МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ІКНІ
Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи №4
на тему: “Створення та керування процесами в операційній системі Linux.”
з дисципліни: “Операційні системи”

Лектор:
старший викладач кафедри ПЗ
Грицай О.Д.

Виконав:
студент групи ПЗ-24
Губик А. С.

Прийняв:
доцент кафедри ПЗ
Горечко О. М.

Львів – 2023

Тема роботи:Створення та керування процесами в операційній системі Linux.

Мета роботи:Ознайомитися з паралельним виконанням процесів в операційній системі Linux, дослідити прискорення виконання завдання, через розпаралелення на процеси. Навчитися створювати процеси в операційній системі Linux, керувати станом та моніторити їх основні параметри.

Теоретичні відомості

Створення і запуск нових процесів в операційній системі Linux має свої особливості. Одразу після самозавантаження ядра, що відновлюється із стиснутого стану, запускається процес ініціалізації системи init або systemd через копіювання ядра системним викликом fork та заміною образу створеного процесу системним викликом exec. Таким ж чином процес ініціалізації запускає всі необхідні служби і програми, утворюючи при цьому дерево процесів зі строгою ієрархією. Тобто всі нові процеси будуть створенні вже існуючим процесом (в крайньому випадку процесом init або systemd). Процес який ініціює новий процес називають батьківським (parent), а створений процес - дочірнім (child).

Отже, якщо виконання fork() успішне, то він повертає різні значення для батьківського і дочірнього процесу: 0 - для дочірнього та PID дочірнього процесу для батьківського. І дочірній і батьківський процес починають своє виконання відразу після виклику fork(). Якщо новий процес не вдалось створити, то повертається значення -1 в батьківський процес і встановлюється значення помилки в errno. Це може статися, якщо в системі замало ресурсів для створення нового процесу (наприклад перевищили максимально дозволену кількість процесів: RLIMITNPROC для getrlimit(), або бракує пам'яті), а також при тупикових ситуаціях, що можуть виникнути у ядрі.

Індивідуальне завдання

Функція: $\frac{x*x^2*x^3}{cosx}$

Програми для диспетчера задач: табуляція з другого завдання, пошук чисел фібоначі

Хід роботи

I. Ознайомитися з системними викликами для процесів. Опрацювати приклади з методичних рекомендацій.

Я ознайомився розділяти програму з виокремленням процесу на батьківський і дочірній та без нього, що буває якщо створення процесів виходить з-під контролю, як можна дізнатись інформацію про статус дочірнього процесу.

II. Табуляція функції

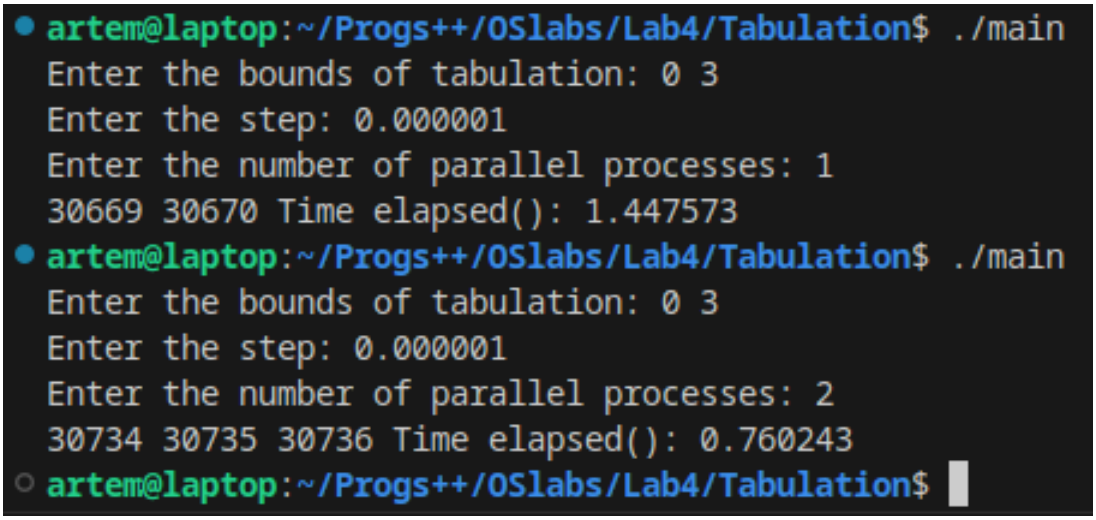

Рис. 1: Tabukation

Діапозон	Крок	К-сть процесів	Час виконання(s)
0 - 3	0.1	1	0.000715
0 - 3	0.1	2	0.001221
0 - 3	0.0001	1	0.018398
0 - 3	0.0001	2	0.009631
0 - 3	0.001	1	0.002205
0 - 3	0.001	2	0.001648
0 - 3	0.000001	1	1.447573
0 - 3	0.000001	2	0.760243
0 - 3	0.000001	6	0.341203

Як бачимо прискорення прямопроцірційне кількості процесів, на які розпаралелена функція.

III. Диспетчер задач

```
artem@laptop:~/Progs++/OSlabs/Lab4/TaskManager$ ./main
26589
26590
Enter command and PID: Warning: locale not supported by C library, locale unchanged
Warning: locale not supported by C library, locale unchanged
show 0
-----------------------------------------------
|  PID  |    Name    |  Status | Nice | U.Time |
| 26589 |  fibonacci | Running |    0 |  0.000 |
| 26590 | tabulation | Running |    0 |  0.000 |
-----------------------------------------------
Enter command and PID: priority 0

Enter new priority(-20 to 19): 5

Enter new priority(-20 to 19): 5
-----------------------------------------------
|  PID  |    Name    |  Status | Nice | U.Time |
| 26589 |  fibonacci | Running |    5 |  0.000 |
| 26590 | tabulation | Running |    0 |  0.000 |
-----------------------------------------------
Enter command and PID: pause 1
-----------------------------------------------
|  PID  |    Name    |  Status | Nice | U.Time |
| 26589 |  fibonacci | Running |    5 |  0.000 |
| 26590 | tabulation | Running |    0 |  0.000 |
-----------------------------------------------
Enter command and PID: show 0
-----------------------------------------------
|  PID  |    Name    |  Status | Nice | U.Time |
| 26589 |  fibonacci | Running |    5 |  0.000 |
| 26590 | tabulation | Stopped |    0 |  0.000 |
-----------------------------------------------
Enter command and PID: kill 1
-----------------------------------------------
|  PID  |    Name    |  Status | Nice | U.Time |
| 26589 |  fibonacci | Running |    5 |  0.000 |
| 26590 | tabulation | Stopped |    0 |  0.000 |
-----------------------------------------------
Enter command and PID: show 0
-----------------------------------------------
|  PID  |    Name    |  Status | Nice | U.Time |
| 26589 |  fibonacci | Running |    5 |  0.000 |
| 26590 | tabulation | Killed |   -1 |  0.010 |
-----------------------------------------------
Enter command and PID: sh
```

Рис. 2: TaskManager

Висновок: Я навчився створювати процеси і отримувати про них інформацію.