Міністерство освіти і науки України
Сумський державний університет

Методичні вказівки

до виконання розрахункової роботи
з дисципліни «**Мережеві операційні системи**»
для студентів
спеціальності 172 *«Телекомунікації та радіотехніка»*
денної форми навчання

Суми
Сумський державний університет
2022

Методичні вказівки до виконання розрахункової роботи з дисципліни «Мережеві операційні системи» / укладачі: В. В. Гриненко, О. В. Д’яченко. – Суми : Сумський державний університет, 2022. – 31 с.

Кафедра електроніки і комп’ютерної техніки

Мета розрахункової роботи (РР) – поглиблення і закріплення теоретичних знань із курсу «Мережеві операційні системи», одержання практичних навичок роботи в ОС Linux та написання скриптів для виконання певних завдань. Під час виконання курсової роботи студенти повинні використовувати принципи та методи, викладені у відповідних розділах лекційного курсу та опрацьовані на лабораторних заняттях.

Виконання, оформлення та захист розрахункової роботи

Для виконання завдань потрібно використовувати вільно розповсюджуване програмне забезпечення, яке може бути отримане з Інтернету. При виконанні роботи в середовищі операційної системи Linux рекомендується використовувати пакети програмного забезпечення з актуальних системних репозиторіїв.

По завершенню виконання розрахункової роботи студент виконує звіт, який повинен містити:

1. Титульний аркуш.
2. Номер варіанту і зміст індивідуального завдання;
3. Протокол виконання роботи з коментарями, де чітко позначено, що вводить користувач з клавіатури, і що в результаті з’являється на екрані (необхідно показати всі результати ілюстровані лістингом; якщо іде запис даних в файл, вміст файлу також слід роздрукувати)
4. Список літератури, який використовується для написання розрахунково роботи.

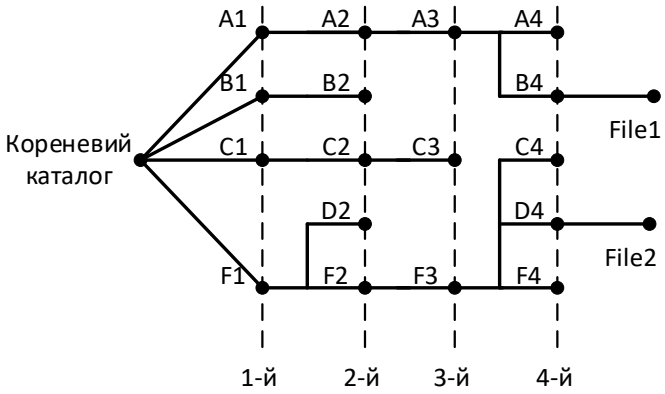
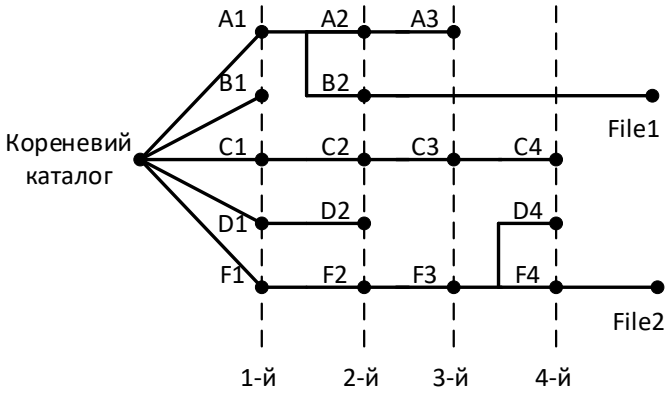
На перевірку викладачу надається оформлений звіт та тексти скриптів і файлів, що використовувались для перевірки правильності виконання завдань.

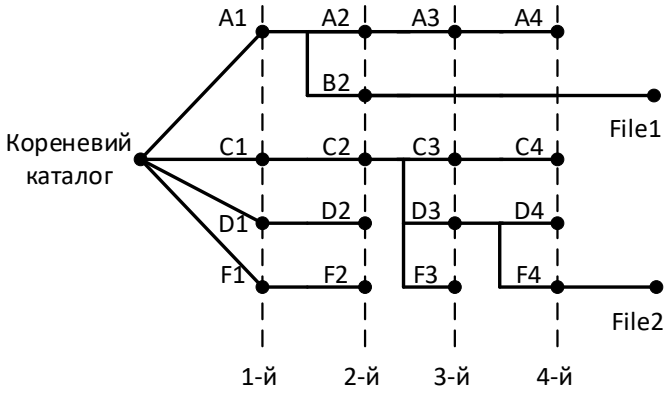
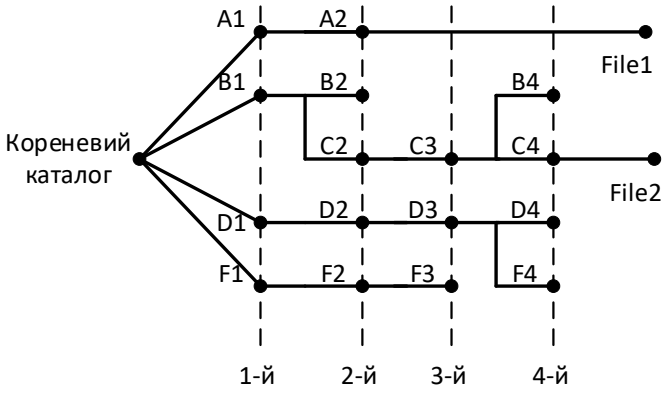
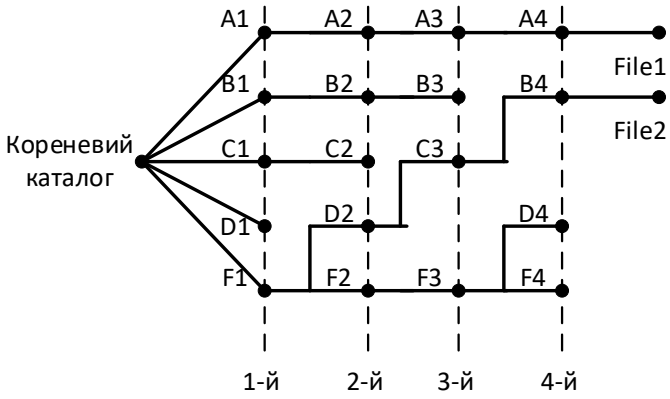
Під час захисту роботи студенти повинен дати правильні відповіді на додаткові запитання за матеріалами цієї розрахункової роботи. Зверніть увагу на те, що студент повинен володіти всім теоретичним і практичним матеріалом, а не тільки тією його частиною, яка безпосередньо входила до індивідуального завдання.

