A drawing of a building

Description automatically generated

Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут»

**ЛАБОРАТОРНА РОБОТА №2.**

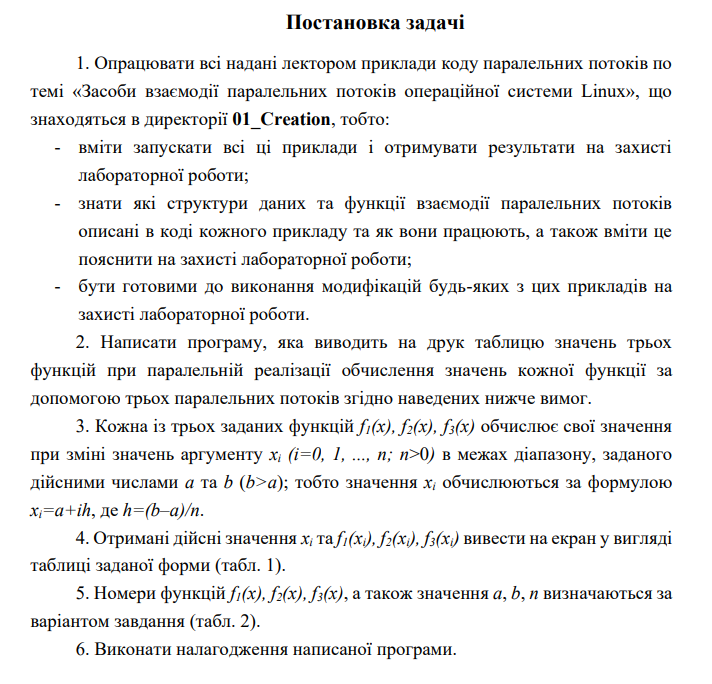
**СТВОРЕННЯ ПАРАЛЕЛЬНИХ ПОТОКІВ ,ОПЕРАЦІЙНОЇ**

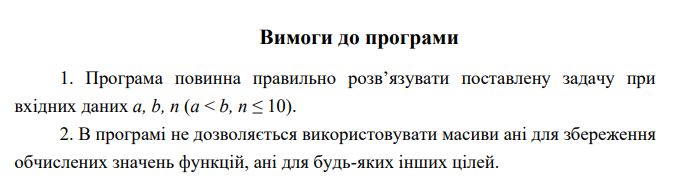
**СИСТЕМИ LINUX ТА НАЙПРОСТІША СИНХРОНІЗАЦІЯ**

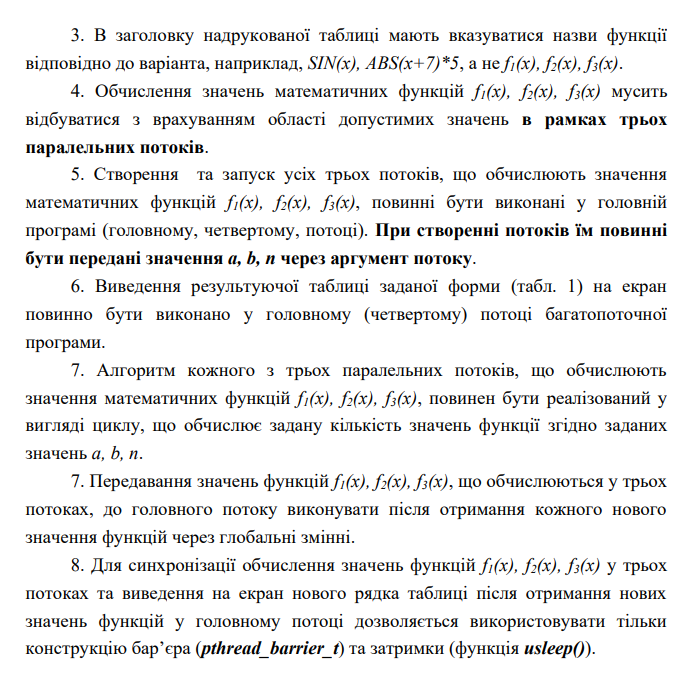
Виконав студент групи: КВ-22

ПІБ: Землянський Едуард

**Київ 2024**

****

****

****

**Варіант**

****

**Функції**

****

****

****

**Код:**

// variant 7. functions 2, 5, 11. a = -pi, b = pi, n = 8

#include <math.h>

#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

#include <iomanip>

#include <iostream>

#define M\_PI 3.14159265358979323846

struct arguments{

    float a;

    float b;

    int n;

