліком є ситуації, що вимагають зворотної сумісності або переносимості: може бути важко передбачити, які саме особливості були використані в даному додатку. Якщо певні апаратні особливості не були змодельовані, то будь-яка програма, що буде використовувати ці функції, не буде працювати.

Оберіть один з дистрибутивів, перша літера назви якого співпадає з четвертою літерою прізвища. Охарактеризуйте його.

Літера «Н» — Hyperbola GNU/Linux-libre

Hyperbola GNU/Linux-libre — це дистрибутив Linux для комп'ютерів на базі архітектур i686, x86-64. Замість звичайного ядра Linux він використовує ядро Linux-libre, яке містить ви-

ключно вільні компоненти і не містить пропрієтарних частин. Цей дистрибутив заснований на дистрибутивах Arch (його знімків snapshot) та Debian вітки «development». Дистрибутив Hyperbola GNU/Linux відрізняється тим, що є повністю вільною операційною системою за версією Free Software Foundation. Також Hyperbola використовує init-систему OpenRC замість розповсюдженої systemd.

Напишіть скрипт

Необхідні можливості:

1. Розкриття та підстановка (результату обчислення арифметичних виразів, підстановка змінних, розкриття тільди, тощо)
2. Використання шаблонів та регулярних виразів.
3. Оператори `test` та `select`/`case`.

```
#!/bin/bash
# Скрипт для перевірки типу введеного символу

echo; echo "Натисніть клавішу, а потім Enter"

# Поки зчитаний символ не дорівнює "X"
# [] — оператор test
while [ "$Keypress" != "X" ]; do
    read Keypress # Зчитати клавішу, введену користувачем

    # п. 3 — Оператор case, п. 1 — підстановка змінної
    case "$Keypress" in
    # Тут і далі п. 2 — регулярні вирази [[:lower:]], [[:upper:]], [0-9], *
    # [[:lower:]] шукає малу літеру
      [[:lower:]]   ) echo "Маленька літера";;
    # [[:upper:]] — велику
      [[:upper:]]   ) echo "Велика літера";;
    # [0-9] — будь-який символ від 0 до 9
      [0-9]         ) echo "Цифра";;
    # * — будь-який символ
      *             ) echo "Інші символи";;
    esac # Кінець оператора case
done
```