МЕТОДИЧНА РОЗРОБКА
для проведення лабораторного заняття з дисципліни
«Клієнт-серверне програмування»

Лабораторна робота 2

Робота з файловою системою ОС Linux

Мета роботи: набуття досвіду роботи з файлами і каталогами в ОС Linux.

1. Теоретичні відомості

Файлова система Linux має багато каталогів, які утворюють ієрархічну деревоподібну систему. На початку цього дерева розташований кореневий каталог, який позначається як символ / (похила лінія). Далі розташовуються інші каталоги та файли. На відміну від операційних систем MS-DOS і Windows, тут майже всі каталоги мають стандартні імена і їх не можна вилучати або перейменовувати.

Каталог */root*. Він є робочим каталогом суперкористувача. Після реєстрації суперкористувач попадає саме в цей каталог.

В ОС Linux введено особливий режим використання облікового запису суперкористувача з ім'ям *root*. Обліковий запис *root* є головною обліковим записом в Linux та інших Unix-подібних операційних системах. Цей обліковий запис має доступ до всіх команд і файлів в системі з повними дозволами на читання, запис і виконання. Він використовується для виконання будь-яких системних задач: створення / оновлення / отримання доступу / видалення облікових записів інших користувачів, установки / видалення / оновлення програмних пакетів і багато чого іншого. Оскільки користувач *root* має абсолютними повноваженнями, будь-які виконувані ним дії є критичними для системи. У зв'язку з цим будь-які помилки користувача *root* можуть мати величезний вплив на нормальну роботу системи. Тому рекомендується відключити доступ до аккаунту та створити обліковий запис адміністратора, який буде налаштований на отримання привілеїв користувача *root* за допомогою команди *sudo* для виконання критичних завдань на сервері.

Якщо необхідно мати обліковий запис суперкористувача *root*, її можна активувати за допомогою наступної команди: **sudo passwd root.**

Зазначена команда ініціює стандартну діалогову процедуру призначення пароля користувача (в даному випадку - суперкористувача з ім'ям root).

Відповідно для відключення облікового запису root слід використовувати наступну команду: **sudo passwd -l root.**

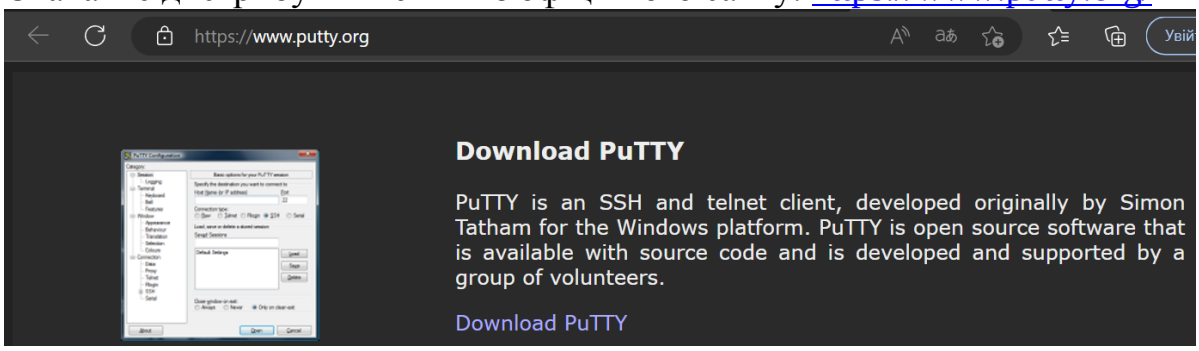
Реально, зазначена команда не видаляє, а лише блокує обліковий запис.

