Міністерство освіти і науки України

Національний авіаційний університет

Факультет кібербезпеки, комп’ютерної та програмної інженерії

Кафедра інженерії програмного забезпечення

*Операційні системи*

**Лабораторна робота №5**

**«Доступ процесів до файлів та каталогів»**

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. Перегляд таблиці особистих процесів.
2. Створення сценарію, який нескінченно виконується.
3. Запуск фонового процесу.
4. Розділення часу між процесами.
5. Перевід фонового процесу в активний стан.
6. Перевід процесу в фон за допомогою ^Z та bg.
7. Запуск декількох фонових процесів.
8. Примусове завершення процесу.
9. Завершення процесу природнім шляхом.
10. Доступ до файлу з обмеженим доступом.

Зміст

[Хід роботи 3](#_Toc67446059)

[1. Перегляд таблиці особистих процесів. 3](#_Toc67446060)

[2. Створення сценарію, який нескінченно виконується. 3](#_Toc67446061)

[3. Запуск фонового процесу. 3](#_Toc67446062)

[4. Розділення часу між процесами. 4](#_Toc67446063)

[5. Перевід фонового процесу в активний стан. 4](#_Toc67446064)

[6. Перевід процесу в фон за допомогою ^Z та bg. 5](#_Toc67446065)

[7. Запуск декількох фонових процесів. 5](#_Toc67446066)

[8. Примусове завершення процесу. 6](#_Toc67446067)

[9. Завершення процесу природнім шляхом. 6](#_Toc67446068)

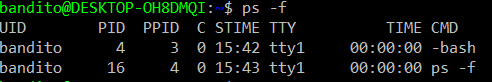
[10. Доступ до файлу з обмеженим доступом. 7](#_Toc67446069)

[Висновок 8](#_Toc67446070)

# Хід роботи

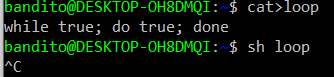
## Перегляд таблиці особистих процесів.

Програма, яка виконується, називається в Linux процесом. Усі процеси система реєструє у *таблиці процесів*, надаючи кожному унікальний номер – *ідентифікатор процесу* (*process identificator, PID*). Маніпулюючи процесами, система має справу саме з їх ідентифікаторами, іншого засобу відрізнити один процес від іншого, за великим рахунком, не має. Для перегляду своїх процесів можна скористуватися утилітою *ps* («*process status*»).



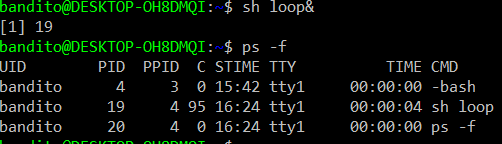
## Створення сценарію, який нескінченно виконується.

За порадою інструкції, ми утворили сценарій для ***sh*** (або ***bash***, на *такому* рівні їхні команди співпадають), який нічого не робить. Точніше було би сказати, що цей сценарій робить нічого, нескінченно повторюючи у циклі команду, вся робота якої складається у тому, що вона закінчується без помилок.



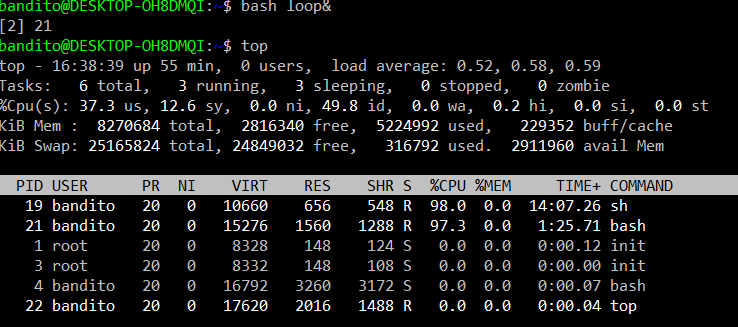
## Запуск фонового процесу.

Процес, який запускається паралельно. Називається фоновим («***background***»). Фонові процеси не мають можливості *вводити* дані з того ж терміналу, що і процес, який їх народив, ***shell*** (тільки з файлу).

****

## Розділення часу між процесами.

Далеко не усім процесам, зареєстрованим у системі, *на справді*, необхідно давати попрацювати нарівні з іншими. Більшості процесів працювати *прямо зараз* не потрібно: вони очікують якої-небудь події, яку їм треба обробити. Частіше за все. Процеси очікують завершення операції вводу-виводу. Щоби подивитися як використовуються ресурси системи, можна використати утиліту *top*.



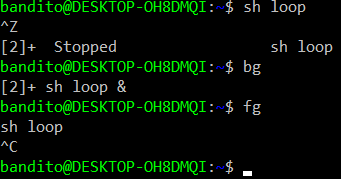
## Перевід фонового процесу в активний стан.

Процес, запущений з оболонки у фоні, можна з *цієї* оболонки зробити активним за допомогою команди ***fg*** («***foreground***» - «*передній план*»).



