Міністерство освіти і науки України

Національний авіаційний університет

Факультет кібербезпеки, комп’ютерної та програмної інженерії

Кафедра інженерії програмного забезпечення

*Операційні системи*

**Лабораторна робота №7**

**«Робота з текстовими даними»**

Розробив: студент ПІ-322

Царук С.О.

Перевірив:

Лосєв Є.О.

Київ 2021

**Мета:** Визначення понять і аналіз роботи нижче названих об’єктів та процесів, які відбуваються при доступі процесів до файлів і каталогів.

**Завдання**

Опрацювання роботи команд:

1. Використання утиліт hexdump та strings.
2. Пере направлення стандартного виведення у файл.
3. Недеструктивне пере направлення стандартного виведення.
4. Пере направлення стандартного введення із файлу.
5. Стандартне виведення помилок.
6. Пере направлення стандартного виводу помилок.
7. Об’єднання стандартного виведення та стандартного виведення помилок.
8. Пере направлення в нікуди.
9. Простіший конвеєр.
10. Підрахунок файлів.
11. Вилучення окремого поля.
12. Пошук рядка у виведенні програми.
13. Пошук послань в файлі info.
14. Заміна символів.
15. Заміна за регулярним виразом.
16. Отримання упорядкованого списку за частотністю слововживань.
17. Пошук всіх файлів, які виконуються, і точно є сценаріями.

Зміст

[Хід роботи 4](#_Toc67499084)

[1. Використання утиліт hexdump та strings. 4](#_Toc67499085)

[2. Перенаправлення стандартного виведення у файл. 5](#_Toc67499086)

[3. Недеструктивне перенаправлення стандартного виведення. 5](#_Toc67499087)

[4. Перенаправлення стандартного виведення із файлу. 6](#_Toc67499088)

[5. Стандартне виведення помилок. 6](#_Toc67499089)

[6. Перенаправлення стандартного виводу помилок. 6](#_Toc67499090)

[7. Об’єднання стандартного виведення та стандартного виведення помилок. 7](#_Toc67499091)

[8. Перенаправлення в нікуди. 7](#_Toc67499092)

[9. Простіший конвеєр. 8](#_Toc67499093)

[10. Підрахунок файлів. 8](#_Toc67499094)

[11. Вилучення окремого поля. 9](#_Toc67499095)

[12. Пошук рядка у виведенні програми. 9](#_Toc67499096)

[13. Пошук послань в файлі info. 10](#_Toc67499097)

[14. Заміна символів. 10](#_Toc67499098)

[15. Заміна за регулярним виразом. 11](#_Toc67499099)

[16. Отримання упорядкованого списку за частотністю слововживань. 11](#_Toc67499100)

[17. Пошук всіх файлів, які виконуються, і точно є сценаріями. 12](#_Toc67499101)

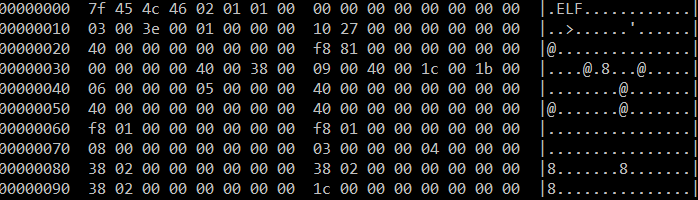
[Висновок 13](#_Toc67499102)

# Хід роботи

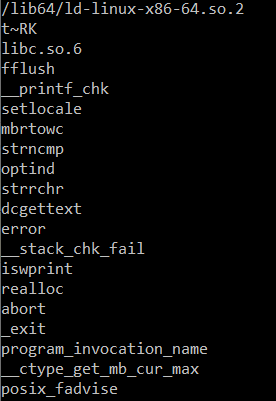
## Використання утиліт hexdump та strings.

Щоб переглянути вміст нетекстового файлу можна скористатися програмою ***hexdump***, яка дає вміст файлу у вигляді шістнадцятирічних ASCII-кодів, або ***strings***, яка показує тільки ті частини файлу, що можуть бути подані у вигляді тексту:



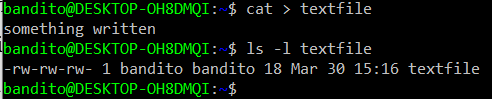






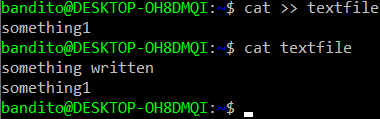
## Перенаправлення стандартного виведення у файл.