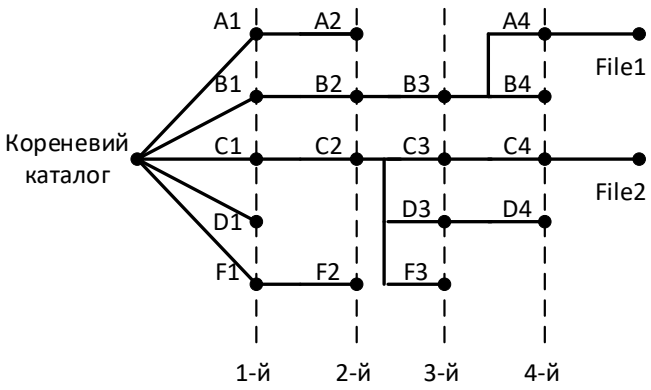
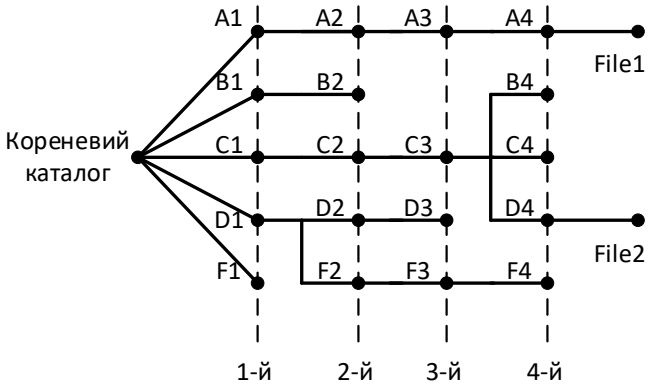
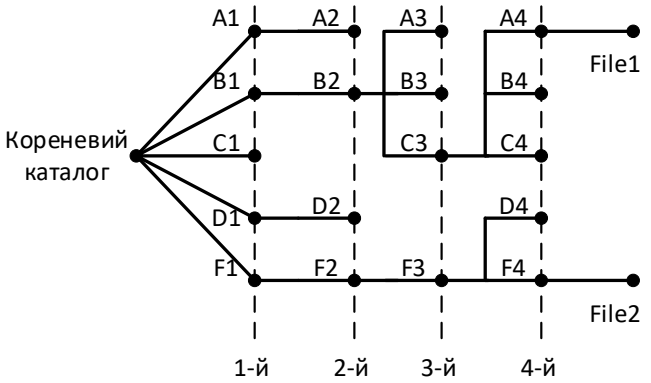
Завдання 1. Створення об’єктів файлової системи

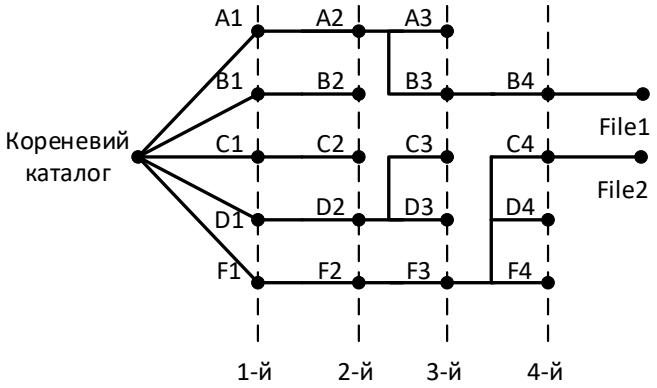
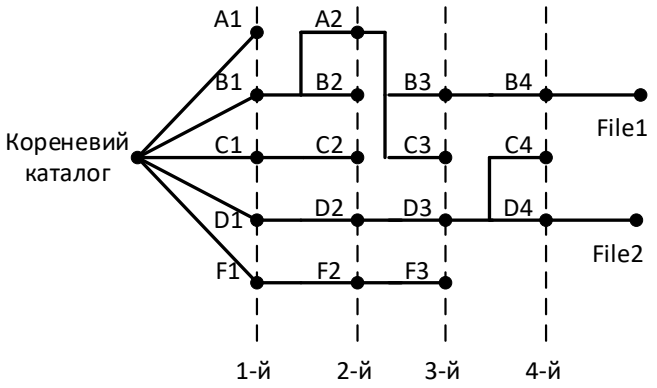
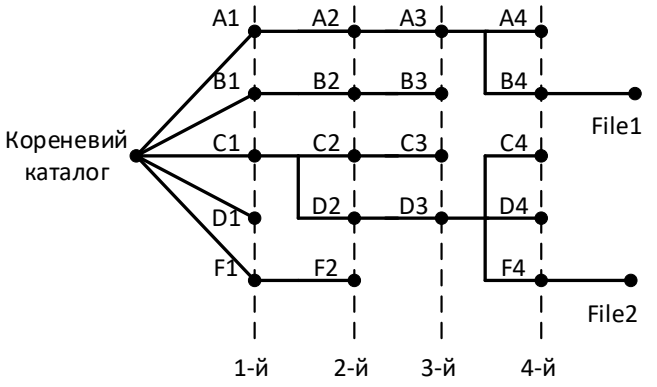
1. Зайдіть в систему під своїм логіном. У домашній директорії користувача необхідно створити директорію ***task1*** з правами доступу ***755*** *(rwxrwxr-x)* для директорій та **664** для файлів *(rw-rw-r--)*. В директорії ***task1*** створіть структуру каталогів, та файли (***File1*** та ***File2***) як вказано у таблиці 1.