Завдання:

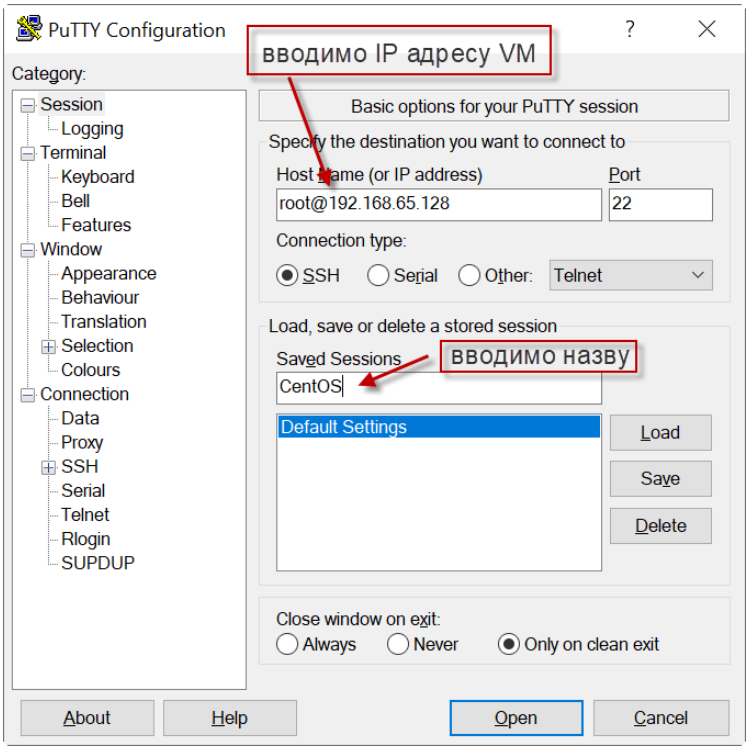
1. Встановити PuTTY - програмний емулятор терміналу для Windows та Linux. Він надає текстовий інтерфейс для віддалених комп’ютерів, що

використовують будь-який з підтримуваних протоколів, включаючи SSH, TCP, rlogin і Telnet. Це популярний інструмент для підключення серверів Linux до комп'ютерів під керуванням Windows.

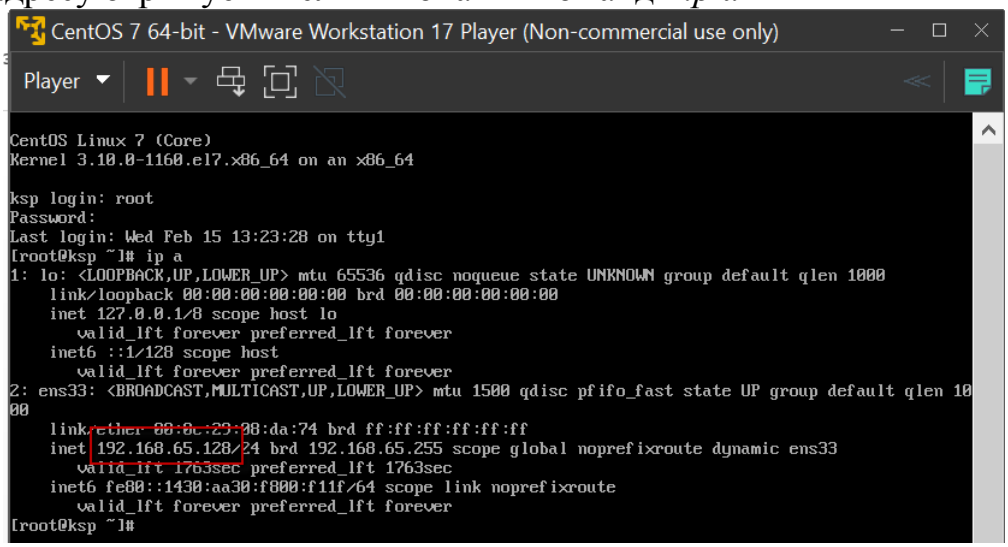
Скачайте дистрибутив PuTTY з офіційного сайту: https://www.putty.org/



2. Встановіть комп'ютер PuTTY.
3. Виконайте налаштування. Обираємо протокол SSH та 22 порт.