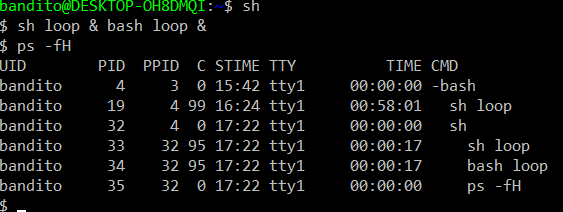
## Перевід процесу в фон за допомогою ^Z та bg.

Ми вирішили спочатку запустити вічний цикл у якості активного процесу, потім передали йому сигнал *STOP* за допомогою «***^Z***», після чого дали команду ***bg*** (***back ground***), яка запускає у фоні останній зупинений процес. Після цього, він знову перевів цей процес у активний режим, і на кінець, убив його.

****

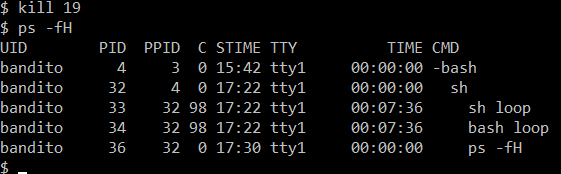
## Запуск декількох фонових процесів.

Ми вирішили позапускати процеси, а потім вибірково вбити їх. Для цього ми, на додаток до вже висячого у фоні процесу ***sh loop***, запустили у якості активного процесу новий командний інтерпретатор, ***sh*** (при цьому змінилося ***запрошення командного рядку***). Із цього shвін запустив у фоні ще один ***sh loop*** і новий ***bash loop***. Зробили ми це одним командним рядком (при цьому команди *розділяються* символом «***&***», тобто «І»; виходить так, що запускається ***і*** та , ***і*** інша команда). У ***ps*** ми використали новий ключ – «***Н***» («***Hierarchy***», «*ієрархія*»), який добавляє у видачу ***ps*** відступи, які показують відношення «*батько - нащадок*» між процесами.

****

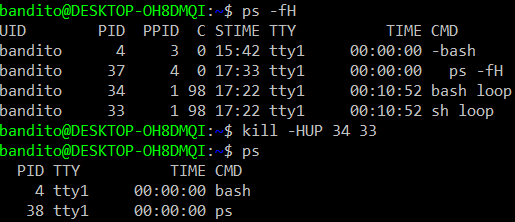
## Примусове завершення процесу.

Ми взялися завершувати процеси. Спочатку ми зупинили роботу давно запущеного sh, який виконував сценарій з вічним циклом (PID 19). Як видно з попереднього прикладу, цей процес за 17 хвилин роботи системи з’їв не менше 58 хвилин процесорного часу, і звісно, нічого корисного не зробив.

****

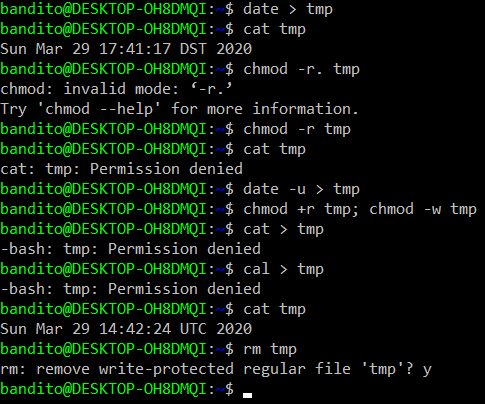
## Завершення процесу природнім шляхом.

*Чекати залишилося недовго. Цей* ***sh*** *завершився природнім шляхом, від команди* ***exit****, залишивши після себе двох дітей (****PID 33*** *i* ***34****), які відразу усиновив «батько усіх процесів» -* ***init*** *(****PID 1****). Коли ми розправилися і з ними – за сигналу* ***1*** *(****HUP****, тобто «****H****ang* ***UP****» - «повісити») нікому навіть повідомити про їх зникнення (якби процес-батько був живий, на зв’язаний з ним термінал вивелося би що-небудь на шталт «****[1]+ Hangup sh loop****»).*



## Доступ до файлу з обмеженим доступом.

Факт використання файлу процесом називається доступом до файлу, а засіб скористатися файлом (каталогом, посиланням і іншим) – видом доступу. Доступ до об’єкту можна змінити командою chmod (change mode, змінити режим (доступу)). У простих випадках формат цієї команди такий: chmod доступ об’єкт. Значення «+r» дозволяє доступ до об’єкту на читання (read), «-r» – забороняє. Аналогічно «+w», «-w», «+x», і «-x» дозволяють і забороняють доступ на запис (write) і використання (execute).



# **Висновок**

В ході виконання даної лабораторної роботи я визначив поняття і аналіз роботи нижче названих об’єктів та процесів, які відбуваються при доступі процесів до файлів і каталогів. Було досліджено роботу з доступом процесів до файлів та каталогів.