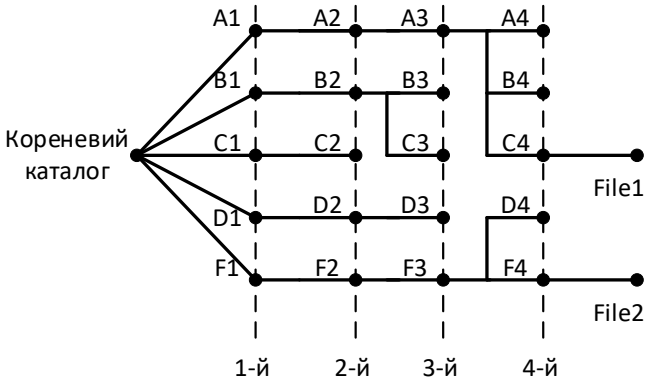
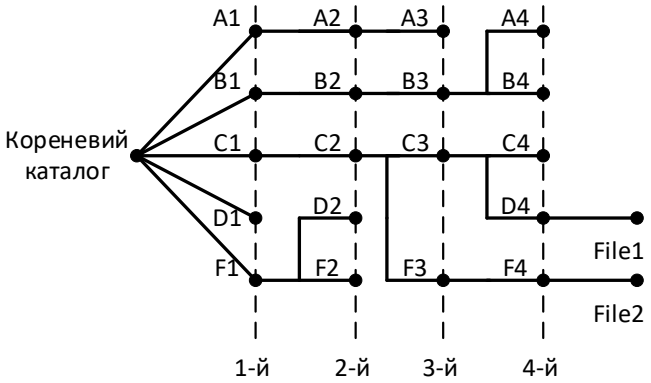
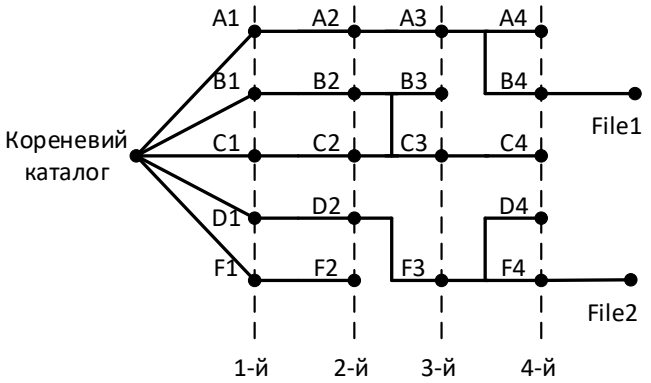
Таблиця 1 – Завдання до п. 1.

Номер варіанту	**Структура каталогів**
1	**2**
1	
2	

1	2
3	 A1 A2 A3 A4 B2 File1 Кореневий каталог C1 C2 C3 C4 D1 D2 D3 D4 F1 F2 F3 F4 File2 1-й 2-й 3-й 4-й
4	 A1 A2 File1 B1 B2 B4 Кореневий каталог C2 C3 C4 File2 D1 D2 D3 D4 F1 F2 F3 F4 1-й 2-й 3-й 4-й
5	 A1 A2 A3 A4 File1 B1 B2 B3 B4 File2 Кореневий каталог C1 C2 C3 D1 D2 D4 F1 F2 F3 F4 1-й 2-й 3-й 4-й

1	2
6	 A1 A2 A4 File1 B1 B2 B3 B4 Кореневий каталог C1 C2 C3 C4 File2 D3 D4 D1 F1 F2 F3 1-й 2-й 3-й 4-й
7	 A1 A2 A3 A4 File1 B1 B2 B4 Кореневий каталог C1 C2 C3 C4 D1 D2 D3 D4 File2 F1 F2 F3 F4 1-й 2-й 3-й 4-й
8	 A1 A2 A3 A4 File1 B1 B2 B3 B4 Кореневий каталог C1 C3 C4 D2 D4 D1 F1 F2 F3 F4 File2 1-й 2-й 3-й 4-й

1	2
9	 A1 A2 A3 B1 B2 B3 B4 Кореневий каталог C1 C2 C3 C4 File1 File2 D1 D2 D3 D4 F1 F2 F3 F4 1-й 2-й 3-й 4-й
10	 A1 A2 B1 B2 B3 B4 Кореневий каталог File1 C1 C2 C3 C4 D1 D2 D3 D4 File2 F1 F2 F3 1-й 2-й 3-й 4-й
11	 A1 A2 A3 A4 B1 B2 B3 B4 Кореневий каталог File1 C1 C2 C3 C4 D1 D2 D3 D4 F1 F2 F4 File2 1-й 2-й 3-й 4-й

1	2
12	 A1 A2 A3 A4 B1 B2 B3 B4 Кореневий каталог C1 C2 C3 C4 File1 D1 D2 D3 D4 F1 F2 F3 F4 File2 1-й 2-й 3-й 4-й
13	 A1 A2 A3 A4 B1 B2 B3 B4 Кореневий каталог C1 C2 C3 C4 D1 D2 D4 File1 F1 F2 F3 F4 File2 1-й 2-й 3-й 4-й
14	 A1 A2 A3 A4 B1 B2 B3 B4 File1 Кореневий каталог C1 C2 C3 C4 D1 D2 D4 F1 F2 F3 F4 File2 1-й 2-й 3-й 4-й

2. З кореневої директорії (***task1***) в файл ***File1*** запишіть інформацію, що зберігається в довідці ***man*** команди, зазначеної в стовпці 2, а в файл ***File2*** запишіть список файлів, що знаходяться в каталогах (стовпець 3) назва яких починається на літери зазначені в стовпці 4.

Таблиця 2 – Завдання до п. 2.