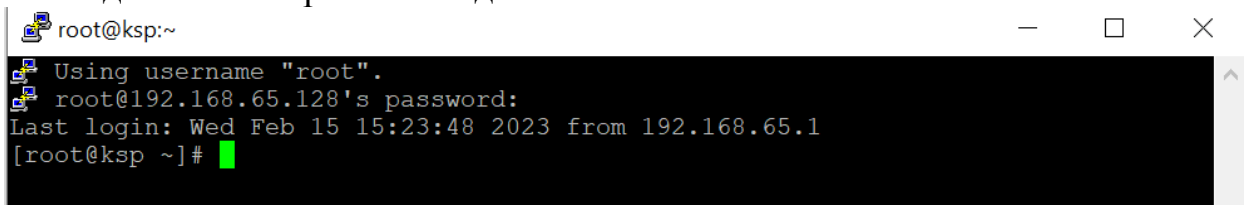


4. IP адресу отримуємо після виконання команди *ip a*



5. Зберігаємо (save) та відкриваємо (open)

6. Вводимо свій пароль і заходим в PuTTY.



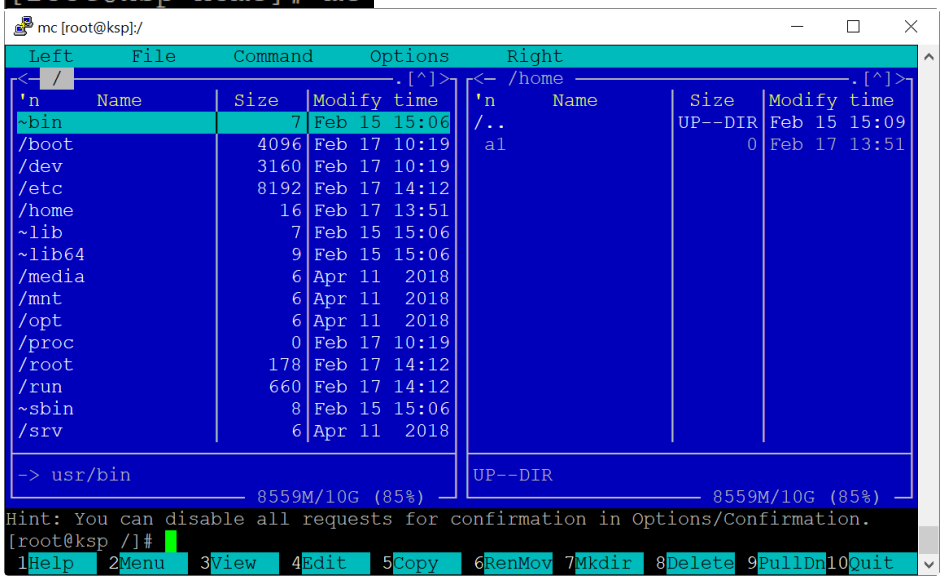
7. Встановити файловий менеджер *mc* «Midnight Commander» для Unix-подібних операційних систем, клон «Norton Commander». Основний інтерфейс складається з двох панелей, на яких відображена файлова система.

Використовуємомкоманду ***yum install mc***

```
[root@ksp home]# yum install mc
```

Заходим в ***mc***

```
Complete!
[root@ksp home]# mc
```



8. Встановити текстовий редактор **nano** (за бажанням). За замовчуванням встановлено редактор **vi**. За бажанням можна виконувати наступні завдання в редакторі **vi**.

```
[root@ksp home]# yum install nano
```

2. Найпростіші команди ОС Linux

Робота в Linux за допомогою командного рядка, нагадує діалог з системою: користувач вводить команди (репліки), отримуючи від системи відповідні репліки, які мають відомості про проведені операції, додаткові запитання до користувача, повідомлення про помилки або просто мовчазне погодження виконати наступну команду.

Найпростіша команда в Linux складається з одного слова – назви програми, яку необхідно виконати.

Приклад 1. Команда ***whoami***

```
[root@ksp /]# whoami
root
[root@ksp /]#
```

Назва цієї команди походить від англійського виразу «Who am I?» («Хто я?»). У відповідь на цю команду система вивела тільки одне слово: «root» і завершила свою роботу, про що свідчить запрошення командного рядка, яке знову з'явилося. Програма whoami повертає назву облікового запису, від імені якого вона була виконана. Ця команда корисна в системах, в яких працює багато різних користувачів, щоби не скористатися помилково чужим обліковим записом.

