

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

Кафедра систем штучного інтелекту

Лабораторна робота №10

з дисципліни
«Операційні системи»

Виконала:

Студентка групи КН-
214

Олескевич Софія

Викладач:

Кривенчук Ю.П.

Лабораторна робота №10

Тема. Синхронізація потоків в ОС Linux

Мета. Ознайомитися з особливостями синхронізації потоків в ОС Linux.
Навчитися організовувати багатопоточність з використанням синхронізації в ОС Linux.

Завдання.

1. Реалізувати алгоритм із лабораторної роботи №3.
2. Здійснити розпаралелювання даного алгоритму на 2, 4, 8 потоків із використанням синхронізації.
3. Реалізувати прогрес (хід) виконання задачі.
4. Для синхронізації потоків використати такі методи: м'ютекси, семафори, умовна змінна, очікування, сигнали, монітори.
5. Порівняти результати виконання програми під ОС Windows та Linux.
6. Результати виконання роботи відобразити у звіті. Індивідуальні завдання. Відповідно до вказаного варіанту реалізувати два механізми синхронізації:

Варіант № 1-12 – м'ютекс

Варіант № 13-25 – **семафор**

Варіант № 1-6 – умовна змінна

Варіант № 7-12 – очікування Варіант

№ 13-18 – сигнали Варіант

№ 19-25 – **монітори**

Варіант 19

CODE:

```
#include <iostream>
#include <stdio.h>
#include <sys/types.h>
```

```

#include <sys/stat.h>
#include <dirent.h>
#include <unistd.h>
#include <vector>
#include <pthread.h>
#include <mutex>
#include <semaphore.h>

using namespace std;

int cnt=0, files=0, catalogs=0, per_thread, size0, num, pr, choise;

vector < pair<string,string> > names;
vector <long> si;
vector <string> res;
sem_t semaphore;
mutex m;

void *sort(void* i)
{
    if(choise==2){ lock_guard<mutex> _{m}; }

    long n;
    int l = n * per_thread, r = (n + 1) * per_thread;
    if(n == per_thread)
        r = names.size();

    if(choise==1){ sem_wait(&semaphore); }

    cout << "Thread with ID " << pthread_self() << " has started!" << endl;
    cout << "Limits from " << l + 1 << " " << r << endl;

    if(choise==1){ sem_post(&semaphore); }

    for(int i=l; i<r; i++)
    {
        if(choise==1){ sem_wait(&semaphore); }

        if(si[i]==size0)

```

```

{
    res.push_back(names[i].first);
}

if(choise==1){ sem_post(&semaphore); }

}

sleep(5);

if(num==n){ nice(pr);}

sleep(5);

if(choise==1){ sem_wait(&semaphore); }

if(choise==2){ lock_guard<mutex>_m; }

cout << "Thread with ID " << pthread_self() << " has ended!" << endl;

if(choise==1){ sem_post(&semaphore); }

return NULL;
}

```

```

void find(string s, string no){

DIR *dir1;

struct dirent *info1;

struct stat a;

dir1 = opendir(s.c_str());

if(dir1 && no!="." && no!="..")

{

    names.push_back({s,"catalog"});

    catalogs++;

    while ((info1 = readdir(dir1)) != NULL) {

        string s1(info1->d_name);

        if (s1 != "." && s1 != "..") {

            stat(s.c_str(),&a);

            si.push_back(info1->d_reclen);

```

```

        cnt++;

        string s2 = s + "/";
        string s1(info1->d_name);

        s2 += s1;
        find(s2, s1);

        cnt--;
    }

}

}

}

else { names.push_back({s,"file"});}

closedir(dir1);

}

int main()
{
    find("/home","");
    cout << "Number of files and catalogs: " << names.size() << endl;
    cout << "All " << endl;
    for(auto x:names)
        cout<<x.first<<" "<<x.second<<endl;
    cout<<"Input size: "<<endl; cin>>size0;
    cout<<"Choose sem/monitor:"<<endl; cin>>choise;
    cout << "Enter the number of threads you would like to start: ";
    int th; cin >> th;
    pthread_t id[th];
    per_thread = names.size()/th;
    sem_init(&semaphore,0,1);
    for(int i=0;i<th;i++)