};

pthread\_barrier\_t barrier;

double res1, res2, res3;

void \* func1(void \* arguments){

    struct arguments \* args = (struct arguments\*) arguments; //

    float step = (args -> b - args -> a) / args -> n;

    float argument = args -> a;

    for (int i = 0; i < args -> n; i++){

        usleep(1);

        argument = args -> a + i \* step;

        res1 = pow(cos(argument), 2) + sin(argument);

        pthread\_barrier\_wait(&barrier);

    }

    return NULL;

}

void \* func2(void \* arguments){

    struct arguments \* args = (struct arguments\*) arguments;

    float step = (args -> b - args -> a) / args -> n;

    float argument = args -> a;

    for (int i = 0; i < args -> n; i++){

        usleep(1);

        argument = args -> a + i \* step;

        res2 = pow(sin(argument), 2) \* (1 + cos(argument));

        pthread\_barrier\_wait(&barrier);

    }

    return NULL;

}

void \* func3(void \* arguments){

    struct arguments \* args = (struct arguments\*) arguments;

    float step = (args -> b - args -> a) / args -> n;

    float argument = args -> a;

    for (int i = 0; i < args -> n; i++){

        usleep(1);

        argument = args -> a + i \* step;

        res3 = (1 + pow(cos(argument), 2)) \* sin(argument);

        pthread\_barrier\_wait(&barrier);

    }

    return NULL;

}

int main(){

    struct arguments function\_args;

    function\_args.a = -M\_PI;

    function\_args.b = M\_PI;

    function\_args.n = 8;

    pthread\_t func1\_thread, func2\_thread, func3\_thread;

    pthread\_barrier\_init(&barrier, NULL, 4);

    pthread\_create(&func1\_thread, NULL, &func1, &function\_args); // creating and starting threads

    pthread\_create(&func2\_thread, NULL, &func2, &function\_args); // passing arguments

    pthread\_create(&func3\_thread, NULL, &func3, &function\_args);

    // printf("\tx\t|  cos(x)^2 + sin(x)  |  sin(x)^2 \* (1 + cos(x))\n");

    std::cout << std::setw(10) << "x" << std::setw(24) << "cos(x)^2+sin(x)" << std::setw(24) << "sin(x)^2\*(1+cos(x))" << std::setw(24) <<  "sin(x)\*(1+cos(x)^2)" << std::endl;

    float step = (function\_args.b - function\_args.a) / function\_args.n;

    float argument = function\_args.a;

    for (int i = 0; i < function\_args.n; i++){

        pthread\_barrier\_wait(&barrier);

        argument += step;

        std::cout << std::setw(10) << argument << std::setw(24) << res1 << std::setw(24) << res2 << std::setw(24) << res3 << std::endl;

    }

    pthread\_join(func1\_thread, NULL); // waiting threads to finish

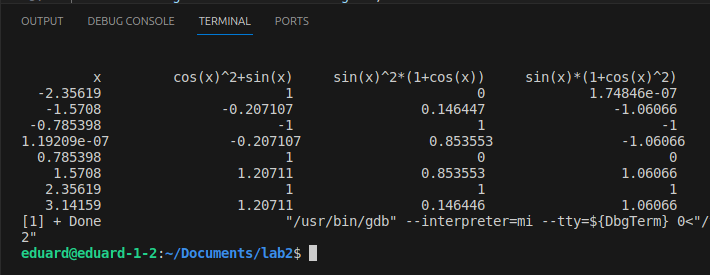
    pthread\_join(func2\_thread, NULL);

    pthread\_join(func3\_thread, NULL);

    return 0;

}

**Результат програми**

****