

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТУ  
“ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

**Кафедра систем штучного інтелекту**

**Лабораторна робота №8**

з дисципліни

«Операційні системи»

**Виконала:**

Студентка групи КН-  
214

Олескевич Софія

**Викладач:**

Кривенчук Ю.П.

Львів 2020

**Тема.** Робота з процесами в ОС Linux

**Мета.** Ознайомитися з багатопоточністю в ОС Linux. Навчитися працювати з процесами, в ОС Linux.

**Завдання.**

1. Виконати в окремому процесі табулювання функцій з лабораторної роботи №2.
2. Реалізувати табулювання функцій у 2-ох, 4-ох, 8-ох процесах. Виміряти час роботи процесів за допомогою API-функцій ОС Linux. Порівняти результати роботи в одному і в багатьох процесах.
3. Реалізувати можливість зміни пріоритету виконання процесу.
4. Порівняти результати виконання програми під ОС Windows та Linux.
5. Результати роботи відобразити у звіті.

### Варіант 19

**CODE:**

```
#include <iostream>
#include <iomanip>
#include <sys/types.h>
#include <sys/wait.h>
#include <cstring>
#include <unistd.h>
#include <sys/resource.h>
#include <cmath>
#include <vector>
using namespace std;

void function(double interval_start, double interval_end, double increment_step);

int main()
{
    int processes_num, steps_num=100000;

    cout << "Enter number of processes: "; cin >> processes_num;

    double interval_start=0, interval_end=0, interval=100000/processes_num,
    increment_step=0.00002;

    double* startValues = new double[processes_num];
```

```

clock_t start = clock();

double point = interval_start;

int pid = 1, curr_id = 0;

for (int i = 0; i < processes_num; i++)
{
    interval_start = ((interval * i) / 50000) - 1;
    interval_end = ((interval * (i + 1)) / 50000) - 1;
    if (pid != 0)
    {
        pid = fork();
        if(pid==0)
        {
            cout << "Process #" << getpid() << endl;
            curr_id = getpid() - getppid();
            curr_id -= 1;
            //cout << "Current ID: " << curr_id << endl;
            cout<<"from "<<interval_start<<" to "<<interval_end<<endl;
            function(startValues[curr_id], startValues[curr_id] + interval, increment_step);
        }
    }
    startValues[i] = point;
    point += interval;
}

```

```

int chose;

do{
    cout<<"1 Set priority of processes: "<<endl;
    cout<<"2 Kill processes: "<<endl;
    cout<<"3 Suspend:"<<endl;
    cout<<"4 Resume: "<<endl;
}

```

```

cin>>choise;
switch(choise)
{
    case 1:
        {
            cout << "Enter ID of the process: ";
            int id; cin >> id;
            cout << "Enter the priority (from -20 up to 19): ";
            int prior; cin >> prior;
            setpriority(PRIO_PROCESS,id,prior);
            break;
        }
    case 2:
        {
            cout << "Enter ID of the process: ";
            int id; cin >> id;
            kill(id,9);
            break;
        }
    case 3:
        {
            cout << "Enter ID of the process: ";
            int id; cin >> id;
            kill(id,SIGSTOP);
            break;
        }
    case 4:
        {
            cout << "Enter ID of the process: ";
            int id; cin >> id;
            kill(id,SIGCONT);