Приклад 2. Команда ***who***

```
[root@ksp ~]# who
root     tty1         2023-02-17 10:19
root     pts/0        2023-02-17 10:20 (192.168.65.1)
```

who виводить список користувачів, які в даний момент зареєстровані в системі. Дана програма виводить по одному рядку на кожного зареєстрованого користувача: у першій колонці вказується ім'я користувача, в другій – «точка входу» в систему, далі іде дата і час реєстрації і ім'я хоста.

Приклад 3. Команда ***last***

Ще одна програма, яка видає інформацію про користувачів, які працювали в системі останнім часом – last. Рядки, які виводяться цією програмою, нагадують вивід програми who, з тою різницею, що тут перераховані і ті користувачі, які вже завершили роботу.

```
[root@ksp ~]# last
root     pts/0        192.168.65.1     Fri Feb 17 11:34   still logged in
root     pts/0        192.168.65.1     Fri Feb 17 10:20 - 11:33  (01:12)
root     tty1                          Fri Feb 17 10:19   still logged in
reboot   system boot  3.10.0-1160.el7. Fri Feb 17 10:19 - 11:34  (01:15)
root     pts/0        192.168.65.1     Wed Feb 15 15:32 - crash (1+18:47)
root     pts/0        192.168.65.1     Wed Feb 15 15:23 - 15:23  (00:00)
root     pts/0        192.168.65.1     Wed Feb 15 15:19 - 15:19  (00:00)
root     tty1                          Wed Feb 15 13:59 - crash (1+20:19)
reboot   system boot  3.10.0-1160.el7. Wed Feb 15 13:59 - 11:34 (1+21:35)
root     tty1                          Wed Feb 15 13:23 - crash  (00:35)
reboot   system boot  3.10.0-1160.el7. Wed Feb 15 13:22 - 11:34 (1+22:12)
root     tty1                          Wed Feb 15 13:13 - crash  (00:09)
reboot   system boot  3.10.0-1160.el7. Wed Feb 15 13:12 - 11:34 (1+22:22)
reboot   system boot  3.10.0-1160.el7. Wed Feb 15 15:10 - 11:34 (1+20:24)

wtmp begins Wed Feb 15 15:10:35 2023
[root@ksp ~]#
```

В виведені програми last з'являється користувач reboot (перезавантаження). В дійсності, такого облікового запису немає, програма last таким засобом виводить інформацію про те, коли була завантажена система.

Приклад 4. Команда ***logout***

```
ksp login: root
Password:
Last login: Fri Feb 17 11:34:35 from 192.168.65.1
[root@ksp ~]# logout
```

У відповідь на цю команду замість чергового запрошення командного рядка поновлюється запрошення до реєстрації в системі. На даній віртуальній

консолі робота з користувачем завершена, і тепер тут знову може зареєструватися любий користувач.

3. **Робота з файловою системою.**

Кожна програма, яка виконується, «працює» у строго визначеному каталозі файлової системи. Такий каталог називається поточним каталогом, можна уявляти, що програма під час роботи «знаходиться» у цьому каталозі, це її «робоче місце». В залежності від поточного каталогу може змінюватися поведінка програми: часто програма буде працювати, по замовчуванню, з файлами, які розташовані саме у поточному каталозі – до них вона «дотягнеться» у першу чергу.

Приклад 1. Поточний каталог: ***pwd***

```
[root@ksp ~]# pwd
/root
```

pwd (абревіатура від print working directory) повертає повний шлях поточного каталогу командної оболонки, природно, саме тієї командної оболонки, за допомогою якої була виконана команда pwd.

Інформація про каталог

Щоби мати можливість орієнтуватися у файловій системі, треба знати вміст кожного каталогу. Запам'ятати усю структуру файлової системи не можливо і не треба: у любий момент можна продивитися вміст любого каталогу за допомогою утиліти ls (скорочення від англійської «list» - «список»).

Приклад 2. Команда ***ls***

```
[root@ksp ~]# ls
anaconda-ks.cfg
```

Команда ***ls*** без параметрів виводить список і каталогів, які вміщуються у поточному каталозі.