Номер варіанту	**Команда**	**Каталог**	**Літери**
1	useradd	/run /usr/bin	b, a, d, y
2	ls	/bin /proc	u, a, v, w
3	find	/proc /run	s, t, m, z
4	grep	/usr/bin /etc	a, b, c, d
5	cp	/etc /lib	e, f, g, h
6	less	/run /usr/bin	i, j, k, l
7	df	/bin /proc	m, n, o, p
8	mount	/usr /proc	q, r, s, t
9	ps	/usr/bin /proc	u, v, w
10	passwd	/etc /usr/bin	x, y, z
11	chmod	/var /usr/bin	a, d, k, l
12	du	/lib /usr/bin	m, g, y
13	umount	/dev /proc	b, x, d, o
14	lsblk	/usr/bin /etc	c, o, k, l

3. У домашній директорії користувача створити файл ***stat.txt***, та занесіть в нього наступну інформацію про перший файл:

- ID та назву групи власника файлу.
- Інформацію згідно таблиці 3.

Таблиця 3 – Завдання до п. 3.

Варіант	**Опція пошуку**
1	Кількість слів у 25-и перших рядках 2-го файлу, в яких зустрічається назва команди з п. 2.
2	Кількість слів в 20-и останніх рядках 2-го файлу, в яких зустрічається назва команди з п. 2.
3	Кількість літер у першому рядку 2-го файлу, в якому зустрічається назва команди з п. 2.
4	Кількість літер в останньому рядку 2-го файлу, в якому зустрічається назва команди з п. 2.
5	Скільки разів з 10-й по 50-й рядок файлу зустрічається назва команди з п. 2.
6	Скільки разів в 50-ти перших рядках файлу зустрічається назва команди з п. 2.
7	Скільки разів в 50-ти останніх рядках файлу зустрічається назва команди з п. 2.
8	Підрахуйте кількість слів в рядках з 1-го по 20-й 2-го файлу де зустрічається команда з п. 2.
9	Підрахуйте кількість символів в останніх 20 рядках 2-го файлу де зустрічається команда з п. 2.
10	Підрахуйте кількість слів в останніх 20 рядках 2-го файлу де не зустрічається команда з п. 2.
11	Підрахуйте кількість символів в рядках з 10-го по 20-й де не зустрічається команда з п. 2.
12	Скільки слів у 5-и рядках після першого рядку у 2-му файлі де зустрічається назва команди з п. 2.
13	Скільки символів у 3-х рядках до рядку першого співпадіння у 2-му файлі де зустрічається назва команди з п. 2.
14	Кількість рядків у 2-му файлі, в яких не зустрічається назва команди з п. 2.

та інформацію про другий файл:

- Кількість рядків у першому файлі.
- I-node файлу, його розмір, ID та ім’я власника файлу.

4. Змініть права доступу до першого, другого файлу та файлу stat.txt, а також директорій де зберігаються зазначені файли відповідно до таблиці.

Таблиця 4 - Завдання до п. 4.

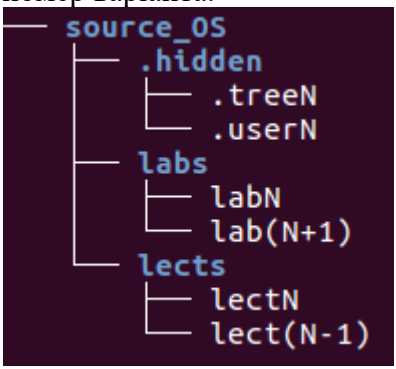
Варіант	**Права для файлів**	**Права для каталогів**
1	644	754
2	624	774
3	6-4	7-5
4	62-	73-
5	644	745
6	624	764
7	6-4	715
8	62-	763
9	644	744
10	624	765
11	6-4	715
12	6--	7-3
13	6-2	753
14	6-4	712
15	662	755

5. У домашній директорії користувача створіть директорію ***Student*** та створіть в ньому жорстке посилання на директорію в якій зберігається ***File1*** та символічне посилання на директорію де міститься ***File2***.

Приклад оформлення завдання 1

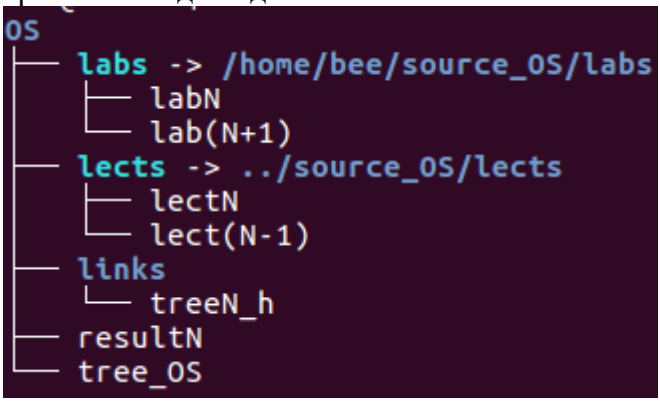
Завдання

1.1. У домашній директорії користувача створити директорію **source_OS** із відповідним вмістом та правами доступу **755 (rwxrwxr-x)** для директорій та **664** для файлів **(rw-rw-r--)**,
де ***N*** – номер варіанта.



Файли **labN**, **lab(N+1)** наповнити рядками **labN**, **lab(N+1)** відповідно; **.userN** – іменем поточного користувача, **.treeN** – деревом **source_OS** із відображенням прихованих файлів.

1.2. У домашній директорії користувача створити директорію OS із відповідним вмістом:



Файли **labs**, **lects** – символічні посилання на відповідні директорії; **treeN_h** – жорстке посилання на файл

source_OS/.hiddeen/.treeN, **tree_OS** містить дерево директорії, файл **resultN** містить результати виконання лабораторної роботи.

2. Створення, форматування та монтування розділів диску.

3. Переміщення об'єктів ФС на монтовані розділи та перевірка властивостей зв'язків.

4. Підготовка файлів звіту resultN.txt, lsN.txt

1. Послідовність виконання завдання

Від імені користувача:

1.1. У домашній директорії користувача створюємо директорію source_OS та наповнюємо її.

source_OS
├─────.hidden
├─────labs
└─────lects

1.1.0. Встановлюємо права доступу на створені файли та директорії **002**:

umask 002

Ця маска передбачає права доступу на створені файли та директорії 775 (rwxrwxr-x).