Команда ***cat*** у цьому прикладі не від командної оболонки ніяких параметрів: ні знака «***>***», наступного за ним імені файлу. У цьому випадку ***cat*** працювала як зазвичай, не знаючи, куди попадуть виведені дані на екран монітору, у файл чи куди ще. Замість того, щоби самій забезпечувати доставку виведення до кінцевого адресату, ***cat*** відправляє усі дані на ***стандартне виведення*** (скорочено - ***stdout***). Підміна стандартного виведення – завдання командної оболонки (***shell***). У даному прикладі ***shell*** утворює пустий файл, ім’я якого вказано після знаку «***>***», і дескриптор цього файлу передається програмі ***cat***  під номером ***1*** (***стандартне виведення***).



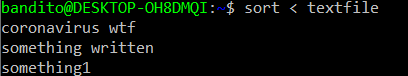
## Недеструктивне перенаправлення стандартного виведення.

Коли *закривається* стандартне виведення (дескриптор 1 звільнюється) і відкриває файл (з ним зв’язується *перший* вільний дескриптор, тобто 1), команді, яка запускається, нічого знати і не потрібно: її стандартне виведення вже підмінене. Ця операція називається *перенаправленням* стандартного виведення. У тому випадку, якщо файл уже є, ***shell*** запише його заново, повністю знищивши все, що в ньому вміщувалося до цього. Тому нам, щоби продовжити записувати дані в ***textfile***, знадобиться інша операція «***>>***».



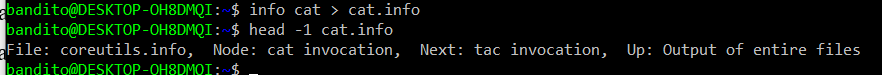
## Перенаправлення стандартного виведення із файлу.

Результат дії цієї команди зовсім аналогічний команді ***sort textfile***, різниця в тому, що коли використовується «***<***», ***sort*** отримує данні із стандартного введення, нічого не знаючи про «***textfile***», звідки вони поступають. Механізм роботи ***shell*** у даному випадку той же, що і при пере направленні виводу: ***shell*** читає дані з файлу «***textfile***», запускає утиліту ***sort*** і передає їй на стандартне введення вміст файлу.



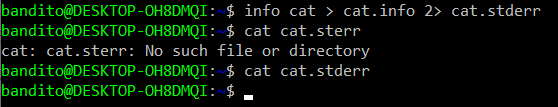
## Стандартне виведення помилок.

Ми винайшли, що незважаючи нашим вказівкам відправлятися у файл, два рядки, які виведено командою ***info***, все рівно проникнули на термінал. Очевидно, ці рядки не попали на ***стандартне виведення*** тому, що не відносяться безпосередньо до керівництва, яке повинна вивести програма, вони інформують користувача про хід *виконання* роботи: запис керівництва у файл. Для такого роду діагностичних повідомлень, а також для повідомлень про помилки, які виникнуть у ході виконання програми, в Linux передбачено ***стандартне виведення помилок*** (скорочено - ***stderr***).



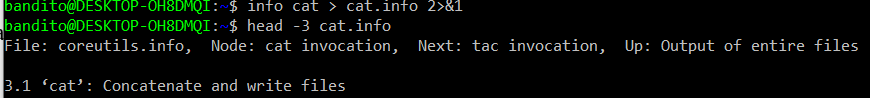
## Перенаправлення стандартного виводу помилок.

Стандартне виведення помилок може бути пере направлено також, як і стандартні введення/виведення, для цього треба використовувати комбінацію символів «***2>***». На цей раз на термінал вже нічого не попало, стандартне виведення відправилося у файл ***cat.info***, стандартне виведення помилок у – ***cat.stderr***. Замість «***>***» і «***2>***» ми могли би написати «***1>***» і «***2>***». Цифри у даному випадку позначають номери дескрипторів файлів, які *відкриваються*.



## Об’єднання стандартного виведення та стандартного виведення помилок.

Іноді, одначе, вимагається об’єднати стандартне виведення і стандартне виведення помилок в одному файлі, а не розділяти їх. У командній оболонці bash для цього є спеціальна послідовність «***2>&1***». Це означає – направити стандартне виведення помилок туди, куди і стандартне виведення:



## Перенаправлення в нікуди.

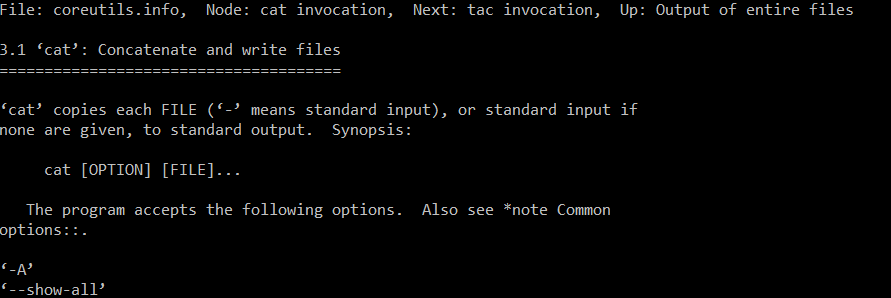
Іноді, наперед відомо, якісь дані, які виведені програмою, не знадобляться. Наприклад, попередження із стандартного виводу помилок. В цьому випадку, можна перенаправити стандартний вивід помилок у ***файл-дирку***, спеціально призначений для знищення даних - ***/dev/null***. Все, що записується у цей файл, просто буде *викинуто* і *ніде не зберігатися*.



## Простіший конвеєр.

Перенаправляти виведення можна не тільки у файл, а і *безпосередньо* на стандартне введення іншої програми. У цьому випадку замість двох команд буде потрібно тільки одну – програми передають одна одній дані «із рук у руки», В Linux такий засіб передавання даних називається **конвеєр**. У ***bash*** для пере направлення стандартного виведення на стандартне введення іншій програмі служить символ «***|***».





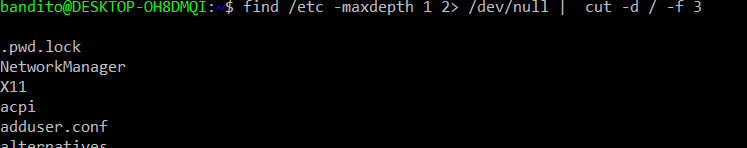
## Підрахунок файлів.

Ми отримали бажане число – «***20***». Для цього нам знадобилася команда ***find*** – яка рекомендувалася у керівництві, як інструмент потрібний необхідних файлів у системі. Ми выкликали ***find*** з одним параметром – каталогом, з якого починати пошук «***.***».***find*** виводить список знайдених файлів по одному на рядок, а оскільки критерії пошуку в даному випадку не уточнювалися, то ***find*** просто вивела список усіх файлів у всіх підкаталогах поточного каталогу (нашого домашнього каталогу). Цей список ми передали утиліті ***wc***, попрохавши її кількість отриманих рядків «***-l***», ***wc*** видала у відповідь число, яке ми шукали.



## Вилучення окремого поля.

Якщо користувача цікавить тільки визначена частина кожного рядка файлу – допоможе утиліта ***cut***. Припустимо, нам знадобилося отримати список всіх файлів і підкаталогів в «***/etc***». І при тому, нам треба, щоби файли, які будуть знайдені, у списку були представлені не повним шляхом, а тільки іменем файлу (скоріше за все, це вимагається для наступної автоматичної обробки цього списку).



## Пошук рядка у виведенні програми.

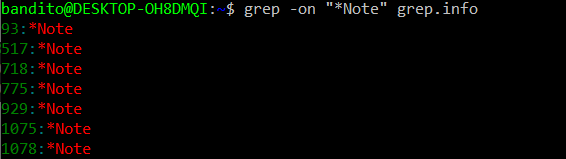
Часто користувачу треба знайти тільки нагадування чого-то конкретного серед даних, які виводяться утилітою, зазвичай ця задача зводиться до пошуку рядків. У яких зустрічається визначене слово або комбінація символів. Для цього підходить стандартна утиліта ***grep***. ***Grep*** може шукати рядок у файлах, а може працювати і як фільтр: отримавши рядки із стандартного введення, вона виведе на стандартне виведення тільки ті рядки, де зустрілося поєднання символів, які шукаються.





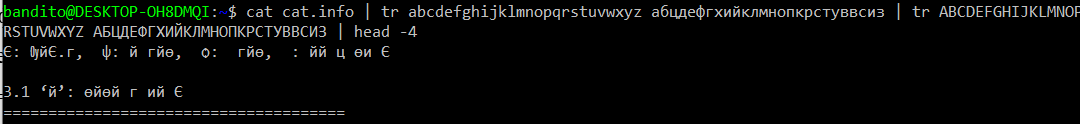
## Пошук послань в файлі info.

Комп’ютер цілком здібний виконати такий запит, якщо його сформулювати на строгій і зрозумілій мові ***регулярних виразів***. Регулярний вираз – це засіб однією формулою задати усі послідовності символів, які підходять користувачу. Перший параметр ***grep***, який взятий у лапки – це і є регулярний вираз для пошуку посилань у форматі info, другий параметр – ім’я файлу, у якому треба шукати. Ключ «***о***» примушує grep виводити рядок не повністю, а тільки ту частину, яка співпала з регулярним виразом (шаблон пошуку), а «***-n***» - виводить номер рядку, у якому зустрілося дане спів падіння.