Утиліта ***ls*** приймає один параметр: ім'я каталогу, вміст якого треба вивести. Ім'я може бути задано любим доступним засобом: у вигляді повного або відносного шляху.

Крім параметру утиліта ***ls*** «розуміє» багато ключів, які потрібні, головним чином для того, що би виводити додаткову інформацію про файли у каталозі і виводити список файлів вибірково.

Приклад 3. Команда ***ls –aF***

```
[root@ksp ~]# ls -aF
./   anaconda-ks.cfg  .bash_logout   .bashrc  .tcshrc
../  .bash_history   .bash_profile  .cshrc
```

Раптово ми виявили, що файлів у нашому каталозі не два, а значно більше. Справа в тому, що утиліта ls, по замовченню, не виводить інформацію про об'єкти, чиє ім'я починається з «.» - в тому числі, з «.» і «..».

Як правило, з «.» починаються імена конфігураційних файлів і конфігураційних каталогів, робота з якими не перетинається з роботою над якою-небудь прикладною задачею.

Символ «.» - це ім'я поточного каталогу. Наступне ім'я у списку «..» - це батьківський каталог. Батьківський каталог – це каталог, в якому знаходиться даний.

Батьківський каталог - каталог у якому вміщується даний. Для корінного каталогу батьківським являється він сам.

Приклад 4. Команда ***ls –a***

Дана утиліта надає можливість вивести список всіх файлів і каталогів.

```
[root@ksp /]# ls -a
.   bin   dev  home  lib64  mnt  proc  run   srv  tmp  var
..  boot  etc  lib   media  opt  root  sbin  sys  usr
```

Команда ***ls –al***

Дана утиліта надає можливість вивести список всіх файлів і каталогів з повною інформацією.

```
[root@ksp /]# ls -al
total 20
dr-xr-xr-x.  17 root root  224 Feb 15 15:09 .
dr-xr-xr-x.  17 root root  224 Feb 15 15:09 ..
lrwxrwxrwx.   1 root root    7 Feb 15 15:06 bin -> usr/bin
dr-xr-xr-x.   5 root root 4096 Feb 17 10:19 boot
drwxr-xr-x.  19 root root 3160 Feb 17 10:19 dev
drwxr-xr-x.  73 root root 8192 Feb 17 10:19 etc
drwxr-xr-x.   2 root root    6 Apr 11  2018 home
lrwxrwxrwx.   1 root root    7 Feb 15 15:06 lib -> usr/lib
lrwxrwxrwx.   1 root root    9 Feb 15 15:06 lib64 -> usr/lib64
drwxr-xr-x.   2 root root    6 Apr 11  2018 media
drwxr-xr-x.   2 root root    6 Apr 11  2018 mnt
drwxr-xr-x.   2 root root    6 Apr 11  2018 opt
dr-xr-xr-x. 112 root root    0 Feb 17 10:19 proc
dr-xr-x---.   2 root root  135 Feb 17 11:33 root
drwxr-xr-x.  24 root root  660 Feb 17 10:19 run
lrwxrwxrwx.   1 root root    8 Feb 15 15:06 sbin -> usr/sbin
drwxr-xr-x.   2 root root    6 Apr 11  2018 srv
dr-xr-xr-x.  13 root root    0 Feb 17 10:19 sys
drwxrwxrwt.  12 root root 4096 Feb 17 11:17 tmp
drwxr-xr-x.  13 root root  155 Feb 15 15:06 usr
drwxr-xr-x.  19 root root  267 Feb 15 15:10 var
[root@ksp /]#
```

Переміщення по дереву каталогів

Користувач може працювати з файлами не тільки у своєму домашньому каталозі, але і в інших каталогах. У цьому випадку буде зручно замінити поточний каталог, тобто «переміститися» в іншу точку файлової системи. Для зміни поточного каталогу командної оболонки використовується команда ***cd*** (від англійської «change directory» - «змінити каталог»). Команда *cd* приймає один параметр: ім’я каталогу в який треба переміститися – зробити поточним.

Зазвичай, в якості імені каталогу можна використати повний і відносний шлях.

