НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи №*1*

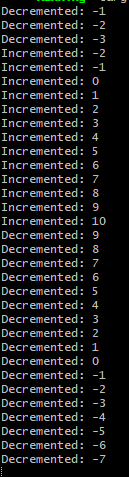
із дисципліни «Розподілені і хмарні обчислення»

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-01 | *Доцент кафедри ПМА*  *Ліскін В. О.* |
| *Романецький М.С.* |  |

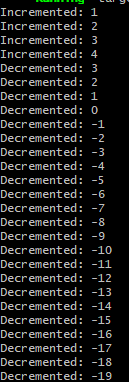
Київ — 2023

Мета роботи: навчитись працювати з потоками

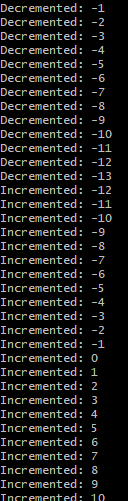
Опис програми: Для демонстрації роботи буде продемонстрована програма з користувацьким інтерфейсом інкремент/декримент. Користувач може змінювати пріоритет потоку на максимальний або мінімальний. Операції виконуються в різних потоках.



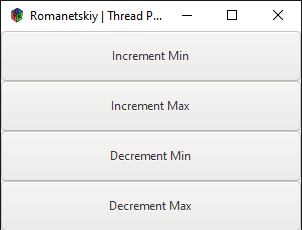
Рівний пріоритет



Пріоритет декременту



Пріоритет інкременту



Користувацький інтерфейс

Код програми:

*// cd /d/KPI/Distributed\_computing/Labs/lab\_1*

#[macro\_use]

extern crate lazy\_static;

use gtk::prelude::\*;

use gtk::{Button, Window, WindowType, Box, Orientation};

use std::sync::{Arc, Mutex};

use std::thread;

use std::time::Duration;

use thread\_priority::\*;

use std::os::windows::io::AsRawHandle;

use winapi::ctypes::c\_void as winapi\_c\_void;

lazy\_static! {

    static ref PRIORITY\_MIN: ThreadPriority = ThreadPriority::Os(WinAPIThreadPriority::Lowest.into());

    static ref PRIORITY\_MAX: ThreadPriority = ThreadPriority::Os(WinAPIThreadPriority::Highest.into());

}

fn increment(refr: Arc<Mutex<i32>>) {

    loop {

        let mut num = refr.lock().unwrap();

        \*num += 1;

        println!("Incremented: {}", num);

        thread::sleep(Duration::from\_millis(100));

    }

}

fn decrement(refr: Arc<Mutex<i32>>) {

    loop {

        let mut num = refr.lock().unwrap();

        \*num -= 1;

        println!("Decremented: {}", num);

        thread::sleep(Duration::from\_millis(100));

    }

}

fn main() {

    if gtk::init().is\_err() {

        println!("Failed to initialize GTK.");

        return;

    }

    let counter = Arc::new(Mutex::new(0));

    let window = Window::new(WindowType::Toplevel);

    window.set\_title("Romanetskiy | Thread Priority Control");

    window.set\_default\_size(300, 200);

    let vbox = Box::new(Orientation::Vertical, 0);

    let btn\_inc\_min = Button::with\_label("Increment Min");

    let btn\_inc\_max = Button::with\_label("Increment Max");

    let btn\_dec\_min = Button::with\_label("Decrement Min");

    let btn\_dec\_max = Button::with\_label("Decrement Max");

    vbox.pack\_start(&btn\_inc\_min, true, true, 0);

    vbox.pack\_start(&btn\_inc\_max, true, true, 0);

    vbox.pack\_start(&btn\_dec\_min, true, true, 0);

    vbox.pack\_start(&btn\_dec\_max, true, true, 0);

    window.add(&vbox);

    let counter\_clone\_inc = Arc::clone(&counter);

    let inc\_thread = thread::spawn(move || {

        increment(counter\_clone\_inc);

    });

    let counter\_clone\_dec = Arc::clone(&counter);

    let dec\_thread = thread::spawn(move || {

        decrement(counter\_clone\_dec);

    });

    let inc\_handle = inc\_thread.as\_raw\_handle() as \*mut winapi\_c\_void;

    let dec\_handle = dec\_thread.as\_raw\_handle() as \*mut winapi\_c\_void;

    btn\_inc\_min.connect\_clicked(move |\_| {

        set\_thread\_priority(inc\_handle, \*PRIORITY\_MIN).expect("Failed to set thread priority");

    });

    btn\_inc\_max.connect\_clicked(move |\_| {

        set\_thread\_priority(inc\_handle, \*PRIORITY\_MAX).expect("Failed to set thread priority");

    });

    btn\_dec\_min.connect\_clicked(move |\_| {

        set\_thread\_priority(dec\_handle, \*PRIORITY\_MIN).expect("Failed to set thread priority");

    });

    btn\_dec\_max.connect\_clicked(move |\_| {

        set\_thread\_priority(dec\_handle, \*PRIORITY\_MAX).expect("Failed to set thread priority");

    });

    window.show\_all();

    window.connect\_delete\_event(|\_, \_| {

        gtk::main\_quit();

        Inhibit(false)

    });

    gtk::main();

}