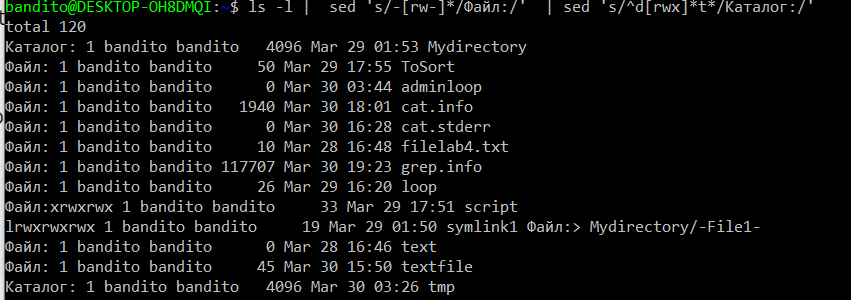
## Заміна символів.

Зручність роботи з потоком не в останню чергу складається в тому, що можна не тільки вибірково передавати результати роботи програм, але і автоматично замінювати один текст іншим прямо у потоці. Для заміни одних символів на інші призначена утиліта ***tr*** (скорочення від англійського «***translate***», «*перетворити, перевести*», яка працює як фільтр. Ми вирішили використати її прямо за призначенням і виконати за її допомогою транслітерацію – заміну латинських символів близькою за звучанням кирилицею.



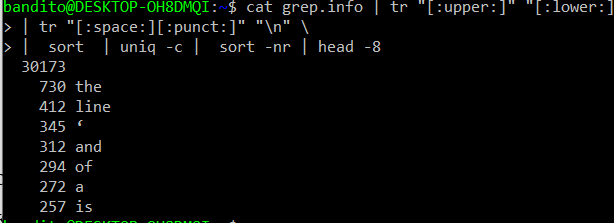
## Заміна за регулярним виразом.

Крім простої заміни окремих символів, можлива заміна послідовностей (слів). Спеціально для цього призначений потоковий редактор ***sed*** (скорочення від англійської «***stream editor***»). Він працює як фільтр і виконує редагування рядків, які поступають: заміну одних послідовностей символів на інші, причому, можна замінювати і ***регулярні вирази***. У ***sed*** дуже широкі можливості, але достатньо незвичний синтаксис, наприклад, виконується командою «***s/що\_замінювати /на\_що\_замінювати/***≫. Щоби у ньому розібратися, треба обов’язково прочитати керівництво ***sed(1)*** і знати регулярні вирази.



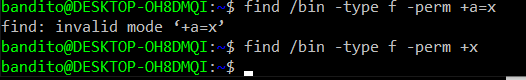
## Отримання упорядкованого списку за частотністю слововживань.

Комп’ютер по нашому плану підрахував, скільки разів які слова були вживані у файлі «***grep.info***» і вивів декілька самих частотних з вказівкою кількості вживань у файлі. Для цього знадобилося, спочатку, замінити всі великі літери маленькими, щоби не було різних засобів написання одного слова, потім, замінити всі пробіли і знаки перетину кінцем рядку (символ «***n***»), щоби у кожному рядку було рівно по одному слову. Ми всюди взяли параметри ***tr*** у лапки, щоби ***bash*** не зрозумів їх невірно. Потім список було відсортовано, всі слова, які повторюються, були замінені одним словом з вказівкою кількості повторів («***uniq -c***»), потім рядки знову відсортовано за зменшенням чисел на початку рядку («***sort -nr***») і виведені перші 8 рядків («***head -8***»).



## Пошук всіх файлів, які виконуються, і точно є сценаріями.

Оскільки в моїй встановленій версії Ubuntu не було виявлено в каталозі /bin фалів які виконуються, я шукав просто файли сценарії. Дані, які отримані у конвеєрі, можна перетворити на керівництво до дії для комп’ютера. Наприклад, для кожного отриманого із стандартного вводу рядка можна запустити яку-небудь команду, передавши їй цей рядок у якості параметру. Для цієї цілі служить утиліта ***xargs***.



# **Висновок**

В ході виконання даної лабораторної роботи я визначив понятття і аналіз роботи нижче названих об’єктів та процесів, які відбуваються при доступі процесів до файлів і каталогів. Було опрацьовано команди для роботи з текстовими даними. Було досліджено роботу з командами вводу і виводу.