Приклад 5. Зміна поточного каталогу ***cd /home***

```
[root@ksp /]# cd /home
[root@ksp home]# ls
[root@ksp home]# ls -al
total 0
drwxr-xr-x.  2 root root   6 Apr 11  2018 .
dr-xr-xr-x. 17 root root 224 Feb 15 15:09 ..
```

Спочатку ми перемістилися у каталог «/home», подивились що ще є у цьому каталозі.

Приклад 6. Перехід в батьківський каталог ***cd ..***

```
[root@ksp home]# cd ..
[root@ksp /]#
```

Створення файлів та каталогів

Користувач, звісно, не повинен зберігати усі свої файли в одному каталозі.

У домашньому каталозі користувача, як і у любому іншому, можна створювати скільки завгодно підкаталогів, у них своїх підкаталогів і так далі. Іншими словами, користувачу належить фрагмент (піддерево) файлової системи, корінням якого є домашній каталог користувача. В даних каталогах користувач може створювати файли.

Приклад 7. Створення порожнього файлу ***touch a1***

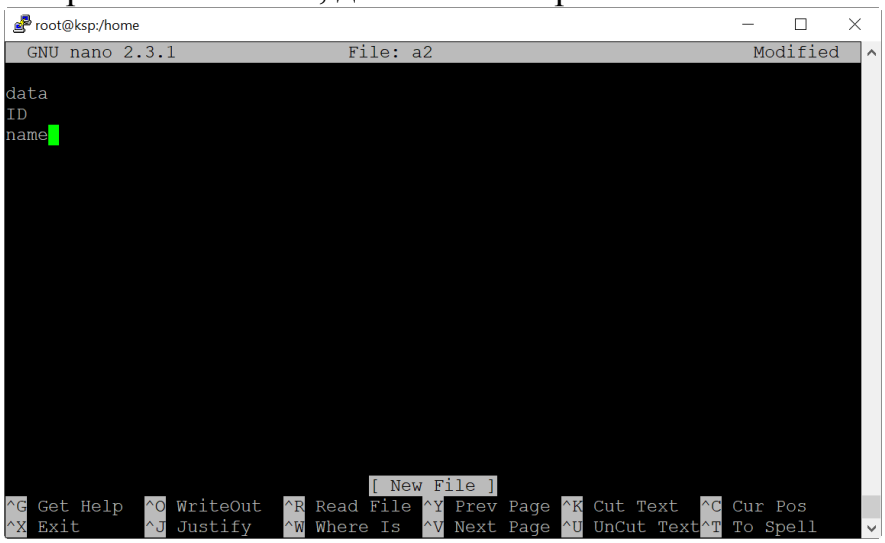
```
[root@ksp home]# touch a1
[root@ksp home]# ls
a1
[root@ksp home]# ll
total 0
-rw-r--r--. 1 root root 0 Feb 17 13:51 a1
```

Створили порожній файл a1 та перевірили його наявність в поточному каталозі.

Командою ***ll*** можна перевірити розмір файлу, який дорівнює 0 (порожній файл).

Приклад 8. Створення файлу a2 з використанням редактора nano.

Потрапляєм в вікно, де можем набрати якийсь текст та зберегти.



Приклад 9. Перегляд вмісту файлу a2 за допомогою команди ***cat a2***.

```
[root@ksp home]# cat a2
data
ID
name
[root@ksp home]#
```

Приклад 10. Створення каталогу ***mrdir F1*** і перевіряємо командою ***ll***.

```
[root@ksp home]# mkdir F1
[root@ksp home]# ll
total 8
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 1 root root 13 Feb 17 14:41 a2
-rw-r--r--. 1 root root 13 Feb 17 14:52 a3
drwxr-xr-x. 2 root root  6 Feb 17 14:56 F1
```

Копіювання і переміщення файлів

Для переміщення файлів і каталогів призначена утиліта ***mv*** (скорочення від англійської «move» - «переміщувати»). У ***mv*** два обов'язкових параметри: перший – файл або каталог, який переміщується, другий - файл або каталог призначення. Імена файлів або каталогів можуть бути задані у любому допустимому вигляді: за допомогою повного або відносного шляху. Крім того, ***mv*** дозволяє переміщати не тільки один файл або каталог, а відразу декілька.