```

```

{
    pthread_create(&(id[i]), NULL, sort, (void*)i);

}

sleep(0.1);

cout << "Enter the number of process and the priority: ";

cin >> num >> pr;

for(int i=0;i<th;i++)

    pthread_join(id[i],NULL);

int b; cout<<"Input 0 to see results:"; cin >> b;

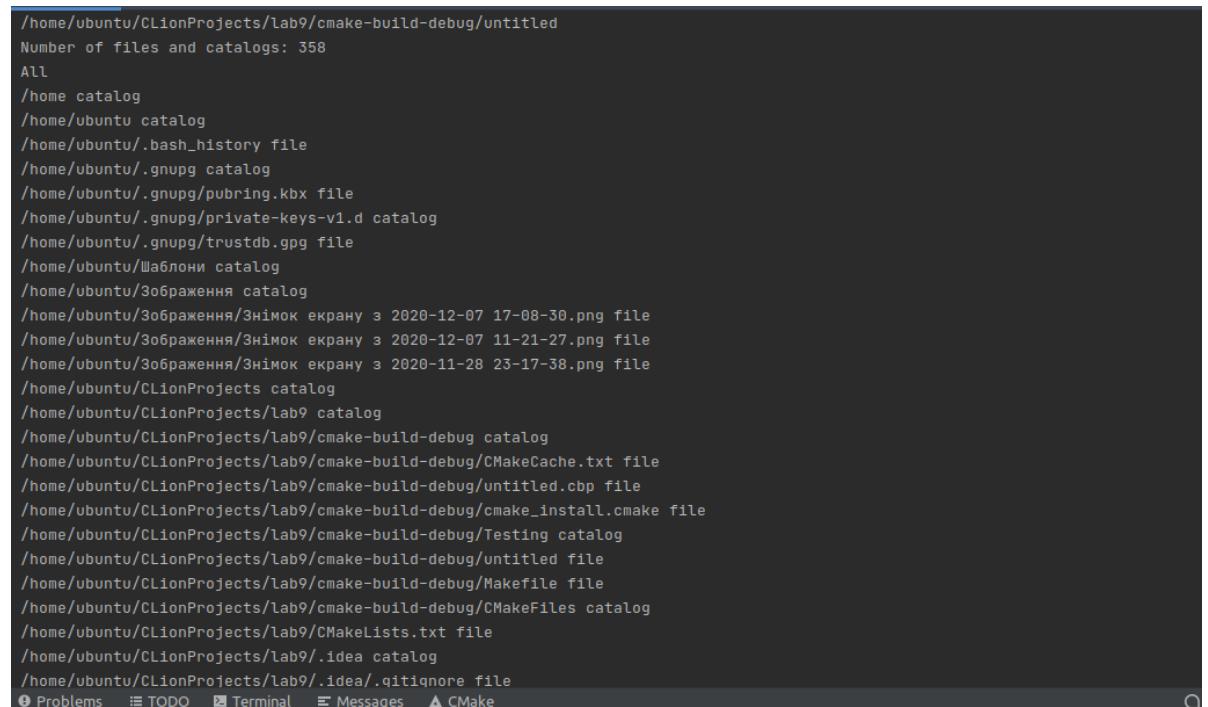
for(int i=0; i<res.size(); i++)

    cout<<res[i]<<endl;

return 0;
}

```

Results



```

/home/ubuntu/CLionProjects/lab9/cmake-build-debug/untitled
Number of files and catalogs: 358
All
/home catalog
/home/ubuntu catalog
/home/ubuntu/.bash_history file
/home/ubuntu/.gnupg catalog
/home/ubuntu/.gnupg/pubring.kbx file
/home/ubuntu/.gnupg/private-keys-v1.d catalog
/home/ubuntu/.gnupg/trustdb.gpg file
/home/ubuntu/Шаблони catalog
/home/ubuntu/Зображення catalog
/home/ubuntu/Зображення/знімок екрану з 2020-12-07 17-08-30.png file
/home/ubuntu/Зображення/знімок екрану з 2020-12-07 11-21-27.png file
/home/ubuntu/Зображення/знімок екрану з 2020-11-28 23-17-38.png file
/home/ubuntu/CLionProjects catalog
/home/ubuntu/CLionProjects/lab9 catalog
/home/ubuntu/CLionProjects/lab9/cmake-build-debug catalog
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/CMakeCache.txt file
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/untitled.cbp file
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/cmake_install.cmake file
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/Testing catalog
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/untitled file
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/Makefile file
/home/ubuntu/CLionProjects/Lab9/cmake-build-debug/CMakeFiles catalog
/home/ubuntu/CLionProjects/Lab9/CMakelists.txt file
/home/ubuntu/CLionProjects/Lab9/.idea catalog
/home/ubuntu/CLionProjects/Lab9/.idea/.qtiignore file

```

⌚ Problems ⚡ TODO 🗑 Terminal 📣 Messages 🔍 CMake

| тд..)

```
Input size:  
64  
Choose sem/monitor:  
1  
Enter the number of threads you would like to start: 4  
Enter the number of process and the priority: Thread with ID 139955056830208 has started!  
Limits from 179 267  
Thread with ID 139955048437504 has started!  
Limits from 268 356  
Thread with ID 139955065222912 has started!  
Limits from 90 178  
Thread with ID 139955073615616 has started!  
Limits from 1 89  
1 15  
Thread with ID 139955056830208 has ended!  
Thread with ID 139955048437504 has ended!  
Thread with ID 139955073615616 has ended!  
Thread with ID 139955065222912 has ended!  
Input 0 to see results:0  
/home/ubuntu/.java/.userPrefs/jetbrains/syncsettings  
/home/ubuntu/.config/ibus/bus  
  
Process finished with exit code 0
```

Htop:

```
3999 ubuntu 20 0 166M 3240 2956 S 0.0 0.1 0:00.10 | | |  
4027 ubuntu 20 0 166M 3240 2956 S 0.0 0.1 0:00.00 | | |  
4026 ubuntu 20 0 166M 3240 2956 S 0.0 0.1 0:00.00 | | |  
4025 ubuntu 35 15 166M 3240 2956 S 0.0 0.1 0:00.00 | | |  
4024 ubuntu 20 0 166M 3240 2956 S 0.0 0.1 0:00.00 | | |  
Next EscCancel Search: lab10
```

Висновок: у ході роботи над цією лабораторною, я реалізувала два методи синхронізації (семафор та монітор), реалізувала пошук файлу за розміром та знайшла інтервали кожного з них, перевірила зміну пріоритету в утиліті htop.