1.1.1. Створюємо дерево директорій source_OS:

```
bee@home:~$     mkdir source_OS
bee@home:~$     cd source_OS
bee@home:~/source_OS$     mkdir lects labs .hidden
```

Можна скористатися опцією **-p** команди **mkdir** і створити дерево, виконавши лише одну команду.

Перевірка:

```
bee@home:~/source_OS$ ls -l
total 8
drwxrwxr-x 2 bee bee 4096 лис 29 12:08 labs
drwxrwxr-x 2 bee bee 4096 лис 29 12:08 lects
```

Пояснення:

–перший файл ***labs*** є директорією;

–права доступу ***labs 775*** — всі дозволи для власника, всі дозволи для групи власника та інші користувачі мають право на

читання і вхід у директорію;

–кількість жорстких посилань на директорію – 2. Коли створюється нова порожня директорія, для неї створюється 2 жорсткі посилання: одне ім'я міститься у батьківській директорії, інше — в самій створеній директорії (***«.»***). Якщо директорія містить піддиректорії, кількість її жорстких зв'язків збільшується на 1 (запис про неї ***«..»*** міститься у дочірній піддиректорії). Жорсткими посиланнями на директорії оперує виключно ОС;

–власник директорії (***bee***);

–група власника (***bee***);

–розмір директорії (***4096b***);

–дата та час створення директорії;

–назва директорії (***labs***).

Щоб переглянути приховані файли, використовуємо опцію ***-a***:

```
bee@home:~/source_OS$ ls -la
total 20
drwxrwxr-x  5 bee bee 4096 лис 29 12:08 .
drwxr-xr-x 19 bee bee 4096 лис 29 12:08 ..
drwxrwxr-x  2 bee bee 4096 лис 29 12:08 .hidden
drwxrwxr-x  2 bee bee 4096 лис 29 12:08 labs
drwxrwxr-x  2 bee bee 4096 лис 29 12:08 lects
```

1.1.2. У директорії **labs** створюємо файли **labN** і **lab(N+1)**, записуючи в кожен файл рядок із назвою файлу (**lab N**, **lab(N+1)** відповідно):

```
bee@home:~/source_OS$ cd labs
bee@home:~/source_OS/labs$ echo labN > labN
bee@home:~/source_OS/labs$ echo 'lab(N+1)' > lab\(N+1\)
```

Перевірка:

```
bee@home:~/source_OS/labs$ cat labN
labN
bee@home:~/source_OS/labs$ cat lab\(N+1\)
lab(N+1)
```

1.1.3. Знаходячись у директорії **labs**, створюємо в **lects** файли **lectN**, **lect(N-1)**, використовуючи відносні шляхи:

```
bee@home:~/source_OS/labs$ touch ../lects/lectN ../lects/lect\(N-1\)
```

Перевірка:

```
bee@home:~/source_OS/labs$ ls -l ../lects
total 0
-rwxrwxrwx 1 bee bee 0 тра 11 12:15 lectN
-rwxrwxrwx 1 bee bee 0 тра 11 12:15 lect(N-1)
```

1.1.4. Переходимо у директорію **.hidden**

cd ../.hidden

Щоб переглянути дерево директорій, потрібно встановити утиліту ***tree***:

yum -y install tree.

Створюємо файли **.userN**, **.treeN**, записуючи в файл логін користувача (для **.userN**) та дерево директорій **source_OS** (для **.treeN**).

```
bee@home:~/source_OS/.hidden$ whoami > .userN
bee@home:~/source_OS/.hidden$ tree -a /home/bee/source_OS > .treeN
```

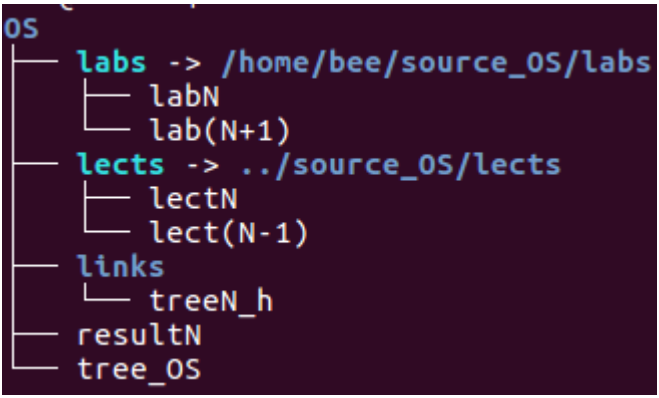
Перевірка:

```
bee@home:~/source_OS/.hidden$ ls -la
total 16
drwxr-xr-x 2 bee bee 4096 бер 26 15:35 .
drwxr-xr-x 5 bee bee 4096 бер 26 16:36 ..
-rw-rw-r-- 1 bee bee  237 бер 26 16:43 .treeN
-rw-rw-r-- 1 bee bee    4 бер 26 16:43 .userN
bee@home:~/source_OS/.hidden$ cat .userN
bee
bee@home:~/source_OS/.hidden$ cat .treeN
/home/bee/source_OS
├── .hidden
│   ├── .treeN
│   └── .userN
├── labs
│   ├── labN
│   └── lab(N+1)
└── lects
    ├── lectN
    └── lect(N-1)
```

1.2. У домашній директорії користувача створюємо директорію **OS**

mkdir ~/OS

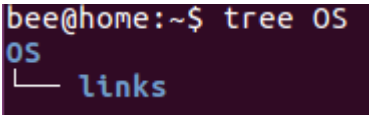
та наповнюємо її:



1.2.1. Створюємо директорію **OS/links**:

mkdir ~/OS/links

Перевірка:



1.2.2. В OS створюємо символічні посилання на директорію **/home/bee/source_OS/labs** (за повним шляхом), **/home/bee/source_OS/lects** (із використанням відносного шляху):

```
bee@home:~/OS$ ln -s /home/bee/source_OS/labs labs
bee@home:~/OS$ ln -s ../source_OS/lects lects
```

Зверніть увагу, що розмір файлу-посилання – довжина шляху до відповідного об’єкта.

```
bee@home:~/OS$ ls -l
total 4
lrwxrwxrwx 1 bee bee   24 лис 29 12:29 labs -> /home/bee/source_OS/labs
lrwxrwxrwx 1 bee bee   18 лис 29 12:29 lects -> ../source_OS/lects
drwxrwxr-x 2 bee bee 4096 лис 29 12:27 links
```

1.2.3. У файл результатів **resultN** розмістимо ім’я користувача та номер варіанта:

```
bee@home:~/OS$ cat ../source_OS/.hidden/.userN > resultN
bee@home:~/OS$ echo 'N' >> resultN
```

Перевірка:

```
bee@home:~/OS$ cat resultN
bee
N
```

1.2.4. Додамо у файл результатів **resultN** зміст попередньо створених файлів **labN**, **lab(N+1)**, скориставшись символічним зв'язком на файл та іменем файлу.