Приклад 11. Переміщення файлів ***mv a2 text2.txt***

Переміщення файлу у середині однієї файлової системи у дійсності рівнозначно його перейменуванню: дані самого файлу залишаються на тих же секторах диску, змінюються каталоги, у яких пройшло переміщення.

Переміщення передбачає знищення посилання на файл з того каталогу звідки його переміщено, і додання посилання на той самий каталог, куди він переміщений. У результаті змінюється повне ім'я файлу – повний шлях, тобто, положення файлу у файловій системі.

```
[root@ksp home]# mv a2 text2.txt
[root@ksp home]# ll
total 12
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 1 root root 13 Feb 17 14:52 a3
-rw-r--r--. 1 root root 13 Feb 17 16:49 a4
drwxr-xr-x. 2 root root 16 Feb 17 16:53 F1
-rw-r--r--. 1 root root 13 Feb 17 14:41 text2.txt
```

Приклад 12. Копіювання файлів ***cp a2 a4***

Копіювання файлу в каталог F1 ***cp a2 F1/***

Для цієї простішої операції копіювання достатньо передати ***cp*** у якості двох параметрів ім'я початкового файлу і ім'я копії. За замовченням ***cp***, як і багато інших утиліт, буде працювати з файлами у поточному каталозі.

```
[root@ksp home]# cp a2 a4
[root@ksp home]# ll
total 12
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 1 root root 13 Feb 17 14:41 a2
-rw-r--r--. 1 root root 13 Feb 17 14:52 a3
-rw-r--r--. 1 root root 13 Feb 17 16:49 a4
[root@ksp home]# cp a2 F1/
[root@ksp home]# ll F1/
total 4
-rw-r--r--. 1 root root 13 Feb 17 16:53 a2
```

Знищення файлів і каталогів

У Linux для знищення файлів призначена утиліта ***rm*** (скорочення від англійської «remove» - «знищувати»).

Приклад 13. Знищення файлів ***rm a3***

```
[root@ksp home]# rm a3
rm: remove regular file 'a3'? yes
[root@ksp home]# ll
total 8
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 1 root root 13 Feb 17 16:49 a4
drwxr-xr-x. 2 root root 16 Feb 17 16:53 F1
-rw-r--r--. 1 root root 13 Feb 17 14:41 text2.txt
```

Іде запит на підтвердження знищення файлу

Приклад 14. Знищення каталогу ***rmdir F1***

```
[root@ksp home]# rmdir F1
rmdir: failed to remove 'F1': Directory not empty
```

Для знищення каталогів призначена інша утиліта ***rmdir*** (від англійської «remove directory»). Втім, ***rmdir*** згодиться знищити каталог, якщо він пустий: у ньому немає ніяких файлів і підкаталогів. Знищити каталог разом з його вмістом, можна командою ***rm*** з ключем «-r» (recursive).

Увага!

Пам'ятайте, якщо ви знищили файл, це означає, він вже не потрібен, і не підлягає відновленню.

В Linux не передбачені процедури відновлення знищених файлів і каталогів. Тому, варто бути дуже обережним відаючи команду ***rm*** і тим більше: ***rm –r***, немає ніякої гарантії, що вдасться відновити випадково знищені дані.

Перегляд вмісту файлів

У Linux для перегляду вмісту файлів застосовуються утиліти ***cat*** та ***more***.

Приклад 14. Перегляд вмісту файлу ***more a4***

```
[root@ksp home]# more a4
data
ID
name
[root@ksp home]#
```

Виведення на екран повідомлення

У Linux для виведення на екран повідомлення застосовується утиліти ***echo***.

Приклад 15. Виведення на екран повідомлення ***echo "HELLO"***

```
[root@ksp home]# echo "HELLO"
HELLO
[root@ksp home]#
```

Жорсткі посилання

Кожний файл уявляє собою область даних на жорсткому диску комп'ютера або іншого носія інформації, яку можна знайти по імені. У файловій системі Linux вміст файлу пов'язується з його іменем за допомогою **жорстких посилань**. Утворення файлу за допомогою любої програми означає, що буде утворене жорстке посилання – ім'я файлу, і відкрита нова область даних на диску. Причому, кількість посилань на одну й ту ж область даних (файл) не обмежено, тобто, у файла може бути декілька імен.