```

```

        break;
    }

}} while(choise!=5);

for (int i = 0; i < processes_num; i++)
    wait(NULL);

clock_t end = clock();

if (pid == 0) cout << "Execution time (process " << curr_id << "): " << double(end - start) /
double(CLOCKS_PER_SEC) << " seconds" << endl;

return 0;
}

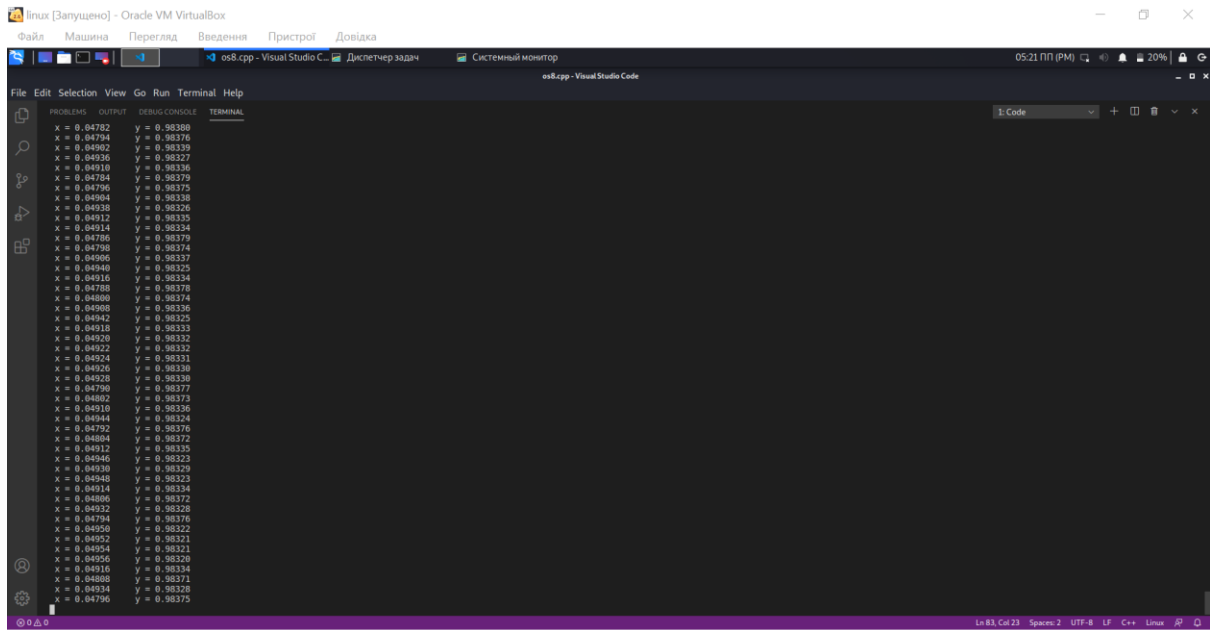
void function(double interval_start, double interval_end, double increment_step)
{
    for (double i = interval_start; i <= interval_end; i += increment_step)
    {
        sleep(0.75);

        cout << " x = " << setprecision(5) << fixed << i << "\t";

        cout << " y = " << setprecision(5) << fixed << pow((1 - i), (1.0 / 3)) << endl;
    }
}

```

## РЕЗУЛЬТАТИ



```
kali@kali:~$ cd "/home/kali/Рабочий стол/" && g++ os8.cpp -o os8 && "/home/kali/Рабочий стол/"os8
Enter number of processes: 5
1 Set priority of processes:
2 Kill processes:
3 Suspend:
4 Resume:
Process #4156
Process #4157
Process #4155
Process #4154
Process #4153
from 0.6 to 1
from 0.2 to 0.6
from -0.2 to 0.2
from -0.6 to -0.2
from -1 to -0.6
1
Enter ID of the process: 4155
Enter the priority (from -20 up to 19): 15
1 Set priority of processes:
2 Kill processes:
3 Suspend:
4 Resume:
2
Enter ID of the process: 4155
```

ЦП: 2%		Процессы: 148		Память: 49%		Подкачка: 4%	
Задача	PID	Состояние	Резидентная	ЦП	Приор.		
os8.cpp - Visual Studio Code	3955	S	125,7 MiB	0%	0		
os8	4136	S	1,5 MiB	0%	0		
os8	4153	S	188,0 KiB	10%	0		
os8	4154	S	188,0 KiB	8%	0		
os8	4156	S	188,0 KiB	10%	0		
os8	4157	S	188,0 KiB	8%	0		
[os8]	4155	Z	0 байт	0%	19		

## Suspend #4157

ЦП: 4%		Процеси: 148		Пам'ять: 49%		Подача: 4%			
Задача		PID	Состояние	Резидентная	ЦП	Приор.			
osB.cpp - Visual Studio Code		3955	S	127.5 MiB	0%	0			
osB		4136	S	1.5 MiB	0%	0			
osB		4153	S	188.0 KiB	9%	0			
osB		4154	S	188.0 KiB	7%	0			
osB		4156	S	188.0 KiB	9%	0			
osB		4157	T	188.0 KiB	0%	0			
[osB]		4155	Z	0 байт	0%	15			

## Resume #4157

ЦП: 6%		Процеси: 147		Пам'ять: 49%		Подача: 4%			
Задача		PID	Состояние	Резидентная	ЦП	Приор.			
osB.cpp - Visual Studio Code		3955	S	127.8 MiB	0%	0			
osB		4136	S	1.5 MiB	0%	0			
osB		4153	S	188.0 KiB	10%	0			
osB		4154	S	188.0 KiB	10%	0			
osB		4156	S	188.0 KiB	10%	0			
osB		4157	S	188.0 KiB	10%	0			
[osB]		4155	Z	0 байт	0%	15			

## ВИСНОВОК:

У ході роботи над цією лабораторною, я здобула навички створення процесів, зміни їх пріорітету, зупинки, відновлення та вбиття.