Спершу додамо 2 розділювача рядків (***важливо для автоматичної перевірки лабораторної роботи***):

```
bee@home:~$ echo -e >> resultN && echo -e >> resultN
bee@home:~/OS$ cat labs/labN >> resultN
bee@home:~/OS$ cat ../source_OS/labs/lab\(N+1\) >> resultN
```

1.2.5. У **links** створюємо файл **treeN_h**, який є жорстким посиланням на файл **treeN**:

```
bee@home:~$ ln source_OS/.hidden/.treeN OS/links/treeN_h
```

Перевірка:

```
bee@home:~$ cat OS/links/treeN_h
/home/bee/source_OS
├── .hidden
│   ├── .treeN
│   └── .userN
├── labs
│   ├── labN
│   └── lab(N+1)
└── lects
    ├── lectN
    └── lect(N-1)
```

1.2.6. Перевіримо, що файли пов'язані жорстким зв'язком, вказують на одній й ті ж дані. Побачити це можна, порівнявши номери індексних дескрипторів цих файлів. Для цього проведемо пошук у рекурсивному виведенні вмісту директорій **OS**, **source_OS** (повне виведення, перегляд прихованих файлів, виведення індексного дескриптору):

```
bee@home:~$ ls -laiR OS source_OS | grep treeN
283485 -rw-rw-r-- 2 bee bee  237 бер 26 16:43 treeN_h
283485 -rw-rw-r-- 2 bee bee  237 бер 26 16:43 .treeN
```

Допишемо цей результат у файл **resultN**, спершу додавши 2 розділювачі рядків:

```
bee@home:~$ echo -e >> resultN && echo -e >> resultN
bee@home:~$ ls -laiR OS source_OS | grep treeN >> OS/resultN
```

1.2.7. У директорії **OS** створюємо файл **tree_OS**, який містить дерево директорій **OS** (опція **-l** в утиліті **tree** дозволяє відображати вміст директорії, на яку вказує символічне посилання):

```
bee@home:~$ tree -l OS > OS/tree_OS
```

Перевірка:

```
OS
├── labs -> /home/bee/source_OS/labs
│   ├── labN
│   └── lab(N+1)
├── lects -> ../source_OS/lects
│   ├── lectN
│   └── lect(N-1)
├── links
│   └── treeN_h
├── resultN
└── tree_OS
```

Завдання 2. Розробка простого скрипта

Варіанти завдань

Завдання 1

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я користувача, якщо зазначений користувач не зареєстрований в системі, то виводить повідомлення про це та повторно запитує ім’я користувача;
- запитує ім’я каталогу;
- виводить кількість звичайних файлів, власником яких є користувач;

Для виконання завдання використовуйте команди ***grep***, ***find***.

Завдання 2

Розробити скрипт, який:

- запитує ім’я користувача;
- якщо зазначений користувач не зареєстрований в системі, то виводить повідомлення про це та повторно запитує ім’я користувача;
- якщо зазначений користувач зареєстрований у системі, то виводить його унікальний ідентифікатор (UID) і імена груп, до яких входить цей користувач, розділені пробілом.

Завдання 3

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує шлях до першого файлу;
- підраховує кількість слів у першому файлі (команда ***wc***);
- запитує шлях до іншого файлу;
- записує у другий файл обчислену кількість слів;
- якщо другий файл існує і містить дані, то запитує у користувача підтвердження на запис у файл.

Завдання 4

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку і знову запитує ім’я файлу;
- запитує ім’я користувача;
- виводить права даного користувача до файлу в форматі: ЧИТАТИ/ПИСАТИ/ВИКОНУВАТИ.

Завдання 5

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку і знову запитує ім’я файлу;
- запитує ім’я користувача;
- якщо користувач не є власником файлу, то виводить ім’я власника файлу і ім’я файлу групи.

Завдання 6

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує перше розширення файлу;
- запитує друге розширення файлу;
- знаходить всі файли у поточному каталозі з першим розширенням і перейменовує їх так, що змінює їх розширення на друге розширення;
- якщо таких файлів не існує, виводить повідомлення про помилку і починає спочатку.

Завдання 7

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку і знову запитує ім’я файлу;
- запитує дату;
- визначає, чи здійснювався доступ до файлу після зазначеної дати, і виводить повідомлення про це;
- якщо доступ до файлу не здійснювався, завершується з кодом 120.

Завдання 8

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку і знову запитує ім’я файлу;
- запитує дату;
- визначає, змінювалося вміст файлу після зазначеної дати, і виводить повідомлення про це;
- якщо вміст файлу змінювалося, завершується з кодом 120.

Завдання 9

Розробити скрипт, який:

- запитує тип дії: пошук по імені файлу або пошук за розміром;
- запитує каталог, в якому потрібно знайти файл;
- запитує ім’я файлу або розмір в залежності від необхідного дії;
- виводить всі файли з заданим ім’ям або всі файли більше зазначеного розміру в залежності від необхідного дії;

Для виконання завдання використовуйте команду ***find***.

Завдання 10

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку і знову запитує ім’я файлу;
- запитує дату;
- визначає, чи змінювався індексний дескриптор файлу після зазначеної дати, і виводить повідомлення про це;
- якщо індексний дескриптор файлу змінювався, завершується з кодом 120.