Приклад 16. Утворення жорстких посилань ***ln a4 a3***

```
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 1 root root 13 Feb 17 16:49 a4
drwxr-xr-x. 2 root root 16 Feb 17 16:53 F1
-rw-r--r--. 1 root root 13 Feb 17 14:41 text2.txt
[root@ksp home]# ln a4 a3
[root@ksp home]# ll
total 12
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 2 root root 13 Feb 17 16:49 a3
-rw-r--r--. 2 root root 13 Feb 17 16:49 a4
drwxr-xr-x. 2 root root 16 Feb 17 16:53 F1
-rw-r--r--. 1 root root 13 Feb 17 14:41 text2.txt
```

Користувач Linux може добавити файлу ще одне ім'я (утворити ще одне жорстке послання на файл) за допомогою утиліти ***ln*** (скорочення від англійської «link» - «зв'язувати»). Перший параметр – це ім'я файлу, на який треба утворити посилання, другий – ім'я нового посилання. За замовченням, посилання буде утворене у поточному каталозі.

Приклад 17. Утворення символічних посилань ***ln –s a4 a5***

```
-rw-r--r--. 1 root root  0 Feb 17 13:51 a1
-rw-r--r--. 2 root root 19 Feb 18 11:45 a3
-rw-r--r--. 2 root root 19 Feb 18 11:45 a4
lrwxrwxrwx. 1 root root  2 Feb 18 11:56 a5 -> a4
```

Символічне посилання (symlink) — це спеціальний файл, який містить адресу іншого файлу, який відкривається при зверненні до символічного посиланням. Таке посилання може використовуватися для організації доступу до одного каталогу з декількох сайтів, без створення його копій. Символічні посилання можуть бути спрямовані на каталог або файл.

3. **Завдання на виконання.**

1. Ознайомитися з теоретичними матеріалом по лабораторній роботі.
2. Навести список користувачів, які в даний момент зареєстровані в системі.
3. Зайти в домашній каталог ***home/.***
4. Створіть в домашньому каталозі ***home/*** каталог користувача під власним іменем.
5. Покажіть повний шлях до поточного каталогу користувача.
6. Створіть в поточному каталозі порожній файл під своїми ініціалами (прізвище латинськими літерами)
7. Зробіть копію даного файлу у файл id.txt.
8. Відкрийте файл id.txt за допомогою редактора і у файлі наберіть інформацію про себе (прізвище, ім'я, групу).
9. Відобразіть інформацію з файлу id.txt на моніторі і зробіть срін.
10. Виконайте жорстке посилання файлу id.txt на id2.txt. Додайте в id2.txt назву свого міста та покажіть, чи відбулись зміни у файлі id.txt (зробити скрін).
11. Зробіть перегляд файлів у поточному каталозі та зробіть скріню
12. Видаліть файл зі власним іменем та зробіть перегляд файлів у поточному каталозі (зробіть скрін).
13. Перейдіть у батьківський каталог.

14. Виведіть на екран повідомлення “HELLO NAME” (name – має бути ваше ім’я) та зробіть скрін.

Підготувати звіт про виконання лабораторної роботи та зробіть висновки.

3. Результати виконання роботи та оформлення звіту

Зміст звіту:
1) титульний аркуш;
2) короткі теоретичні відомості;
3) скріни з виконаними завданнями та висновки.

Контрольні питання
1. Що таке файлова система?
2. Що таке каталог?
3. Що таке шлях до файлу?
4. Абсолютний і відносний шлях?
5. Типи файлів які вам відомі?
6. Посилання. Типи посилань.
7. Команда створення посилання.
8. Команди для роботи з каталогами.
9. Команди для роботи з файлами.

Методичну розробку склав
доцент кафедри інформаційних технологій
та систем електронних комунікацій Юрій БОРЗОВ