Завдання 11

Розробити скрипт, який:

- запитує тип дії: показати поточний каталог, піднятися на каталог вище, перейти в каталог;
- якщо обрано дію перейти в каталог, вивести список варіантів і дозволити користувачеві вибрати один з них;
- якщо обрано дію піднятися на каталог вище, то створити в ньому файл і розглянути його права доступу, модифікувати їх.

Завдання 12

Розробити скрипт, який:

- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку;
- якщо файл існує, виводить інформацію щодо файлу у форматі:
  - ім’я файлу (не включаючи шлях до файлу);
  - тип файлу;
  - розмір файлу;
  - власник файлу;
  - права доступу;
  - дата створення файлу;

Для виконання завдання використовуйте команди ***ls*** і ***stat***.

Завдання 13

Розробити скрипт, який:

- запитує тип операції над файлом: створити, видалити, перемістити;
- для створення файлу запитує ім’я нового файлу;
- якщо файл з таким ім’ям існує, виводить повідомлення про помилку;
- для видалення або переміщення файлу запитує ім’я файлу;
- якщо файл із таким ім’ям не існує, виводить повідомлення про помилку;
- для переміщення додатково запитує шлях до каталогу, в який потрібно перемістити файл;
- якщо каталог не існує, виводить повідомлення про помилку.

Завдання 14

Розробити скрипт, який:

- запитує шлях до каталогу;
- запитує розмір файлу;
- знаходить всі файли більше заданого розміру і видаляє їх, попередньо питаючи підтвердження користувача;

Для виконання завдання використовуйте програму ***find***.

Завдання 15

Розробити скрипт, який:

- запитує шлях до файлу;
- для даного файлу виводить порядково тимчасові мітки у форматі:
- час останнього доступу;
- час останньої зміни;
- час зміни індексного дескриптора.

Для виконання завдання використовуйте команди ***ls*** или ***stat***.

Завдання 16

Розробити скрипт, який:

- виводить ім’я поточного каталогу;
- запитує ім’я файлу;
- якщо файл не існує, виводить повідомлення про помилку і знову запитує ім’я файлу;
- знаходить у файловій системі жорсткі посилання на даний файл та виводить їх імена.

Завдання 17

Розробити скрипт, який:

- запитує шлях до першого файлу;
- запитує шлях до другого файлу;
- встановлює права доступу до другого файлу за зразком першого файлу.

Завдання 18

Розробити скрипт, який:

- запитує шлях до першого файлу;
- запитує шлях до другого файлу;
- визначає, чи введені імена файлів жорсткими посиланнями на один і той же файл;
- якщо не є, виводить повідомлення про помилку в потік **stderr**;
- якщо на останній ітерації виводилося повідомлення про помилку і користувач закінчив роботу скрипта, завершується з кодом 250.

Завдання 3. Розробка скрипта з параметрами

1. Написати скрипт, який буде запускатися у наступному форматі

./scrname.sc par1 par2 par3 par4

2. Параметр **par1** визначає варіант роботи скрипта.

a. Варіант 1 (**par1=1**). Створення файлу.

У файл з іменем name**{PID скрипта}** записати **par2** (до 10) записів. Кожен запис повинен починатися заголовком «**Note {номер запису}**». Перший запис повинен складатися з перших **par5** рядків man-сторінки команди **par3** (**par5** запитується при виконанні скрипта). Всі інші записи (i=1.. (par2-1)) повинні складатися з **(par6*10+{Номер варіанту})** починаючи з i-го рядка man-сторінки команди **par3**. Параметр **par6** визначається по таблиці 1.

b. Варіант 2 (**par1=2**). Пошук інформації.

Створіть фай, що містить інформацію довідки команди **ls**. В створений файл додайте різну інформацію, що відповідає формату для пошуку заданому у таблиці 2 (**«Шаблон пошуку»**). Необхідно додати не менше 15 слів з різним розташуванням по тексту. За необхідності додайте до доданих рядків інформацію для виконання умови стовпця **«Для запису»**. У файл з іменем **name{PID скрипта}** записати інформацію зазначену в стовпці **«Для запису»** таблиці 2. Для пошуку рядків використовуйте команду grep.

Завдання

Створіть фай, що містить інформацію довідки команди **ls**. В створений файл додайте різну інформацію, що відповідає формату для пошуку заданому у таблиці (**«Шаблон пошуку»**). Необхідно додати не менше 15 слів з різним розташуванням по тексту та 10 слів за схожим форматом. За необхідності додайте до доданих рядків інформацію для виконання умови для редагування зазначеній у стовпці **«Для запису»**. Напишіть

скрипт, який у створеному файлі знаходить рядки з інформацією, що відповідає **«Шаблону пошуку»**. Знайдені рядки необхідно зберегти у файлі з іменем **name{PID скрипта}** та провести редагування відповідно умові зазначеній у стовпці **«Для запису»**. Для пошуку рядків використовуйте команду grep. Під час виконання скрипта передбачте перевірку правильності введення параметрів.

с. Варіант 3 (**par1=3**). Пошук файлів. В директорії **par2** знайдіть всі файли, що відповідають критерію пошуку вказаному в таблиці 3. Для цього створіть директорію та додайте у нього файли, які будуть або не будуть відповідати формату для пошуку файлів. В файл статистики пошуку **stat** для кожного зі знайдених файлів запишіть наступну інформацію: **«Ім'я файлу» «Параметр»**. **«Параметр»** файлу вказано в таблиці 3. Після цього кожен зі знайдених файлів скопіюйте в директорію **par3**. Після виконання скрипта поверніть кількість знайдених файлів.

3. Під час виконання скрипта передбачте перевірку правильності введення параметрі

Таблиця 1

Номер варіанту	**Алгоритм обчислення**
1	1+3*(Перша цифра UID користувача з номером par4. Файл /etc/passwd)
2	2+2*(Остання цифра GID групи з номером par4. Файл /etc/group)
3	3+2*(Кількість цифр в UID користувача з номером par4. Файл /etc/passwd)
4	4+4*(Кількість літер у імені оболонки користувача з номером par4. Файл /etc/passwd)
5	5+(Кількість символів рядка з номером par4. Файл /etc/passwd)
6	6+4*(Остання цифра UID користувача з номером par4. Файл /etc/passwd)
7	7+2*(Кількість цифр GID групи з номером par4. Файл /etc/group)
8	8+4*(Кількість літер у назві домашньої директорії користувача з номером par4. Файл /etc/passwd)
9	9+3*(Кількість цифр в GID користувача з номером par4. Файл /etc/passwd)
10	10+3*(Перша цифра GID групи з номером par4. Файл /etc/group)
11	11+2*(Перша цифра GID користувача з номером par4. Файл /etc/passwd)
12	12+2*(Кількість літер в назві групи з номером par4. Файл /etc/group)
13	13+4*(Кількість літер у логіні користувача з номером par4. Файл /etc/passwd)
14	14+2*(Остання цифра GID користувача з номером par4. Файл /etc/passwd)
15	5+2*(Кількість символів рядка з номером par4. Файл /etc/group)

Таблиця 2

Номер варіанту	Шаблон пошуку	Для запису
1	**2**	**3**
1	MAC адреса у форматі «35:12:AC:C1:54»	У кожному рядку зі знайденим «Шаблон пошуку» замінити слово «find» на слово «yes».
2	IP адресу у форматі «192.168.10.152»	Номер рядка та кількість слів у рядку де знаходиться «Шаблон пошуку»
3	Адресу поштової скриньки «stud.23@mail.ssu.ua»	З кожного рядка де було знайдено «Шаблон пошуку» виписати перше слово у рядку.
4	Номер телефону у форматі «+38-000-000-0000»	Обирає всі рядки де було знайдено шаблон з кількістю символів більше 20.
5	Час у форматі «14:25:22»	У кожному рядку де не було знайдено «Шаблон пошуку» замінити слово «find» на слово «yes».
6	Автомобільний номер у форматі «MA4002CB»	Зберегти рядки де було знайдено «Шаблон пошуку» окрім кожного третього, починаючи з першого.
7	Дату у форматі «13 July 2020»	З кожного рядка де було знайдено «Шаблон пошуку» виписати слова з другого по четверте у рядку.
8	Автомобільний номер у форматі «19MA4002»	У кожному рядку де не було знайдено «Шаблон пошуку» замінити слово «find» незалежно від регістру на слово «yes».

9	MAC адреса у форматі 35-12-AC-C1-54	У кожному рядку зі знайденим «Шаблон пошуку» замінити кожне четверте слово «find» на слово «yes».
1	**2**	**3**
10	Адресу сайту у форматі «sumdu.edu.ua»	Зберегти кожен третій рядок де було знайдено «Шаблон пошуку».
11	Серію паспорта у форматі «МА 026258»	З кожного рядка де не було знайдено «Шаблон пошуку» виписати четверте слово у рядку.
12	Дату у форматі «13-07-20»	Зберегти парні рядки де було знайдено «Шаблон пошуку».
13	Прізвище з ініціалами у форматі «Ivanov T.B.»	У кожному рядку де не було знайдено «Шаблон пошуку» замінити кожне четверте слово «find» на слово «yes».
14	Час у форматі «02:25:22 p.m.»	Зберегти непарні рядки де було знайдено «Шаблон пошуку».
15	Дату у форматі «13/07/2020»	З кожного рядка де не було знайдено «Шаблон пошуку» виписати слова з першого по п’яте у рядку. З кожного рядка де було знайдено «Шаблон пошуку» виписати слова з другого по четверте у рядку.

Таблиця 3

Номер варіанту	Критерій пошуку файлу	Параметр файлу
1	Містить «file» на початку імені	Права доступу власника файлу
2	Має права доступу 0654	Кількість рядків у файлі
3	Файли тільки для запису	Власник файлу
4	Містить «file» в імені	Номер inode файлу
5	Що не належать користувачу «stud»	Тип файлу
6	Не містить «file» наприкінці імені	Група власників файлу
7	Не мають права доступу 0654	Права доступу групи власників файлу
8	Не містить «file» на початку імені	Кількість слів у файлі
9	Що належать користувачу «stud»	UID власника файлу
10	З встановленим бітом SetUID	Час останнього доступу до файлу
11	Містить «file» наприкінці імені	Значення додаткових атрибутів доступу до файлу
12	Що належать групі «stud»	Права доступу всіх інших до файлу
13	Не містить «file» в імені	Кількість жорстких посилань на inode файлу
14	Файли тільки для читання	GID групи власників файлу
15	Що не належать групі «stud»	Час зміни файлу

СПИСОК ЛІТЕРАТУРИ

1. Основи роботи та програмування в операційній системі Linux: навчальний посібник /Н.О. Матвєєва, В.С. Хандецький, О.В. Спірінцева - Д.: ЛІРА, 2018.
2. Операційні системи: навч. посіб. / В. Г. Зайцев, І. П. Дробязко; КПІ ім. Ігоря Сікорського. – Київ: КПІ ім. Ігоря Сікорського, 2019. – 240 с.
3. Основи операційних систем. Навчальний посібник. – Черкаси: ЧНУ імені Богдана Хмельницького, 2018. – 524 с.
4. Налаштування та адміністрування офісних Linux-серверів https://yakim.org.ua/images/stories/articles/Linux-course-book-ukr-2019.pdf
5. Путівник по Linux. https://linuxguide.rozh2sch.org.ua.

Електронне навчальне видання

Методичні вказівки
до виконання розрахункової роботи
з дисципліни «**Мережеві операційні системи**»
для студентів
спеціальності 172 *«Телекомунікації та радіотехніка»*
денної форми навчання

Відповідальний за випуск А. С. Опанасюк
Редактор Н. З. Клочко
Комп'ютерне верстання О В. Д'яченка

Формат 60х84/16. Ум. друк. арк. 1,86. Обл.-вид.арк. 1,94.

Видавець і виготовлювач
Сумський державний університет,
вул. Р.- Корсакова, 2, м. Суми, 40007
Свідоцтво суб'єкта видавничої справи ДК № 3062 від 17.12.2007.